INSTITUTO DE PESQUISAS ENERGÉTICAS E NUCLEARES
Autarquia Associada à Universidade de São Paulo

ALGORITMO RASTREADOR WEB ESPECIALISTA NUCLEAR

THIAGO REIS

Dissertação apresentada como parte dos requisitos para a obtenção do grau de Mestre em Ciências na área de Tecnologia Nuclear – Reatores.

Orientador:
Dr. Antônio Carlos de Oliveira Barroso

São Paulo
2013

AGRADECIMENTOS

A minha família e a minha noiva Angelina pelo apoio, incentivo e compreensão.

A Comissão Nacional de Energia Nuclear (CNEN) e ao Instituto de Pesquisas Energéticas e Nucleares (IPEN) pela estrutura de ensino e pesquisa.

Aos Drs. Mário Olímpio de Menezes, Kengo Imakuma e Benedito Dias Baptista Filho pela participação e contribuições neste trabalho.

Ao Prof. Dr. Antônio Carlos de Oliveira Barroso pela oportunidade de realizar este trabalho e orientação ao longo do projeto.

E a todos aqueles que colaboraram para a criação, desenvolvimento e conclusão deste trabalho os meus sinceros agradecimentos.

ALGORITMO RASTREADOR WEB ESPECIALISTA NUCLEAR

Thiago Reis

RESUMO

Nos últimos anos a Web obteve um crescimento exponencial, se tornando o maior repositório de informações já criado pelo homem e representando uma fonte nova e relevante de informações potencialmente úteis para diversas áreas, inclusive a área nuclear. Entretanto, devido as suas características e, principalmente, devido ao seu grande volume de dados, emerge um problema desafiador relacionado à utilização das suas informações: a busca e recuperação informações relevantes e úteis. Este problema é tratado por algoritmos de busca e recuperação de informação que trabalham na Web, denominados **rastreadores web**.

Neste trabalho é apresentada a pesquisa e desenvolvimento de um algoritmo rastreador que efetua buscas e recupera páginas na Web com conteúdo textual relacionado ao domínio nuclear e seus temas, de forma autônoma e massiva. Este algoritmo foi projetado sob o modelo de um sistema especialista, possuindo, desta forma, uma base de conhecimento que contem tópicos nucleares e palavras-chave que os definem e um mecanismo de inferência constituído por uma rede neural artificial perceptron multicamadas que efetua a estimação da relevância das páginas na Web para um determinado tópico nuclear, no decorrer do processo de busca, utilizando a base de conhecimento. Deste modo, o algoritmo é capaz de, autonomamente, buscar páginas na Web seguindo os hiperlinks que as interconectam e recuperar aquelas que são mais

relevantes para o tópico nuclear selecionado, emulando a habilidade que um especialista nuclear tem de navegar na Web e verificar informações nucleares.

Resultados experimentais preliminares apresentam uma precisão de recuperação de 80% para o tópico “área nuclear em geral” e 72% para o tópico de “energia nuclear”, indicando que o algoritmo proposto é efetivo e eficiente na busca e recuperação de informações relevantes para o domínio nuclear.

NUCLEAR EXPERT WEB CRAWLER ALGORITHM

Thiago Reis

ABSTRACT

Over the last years the Web has obtained an exponential growth, becoming the largest information repository ever created and representing a new and valuable source of potentially useful information for several topics and also for nuclear-related themes. However, due to the Web characteristics and, mainly, because of its huge data volume, finding and retrieving relevant and useful information are non-trivial tasks. This challenge is addressed by web search and retrieval algorithms called **web crawlers**.

This work presents the research and development of a crawler algorithm able to search and retrieve webpages with nuclear-related textual content, in autonomous and massive fashion. This algorithm was designed under the expert systems model, having, this way, a knowledge base that contains a list of nuclear topics and keywords that define them and an inference engine composed of a multi-layer perceptron artificial neural network that performs webpages relevance estimates to some knowledge base nuclear topic while searching the Web. Thus, the algorithm is able to autonomously search the Web by following the hyperlinks that interconnect the webpages and retrieving those that are more relevant to some predefined nuclear topic, emulating the ability a nuclear expert has to browse the Web and evaluate nuclear information.

Preliminary experimental results show a retrieval precision of 80% for the “nuclear general domain” topic and 72% for the “nuclear power” topic,

indicating that the proposed algorithm is effective and efficient to search the Web and to retrieve nuclear-related information.

SUMÁRIO

Página

LISTA DE TABELAS

LISTA DE FIGURAS

LISTA DE ABREVIATURAS, SIGLAS E SÍMBOLOS

CNEN	Comissão Nacional de Energia Nuclear
FIFO	*First In, First Out*
HTML	*HyperText Markup Language*
IPEN	Instituto de Pesquisas Energéticas e Nucleares
RNA	Rede Neural Artificial
URL	*Uniform Resource Locator*

1 INTRODUÇÃO

Nos últimos anos a *World-Wide Web*, ou simplesmente **Web**, obteve um crescimento exponencial, se tornando o maior repositório de informações já criado pelo homem. No ano de 2005 a Web continha uma quantidade estimada de 11,5 bilhões de documentos indexáveis [1] (equivalente a 34,5 vezes 10^{15} bits de informação, onde o tamanho médio por documento é de, aproximadamente, 3 milhões de bits [2]). Para efeito de comparação, em meados de 1980 o maior repositório de informações existente era uma biblioteca, das quais as maiores do mundo continham o equivalente a 10^{14} bits de informação codificada em palavras e 10^{15} bits de informação codificada em imagens [3]. Portanto, a Web representa uma fonte nova e relevante de informações potencialmente uteis para diversas áreas, inclusive a área nuclear. Ciência e tecnologia nuclear, aplicações e indústria nuclear, gestão de conhecimento nuclear, aceitação pública da energia nuclear, entre outros, são exemplos de temas nucleares presentes na Web, tornando dela uma fonte única para a recuperação e análise de informações nucleares em uma perspectiva e abrangência global.

Entretanto, devido às características da Web – repositório distribuído, acessível publicamente, "poluído", multicultural, colaborativo, com dados heterogêneos – e, principalmente, devido ao seu grande volume de dados e ao dinamismo no qual estes dados são criados e atualizados, emerge um problema desafiador relacionado à sua utilização: a busca e recuperação informações relevantes e úteis. Este problema é tratado por algoritmos e sistemas de busca e recuperação de informação que trabalham na Web*,* denominados **rastreadores web** [4].

Um rastreador web – também chamado de: *web crawler*, *web spider*, *web robot*, *web bot*, *web indexer*, *web agent*, *wanderer* ou *worm* – é um programa de computador que busca e recupera informações da Web de forma automática, “visitando” suas páginas e “movendo-se” através dos hiperlinks que as conectam para acessar novas informações. Um rastreador pode buscar e recuperar milhões

de páginas para que sejam armazenadas e indexadas em um repositório central de um sistema de recuperação de informação, como também para que as informações contidas nestas páginas possam ser analisadas para diversos propósitos. As páginas recuperadas da Web pelos rastreadores podem ser úteis a diversas aplicações, como, por exemplo, para suportar um mecanismo de busca geral (como: *Google*[1], *Bing*[2], *Yahoo*[3]), um mecanismo de busca especializado em um determinado assunto, para análises de inteligência de negócios (coletar informações sobre empresas concorrentes, parceiros de negócio, notícias relevantes), para o monitoramento de *web sites* de interesse, etc.

1.1 Objetivos

Sintetizando as questões apresentadas, surge a motivação para o desenvolvimento deste trabalho frente à junção de dois fatores: (1) a importância da Web como repositório de informações nucleares e (2) o estudo e desenvolvimento de algoritmos capazes lidarem com o ambiente de informações da Web, buscando e recuperando as suas informações potencialmente úteis em larga escala.

Assim, este trabalho tem como objetivos projetar, implementar e avaliar um algoritmo de busca e rastreamento preferencial de páginas na Web, capaz de executar buscas heurísticas na Web por páginas com conteúdo textual relacionado à área nuclear e seus temas, de forma autônoma e massiva.

1.2 Trabalhos antecedentes

O trabalho apresentado nesta dissertação é a continuidade do projeto chamado Sistema de Mineração Web Especialista Nuclear [5]. Neste projeto foi proposto um arcabouço computacional que integra diversos algoritmos e técnicas com o propósito de realizar buscas e recuperação de informações nucleares na Web, além de também identificar se os textos recuperados expressam alguma opinião negativa ou positiva sobre a área nuclear. Esta dissertação aprofunda e

[1] http://www.google.com
[2] http://www.bing.com
[3] http://www.yahoo.com

desenvolve o âmbito do rastreador web nuclear inicialmente especificado no projeto.

1.3 Organização da dissertação

Nesta dissertação optou-se por uma abordagem sucinta e objetiva dos assuntos tratados, tornando-a mais concisa e sintética possível, porém sem a perda de informações relevantes e aspectos essenciais relativos ao trabalho em questão.

Assim, primeiramente é realizada no capítulo 2 uma revisão bibliográfica que apresenta o arcabouço teórico que fundamenta os algoritmos de rastreamento web juntamente com a descrição e análise não exaustiva de métodos e algoritmos relacionados com o algoritmo proposto neste trabalho. Posteriormente, no capítulo 3, são apresentadas, de forma integrada, as técnicas e métodos utilizados neste trabalho (sistemas especialistas, redes neurais artificiais, modelo de espaço vetorial) e seus propósitos e aplicações no algoritmo desenvolvido, seguido do detalhamento do algoritmo rastreador nuclear e seu funcionamento. No capítulo 4 é apresentado o procedimento experimental realizado para a avaliação do algoritmo, os resultados mensurados e discussão deles. Por fim, no capítulo 5 são realizadas algumas observações referentes ao cumprimento dos objetivos propostos e limitações do presente trabalho, contribuições, benefícios e trabalhos futuros.

2 REVISÃO BIBLIOGRÁFICA

Nesta revisão bibliográfica são apresentados alguns fundamentos dos algoritmos rastreadores web como também pesquisas destes algoritmos encontradas na literatura e diretamente relacionados a este trabalho. Os algoritmos são apresentados em pseudocódigo, baseado na linguagem Java, para facilitar a descrição dos seus princípios operacionais de alto nível.

2.1 Arcabouço teórico

A informação na Web é primariamente organizada, apresentada e acessada através do uso de **páginas** escritas em código **HTML**[4]. Uma página pode referenciar outra página (chamadas **página-fonte** e **página-alvo**, respectivamente) por meio de **hiperlinks** que as conectam, representam e descrevem a relação entre elas. Um hiperlink é definido dentro do código HTML da página-fonte por um texto tal como:

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

Dentro do código HTML do hiperlink existe uma **URL**[5] que é o endereço na Web da página-alvo referenciada pelo hiperlink. Hiperlinks interconectam as páginas e, portanto, as informações através da Web, formando o conceito subjacente de **hipertexto** que define a estrutura da Web e torna a sua navegação possível.

Este ambiente hipertextual da Web pode ser modelado como um grafo direcionado, conexo e esparso, cujos vértices correspondem às páginas e as arestas correspondem aos hiperlinks que as conectam [6][7][8][9]. Uma aresta conecta a página-fonte X à página-alvo Y se existe um hiperlink na página-fonte X

[4] É a abreviação para a expressão inglesa *HyperText Markup Language* que significa linguagem de marcação de hipertexto e é uma linguagem de marcação utilizada para produzir páginas na Web. Páginas HTML podem ser interpretadas por programas chamados navegadores Web tais como Internet Explorer, Mozila Firefox e Google Chrome.

[5] É a abreviação para a expressão inglessa *Uniform Resource Locator* que significa localizador padronizado de recursos e é o endereço de um recurso disponível em uma rede (um arquivo, uma impressora, uma página, etc.); seja a Internet, a Web, ou uma rede corporativa intranet.

que referencia a página-alvo Y. Este modelo é denominado **grafo da web**. Pesquisas anteriores [10][11][12] demonstraram empiricamente importantes propriedades do grafo da web, como sumarizado abaixo:

- páginas que estão ligadas umas as outras no grafo da web possuem uma relação de recomendação, dado que o autor da página-fonte referenciou a página-alvo usando um hiperlink;
- páginas que estão ligadas umas as outras no grafo da web são propensas a possuir conteúdo relacionado ou similar;
- o texto do hiperlink de uma página-fonte (também conhecido como texto da âncora) geralmente descreve a página-alvo qual ele referencia.

Um algoritmo rastreador explora o grafo da web identificando os hiperlinks existentes em uma determinada página-fonte e os utiliza para buscar as páginas-alvo as quais estes hiperlinks referenciam. O algoritmo rastreador começa a busca recuperando um conjunto de páginas inicialmente informadas a ele (por um usuário ou por outro programa), denominadas de **páginas-semente**, e extraindo os hiperlinks contidos nestas páginas. Em seguida, o rastreador recupera as páginas-alvo as quais os hiperlinks extraídos referenciam e extrai os novos hiperlinks que estão contidos nelas. Este processo é repetido até que um número suficiente de páginas seja recuperado ou alguma outra condição de término seja atingida. Cada repetição deste processo corresponde a uma nova **iteração da busca**.

As URLs das páginas que estão aguardando para serem recuperadas (páginas-semente ou páginas-alvo) são organizadas pelos rastreadores em uma estrutura de dados na forma de uma lista de URLs, denominada **fronteira de busca**. Na terminologia dos algoritmos de busca em grafo, a fronteira de busca é uma lista aberta dos vértices não visitados. Este processo é definido pelo procedimento da Figura 1:

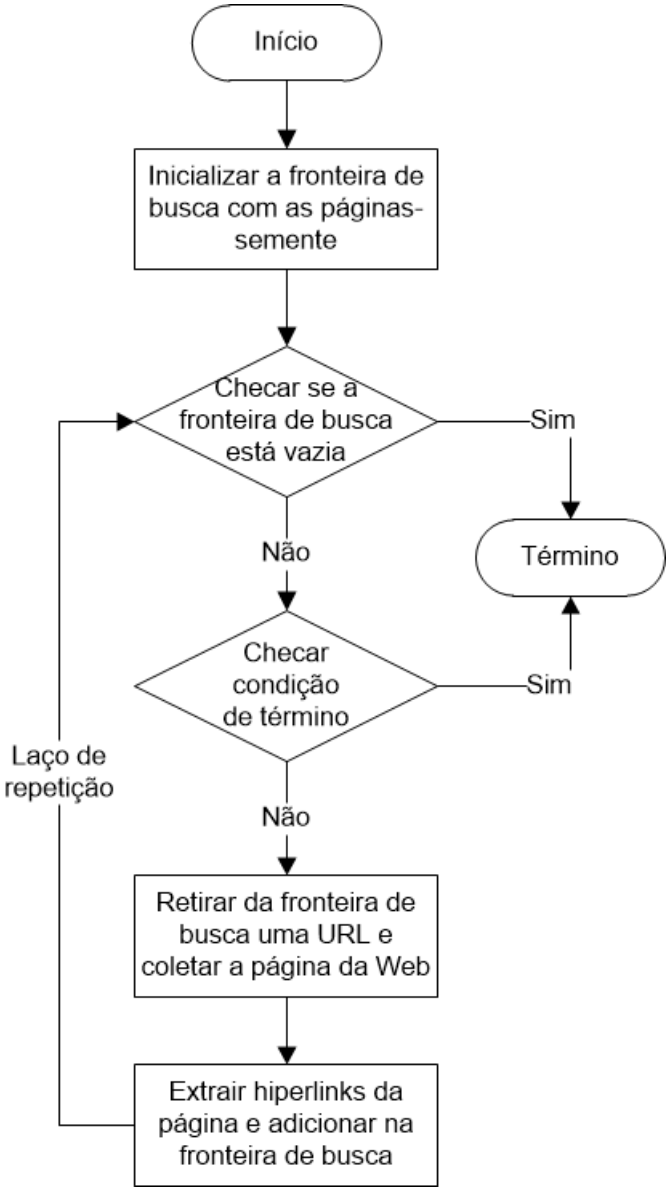

Figura 1 – Procedimento base de rastreamento web

Para que o número de páginas existentes na fronteira de busca não aumente além das capacidades de memória é comum definir uma quantidade máxima de páginas a serem mantidas na fronteira a cada iteração da busca.

Algoritmos rastreadores e de busca na Web são baseados em duas técnicas tradicionais de busca em grafos, conhecidas como **busca em amplitude** (Figura 2) e **busca por melhor escolha** (Figura 3), e utilizam o grafo da web em totalidade como sendo o seu **espaço de busca** [13]. Estes dois algoritmos seguem o procedimento definido na Figura 1, diferenciando-se apenas pela estrutura de dados que é utilizada pela sua fronteira de busca. O algoritmo de busca em amplitude utiliza uma fronteira de busca ordenada como uma **fila** para

adicionar e remover as URLs em ordem FIFO (*first-in, first-out*), enquanto o algoritmo de busca por melhor escolha utiliza uma fronteira de busca ordenada como uma **fila de prioridade** para adicionar e remover as URLs em ordem de maior importância.

```
01  Busca-em-Amplitude (paginas_semente) {
02    criar fronteira como fila;
03    para-cada URL em (paginas_semente) {
04      adicionar_no_fim(fronteira, URL);
05    }
06    enquanto (fronteira <> vazia) {
07      URL = remover_do_inicio(fronteira);
08      pagina = recuperar(URL);
09      hiperlinks = extrair_hiperlinks(pagina);
10      adicionar_no_fim(fronteira, hiperlinks);
11      se (tamanho(fronteira) > tamanho_maximo) {
12        remover_do_fim(fronteira);
13      }
14    }
15  }
```

Figura 2 – Pseudocódigo do algoritmo de busca em amplitude

```
01  Busca-por-Melhor-Escolha (consulta, paginas_semente) {
02    criar fronteira como fila_de_prioridade;
03    para-cada URL em (paginas_semente) {
04      adicionar_prioritariamente(fronteira, URL, 1);
05    }
06    enquanto (fronteira <> vazia) {
07      URL = remover_prioritario(fronteira);
08      pagina = recuperar(URL);
09      hiperlinks = extrair_hiperlinks(pagina);
10      adicionar_prioritariamente(fronteira, hiperlinks);
11      se (tamanho(fronteira) > tamanho_maximo) {
12        remover_menos_prioritarios(fronteira);
13      }
14    }
15  }
```

Figura 3 – Pseudocódigo do algoritmo de busca por melhor escolha

Para efetuar uma **busca exaustiva**[6] ou **busca não informada** (Figura 4a) e recuperar todas as páginas existentes a partir das páginas-semente, um rastreador web deve implementar o algoritmo de busca em amplitude, recuperando as páginas nível por nível, percorrendo caminhos curtos, inicialmente se mantendo próximo das páginas-semente e se afastando gradualmente a cada iteração da busca. Estes rastreadores são geralmente utilizados para buscar e recuperar o máximo possível de páginas, independentemente do seu conteúdo,

[6] Busca exaustiva é a procura exaustiva de soluções do problema no espaço de busca, por meio da verificação de todas as soluções possíveis.

para fins de indexação e arquivamento como também para suportar mecanismos de busca de propósito geral, executando atualizações incrementais dos seus índices e repositório de páginas.

Em contrapartida, existem rastreadores web projetados para buscar e recuperar somente páginas que possuem um conteúdo específico e, assim, eles cuidadosamente priorizam as URLs existentes na fronteira de busca, controlando o processo de exploração do grafo da web. Estes rastreadores web – normalmente denominados **rastreadores preferenciais**, **focados** ou **orientados a tópicos** – concentram suas buscas em um tópico ou consulta de interesse predeterminada, estimando a relevância ao tópico de interesse da página-alvo para a qual um hiperlink referencia antes de realmente recuperá-la. Desta forma, estes rastreadores web efetuam uma **busca heurística**[7] ou **busca informada** (Figura 4b), implementando o algoritmo de busca por melhor escolha e priorizando cada URL existente na fronteira de busca de acordo com a sua relevância estimada. Assim, eles poupam recursos computacionais e tempo de busca, pois recuperam somente uma pequena fração das páginas da Web, sendo geralmente utilizados para construir repositórios de páginas especializados, para descoberta automatizada de recursos, conteúdos e informações na Web e como agentes de *software* facilitadores [4]. As estratégias de busca dos rastreadores heurísticos são baseadas em **heurísticas de busca na web** derivadas das propriedades do grafo da web previamente descritas.

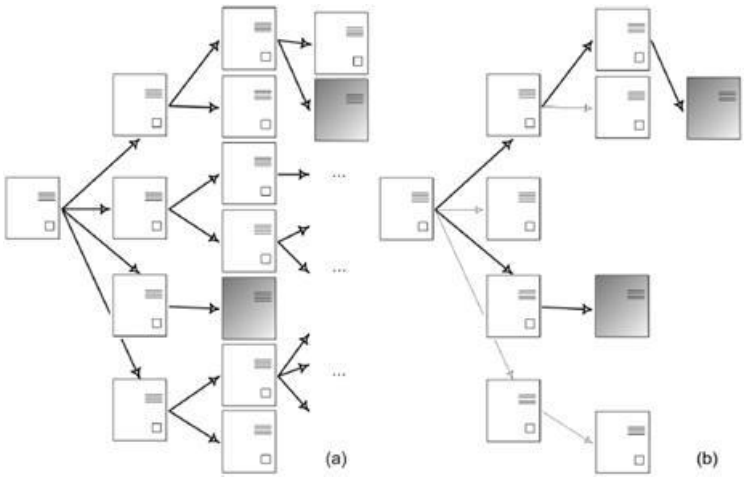

Figura 4 – Busca de rastreador web exaustivo (a) e heurístico (b)

[7] Busca heurística é a procura de soluções aproximadas do problema no espaço de busca, por meio da verificação de algumas soluções possíveis definidas de acordo com regras.

Uma característica importante dos rastreadores heurísticos é o seu comportamento com relação à **investigação** versus a **exploração** do grafo da web [14]. Esta é uma característica universal nas áreas de aprendizado computacional e inteligência artificial. Um algoritmo que tende mais para um comportamento de exploração pode concentrar sua busca em áreas do grafo da web que são mais promissoras, porém pode ficar "preso" em um ótimo local e não atingir um ótimo global, deixando de recuperar páginas importantes que estão em outras áreas do grafo. No outro extremo, algoritmos que tendem mais para um comportamento de investigação podem gastar muito tempo buscando em áreas sub-ótimas do grafo da web, porém sem o risco de ficarem "presos" em ótimos locais e com maiores chances de atingirem ótimos globais. Pant et al. em 2002 [15] analisaram importantes aspectos destas características no contexto dos rastreadores heurísticos.

2.2 Rastreador *WebCrawler*

O algoritmo de busca em amplitude para a tarefa de rastreamento na web (Figura 2) foi primeiramente explorado em 1994 por Pinkerton na pesquisa chamada WebCrawler [16], posteriormente em 1998 por Cho et al. [17] e em 2001 por Najork e Wiener [18].

2.3 Rastreadores *Naïve Best-First* e *Naïve Best-N-First*

O rastreador *Naïve Best-First* foi estudado por Cho et al. em 1998 [17] e por Hersovici et al. em 1998 [19]. Ele é o principal algoritmo de busca heurística, sendo os outros algoritmos heurísticos variações dele. O algoritmo *Naïve Best-First* é similar ao procedimento base de rastreamento web (Figura 1), porém ele estima a relevância de uma página adjacente p utilizando uma **função de avaliação heurística** $f(p)$, cujo resultado, em geral, depende: (1) do hiperlink que leva à página p; (2) do objetivo da busca sendo executada; (3) da informação obtida na busca até este ponto e também (4) do conhecimento prévio de heurísticas de busca na Web e do tópico de interesse a ser encontrado. Uma descrição completa do algoritmo e avaliações encontra-se em Menczer et al. [20].

O algoritmo *Naïve Best-First* recupera as páginas-alvo em ordem de maior relevância estimada, utilizando uma fronteira que implementa uma estrutura de dados como uma fila de prioridade, logo, inserindo e removendo as URLs das páginas-alvo nesta estrutura e ordenando-as de acordo com o resultado da função de avaliação heurística sendo aplicada, conforme o pseudocódigo apresentado na Figura 5.

```
01  Naive-Best-First (consulta, paginas_semente) {
02    criar fronteira como fila_de_prioridade;
03    para-cada URL em (paginas_semente) {
04      adicionar_prioritariamente (fronteira, URL, 1);
05    }
06    enquanto (fronteira <> vazia) {
07      URL = remover_prioritario(fronteira);
08      pagina = recuperar(URL);
09      relevancia = avaliacao(consulta, pagina);
10      hiperlinks = extrair_hiperlinks(pagina);
11      adicionar_prioritariamente(fronteira, hiperlinks, relevancia);
12      se (tamanho(fronteira) > tamanho_maximo) {
13        remover_menos_prioritarios(fronteira);
14      }
15    }
16  }
```

Figura 5 – Pseudocódigo do rastreador *Naïve Best-First* [20]

Desta forma, o algoritmo *Naïve Best-First* seleciona quais páginas-alvo são, provavelmente, mais relevantes e as recupera a cada iteração da busca.

O algoritmo *Naïve Best-N-First* é uma generalização do algoritmo *Naïve Best-First*, onde a cada iteração da busca um lote de N URLs é selecionado da fronteira de busca para a recuperação ao invés de apenas uma única URL. O parâmetro N possibilita um controle fino do comportamento do algoritmo em relação à investigação versus exploração do grafo da web, pois valores menores de N tendem a um comportamento de maior exploração e valores maiores tendem a um comportamento de maior investigação. Após completar a recuperação do lote de N páginas, o rastreador seleciona um novo lote de N páginas da fronteira para a recuperação na próxima iteração da busca, conforme o pseudocódigo apresentado na Figura 6.

```
01  Naive-Best-N-First (consulta, paginas_semente, N) {
02    criar fronteira como fila_de_prioridade;
03    para-cada URL em (paginas_semente) {
04      adicionar_prioritariamente (fronteira, URL, 1);
05    }
06    enquanto (fronteira <> vazia) {
07      URLs = remover_prioritarios(fronteira, N);
08      para-cada URL em (URLs) {
09        pagina = recuperar(URL);
10        relevancia = avaliacao(consulta, pagina);
11        hiperlinks = extrair_hiperlinks(pagina);
12        adicionar_prioritariamente(fronteira, hiperlinks, relevancia);
13        se (tamanho(fronteira) > tamanho_maximo) {
14          remover_menos_prioritarios(fronteira);
15        }
16      }
17    }
18  }
```

Figura 6 – Pseudocódigo do rastreador *Naïve Best-N-First* [20]

Obviamente que para $N = 1$ o *Naïve Best-N-First* terá o mesmo comportamento que o *Naïve Best-First*.

2.4 Rastreador Focado

Chakrabarti et al. em 1999 [21] propôs um rastreador web focado baseado em um classificador Bayesiano de hipertexto que determina a probabilidade de uma determinada página recuperada pertencer a uma categoria de uma taxonomia de tópicos pré-determinados como, por exemplo, os tópicos do *Yahoo*[8] ou ODP[9]. O usuário provê exemplos de URLs de interesse e então o algoritmo as classifica dentre as várias categorias da taxonomia, treinando desta forma o classificador. Iterativamente, o usuário pode corrigir a classificação, adicionar novas categorias à taxonomia e marcar categorias como "boas" em relação ao seu conteúdo de interesse. Um escore de relevância é computado para cada página recuperada conforme a Equação 1.

$$R(página) = \sum_{categoria \in boa} \Pr(categoria|página)$$

Equação 1 – Escore de relevância do rastreador focado

[8] http://www.yahoo.com
[9] http:// www.dmoz.org

O rastreador utiliza o escore de relevância da página recuperada para ordenar as URLs extraídas dela na sua fronteira de busca de modo similar ao rastreador *Naïve Best-First*.

2.5 Rastreador Focado Contextual

O rastreador focado contextual proposto por Diligenti et al. em 2000 [22] utiliza um classificador Bayesiano treinado para estimar a distância no grafo da web entre a página recuperada e outras páginas relevantes. Dado um conjunto de páginas de interesse (páginas-semente), o rastreador contextual constrói uma representação das páginas que ocorrem a certa distância deste conjunto, analisando o grafo da web nas redondezas destas páginas. Esta representação é chamada de grafo de contexto e registra a distância relativa das páginas, definida como sendo o número mínimo de hiperlinks que devem ser "atravessados" para se chegar a uma das páginas do conjunto de interesse. Então, dada uma página, o grafo de contexto é utilizado para treinar o classificador Bayesiano para predizer quantas outras páginas ele deve "atravessar" para chegar a ela. Assim, páginas classificadas como mais próximas de páginas relevantes são recuperadas primeiramente, deste modo mantendo uma fronteira de busca similar ao rastreador *Naïve Best-First*.

2.6 Rastreador *PageRank*

O algoritmo *PageRank* foi proposto por Brin e Page em 1998 [23] como um modelo de comportamento de um visitante aleatório na Web e utilizado por Cho et al. em 1998 [17] para rastreamento web. Neste algoritmo a relevância de cada página é definida por um modelo que estima a probabilidade de que um visitante aleatório visite uma determinada página em um dado momento. Deste modo, a relevância de uma página é definida recursivamente pela relevância das páginas-fonte que levam a ela. Formalmente:

$$PR(p) = (1 - \gamma) + \gamma \sum_{d \in in(p)} \frac{PR(d)}{|out(d)|}$$

Equação 2 – *PageRank* [23]

onde p representa a página sendo avaliada, $in(p)$ representa o conjunto de páginas-fonte que levam a p, $out(d)$ representa o conjunto de hiperlinks que saem de d e $\gamma < 1$ é um fator de redução que representa a probabilidade de o visitante aleatório visitar uma outra página aleatoriamente.

Devido ao algoritmo do *PageRank* requerer um cálculo recursivo até convergir, sua computação torna-se intensiva. Idealmente o *PageRank* deve ser recalculado a cada vez que uma URL seja selecionada da fronteira de busca, porém, para tornar a sua computação menos intensiva, o algoritmo apresentado em pseudocódigo na Figura 7 recalcula o *PageRank* em intervalos regulares:

```
01  PageRank (consulta, paginas_semente, frequencia) {
02    criar fronteira como fila_de_prioridade;
03    para-cada URL em (paginas_semente) {
04      adicionar_prioritariamente (fronteira, URL, 1);
05    }
06    enquanto (fronteira <> vazia) {
07      se (multiplos(visitados, frequencia)) {
08        recomputar_escores_pagerank();
09      }
10      URL = remover_prioritario(fronteira);
11      pagina = recuperar(URL);
12      relevancia = similaridade(consulta, pagina);
13      hiperlinks = extrair_hiperlinks(pagina);
13      adicionar_prioritariamente(fronteira, hiperlinks, relevancia);
15      se (tamanho(fronteira) > tamanho_maximo) {
16        remover_menos_prioritarios(fronteira);
17      }
18    }
```

Figura 7 – Pseudocódigo do rastreador *PageRank* [23]

2.7 Rastreador *Fish-Search*

Um dos primeiros algoritmos adaptativos de rastreamento web desenvolvido foi o *Fish-Search*, proposto por De Bra et al. em 1994 [24]. Basicamente, o algoritmo procura mais extensivamente em áreas da Web onde as páginas relevantes foram encontradas e, de maneira contrária, ele descontinua a busca em regiões onde páginas relevantes não foram mais encontradas. O algoritmo utiliza uma função de avaliação heurística binária.

O algoritmo *Fish-Search* pode ser comparado a um cardume. Quando a comida é encontrada no mar (ou seja, páginas relevantes na Web), os peixes (instâncias do algoritmo) se reproduzem e continuam procurando por mais

comida, porém, na falta de comida (páginas relevantes) ou quando a água está poluída (largura de banda ruim), eles morrem.

O algoritmo obtém como entrada a página-semente e a consulta do usuário e dinamicamente constrói a fronteira de busca como um fila de prioridade contendo as URLs das páginas adjacentes a serem recuperadas, inicializada com a página-semente. A cada iteração da busca, a primeira URL é removida da fronteira e recuperada. O conteúdo da página recuperada é analisado por uma função de avaliação heurística que verifica se a página é relevante ou não relevante para a consulta formulada e, baseada no resultado desta função, uma regra heurística decide se deve continuar a busca nas páginas adjacentes ou não. Sempre que uma página é recuperada são extraídos os hiperlinks contidos nela. As páginas para as quais estes hiperlinks apontam são associadas a um valor de profundidade. Se a página fonte é relevante, o valor de profundidade das páginas adjacentes é configurado para um valor predefinido. Caso contrário, o valor de profundidade das páginas adjacentes é configurado como uma unidade abaixo do valor da página fonte. Quando o valor de profundidade atinge zero a busca é encerrada e nenhuma URL das páginas adjacentes é inserida na fronteira.

```
01  Get as Input parameters, the initial node, the width (width),
      depth (D) and size (S) of the desired graph to be explored,
      the time limit, and a search query;
02  Set the depth of the initial node as depth = D,
      and Insert it into an empty list;
03  While the list is not empty,
      and the number of processed nodes is less than S,
      and the time limit is not reached;
04    Pop the first node from the list and make it the current node;
05    Compute the relevance of the current node;
06    If depth > 0 Then:
07      If current_node is irrelevant Then:
08        For each child, child_node,
            of the first width children of current_node:
09          Set potential_score(child_node) = 0.5;
10        For each child, child_node,
11          of the rest of the children of current_node:
12          Set potential_score(child_node) = 0;
13      Else:
14        For each child, child_node,
            of the first (a * width) children of current_node:
           (where a is a pre-defined constant typically set to 1.5)
15         Set potential_score(child_node) = 1;
16        For each child, child_node,
            of the rest of the children of current_node:
17          Set potential_score(child_node) = 0;
18      For each child, child_node, of current_node:
19        If child_node already exists in the priority list Then:
20          Compute the maximum between the existing score
              in the list to the newly computed potential score;
21          Replace the existing score in the list by that maximum;
22          Move child_node to its correct location
              in the sorted list if necessary;
23        Else Insert child_node at its right location in the
            sorted list according to its potential_score value;
24      For each child, child_node, of current_node:
25        Compute its depth, depth(child_node), as follows:
26          If current_node is relevant Then:
27            Set depth(child_node) = D;
28          Else depth(child_node) = depth(current_node) - 1;
29          If child_node already exists in the priority list Then:
30            Compute the maximum between the existing depth
                in the list to the newly computed depth;
31            Replace the existing depth in the list
                by that maximum;
32  End While;
```

Figura 8 – Pseudocódigo do rastreador *Fish-Search* [24]

2.8 Rastreadores *Shark-Search* e *Shark-N-Search*

O algoritmo *Shark-Search* foi desenvolvido por Hersovici et al. em 1998 [19]. Este algoritmo é uma versão refinada e mais agressiva do algoritmo *Fish-Search*. O algoritmo *Shark-Search* possui duas principais melhorias em relação ao *Fish-Search*: a utilização de uma função de avaliação heurística contínua juntamente com um método refinado de avaliação dos hiperlinks que utiliza o

texto do hiperlink, o texto em torno do hiperlink e a estimativa de relevância herdada das páginas-fonte, conforme o pseudocódigo apresentado na Figura 9.

```
01  Shark-Search (consulta, paginas_semente) {
02    criar fronteira como fila_de_prioridade;
03    para-cada URL em (paginas_semente) {
04      definir_profundidade(URL, d);
05      adicionar_prioritariamente (fronteira, URL, 1);
06    }
07    enquanto (fronteira <> vazia) {
08      URL = remover_prioritario(fronteira);
09      pagina = recuperar(URL);
10      relevancia_1 = similaridade(consulta, pagina);
11      se (profundidade(pagina) > 0) {
12        para-cada hiperlink em (extrair_hiperlinks(pagina)) {
13          relevancia_2 = (1 - r) *
14            relevancia_vizinhos(hiperlink) + r *
15            relevancia_herdada(hiperlink);
16          se (relevancia_1 > 0) {
17            definir_profundidade(hiperlink, d);
18          } senao {
19            definir_profundidade(hiperlink, profundidade(pagina)-1);
20          }
21          adicionar(fronteira, hiperlink, relevância_2);
22        }
23        se (tamanho(fronteira) > tamanho_maximo) {
24          remover_menos_prioritarios(fronteira);
25        }
26      }
27    }
28  }
```

Figura 9 – Pseudocódigo do rastreador *Shark-Search* [18]

Neste algoritmo, os parâmetros d e r representam respectivamente a maior profundidade e a importância relativa da estimativa de relevância herdada por uma determinada página da sua página-fonte.

De maneira análoga ao algoritmo *Naive Best-N-First*, o algoritmo *Shark-N-Search* [14] é uma generalização do algoritmo *Shark-Search* onde o parâmetro N representa o tamanho do lote de páginas a serem recuperadas, fornecendo o controle fino sobre o seu comportamento de investigação versus exploração, conforme o pseudocódigo apresentado na Figura 10.

```
01  Shark-N-Search (consulta, paginas_semente, N) {
02    criar fronteira como fila_de_prioridade;
03    para-cada URL em (paginas_semente) {
04      definir_profundidade(URL, d);
05      adicionar_prioritariamente (fronteira, URL, 1);
06    }
07    enquanto (fronteira <> vazia) {
08      URLs_para_recuperar = remover_prioritarios(fronteira, N);
09      para-cada URL em (URLs_para_recuperar) {
10        URL = remover (fronteira);
11        pagina = recuperar(URL);
12        relevancia_1 = similaridade(consulta, pagina);
13        se (profundidade(pagina) > 0) {
14          para-cada hiperlink em (extrair_hiperlinks(pagina)) {
15            relevancia_2 = (1 - R) *
16              relevancia_vizinhos(hiperlink) + R *
17              relevancia_herdada(hiperlink);
18            se (relevancia_1 > 0) {
19              definir_profundidade(hiperlink, d);
20            } senao {
21              definir_profundidade(hiperlink, profundidade(pagina)-1);
22            }
23            adicionar(fronteira, hiperlink, relevância_2);
24          }
25          se (tamanho(fronteira) > tamanho_maximo) {
26            remover_menos_prioritarios(fronteira);
27          }
28        }
29      }
30    }
31  }
```

Figura 10 – Pseudocódigo do rastreador *Shark-N-Search* [13]

2.9 Rastreador *InfoSpider*

O algoritmo denominado *InfoSpider*, desenvolvido por Menczer em 1998 [25], utiliza uma abordagem baseada na evolução e aprendizado de uma população adaptativa de rastreadores. Cada rastreador da população adapta-se ao ambiente hipertextual local, aprendendo a estimar a relevância das páginas a serem recuperadas em tempo real, enquanto a população como um todo visa abranger todas as áreas promissoras do grafo da Web por meio de reprodução seletiva dos rastreadores individuais.

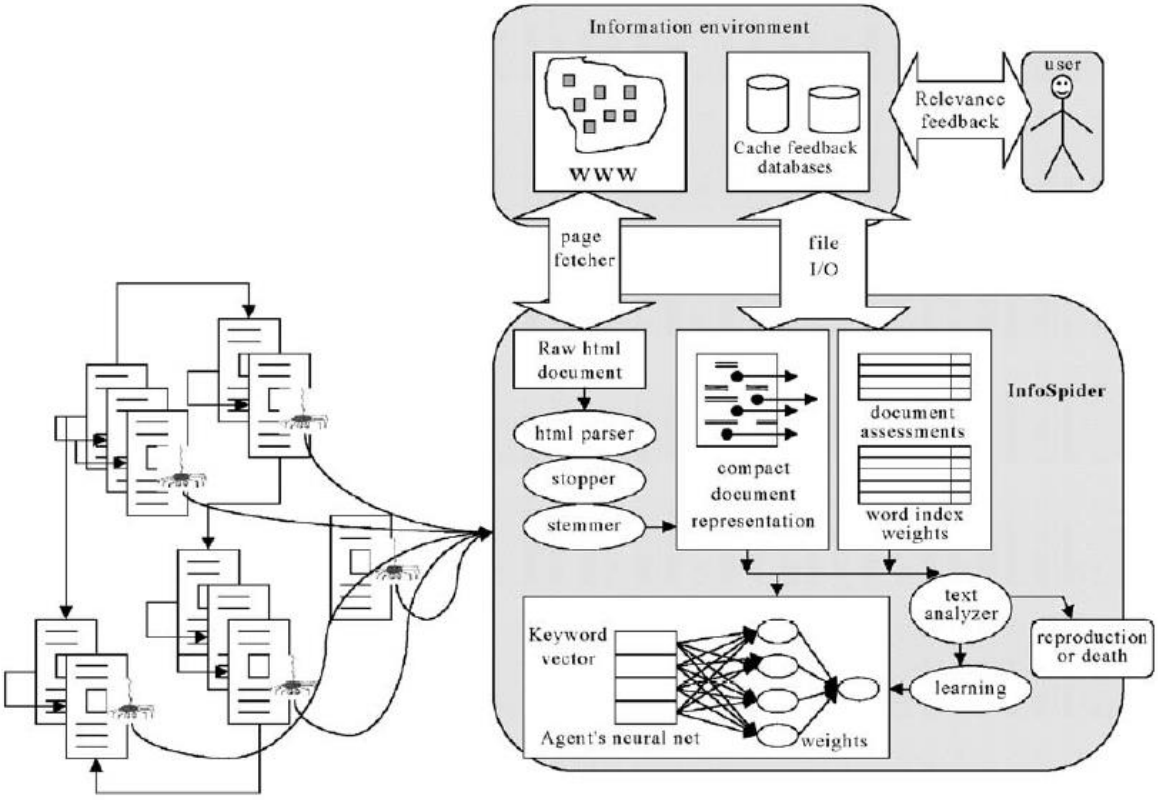

Figura 11 – Diagrama da representação adaptativa do *InfoSpider* [25]

A representação adaptativa do *InfoSpider* (Figura 11) consiste em um vetor de termos e uma rede neural artificial formada por um perceptron de uma única camada que possui como entrada o vetor de termos. Os termos contidos neste vetor representam a “opinião” do rastreador sobre quais palavras melhor discriminam as páginas relevantes em relação à requisição do usuário. A saída da rede neural é um coeficiente que representa a relevância estimada da página a ser recuperada. Especificamente, para cada hiperlink contido em uma determinada página recuperada, cada entrada da rede neural é computada através da contagem das palavras da página recuperada que correspondem aos termos existentes no vetor de termos, onde cada termo do vetor de termos, por sua vez, corresponde a um determinado nó de entrada da rede neural. Esta contagem é ponderada por pesos que decaem com o aumento da distância do termo em relação ao hiperlink em questão, dentro de uma janela de tamanho p. Para cada hiperlink l e cada termo k, a rede neural recebe como entrada:

$$in(k,l) = \sum_{i:dist(k_i,l)\leq p} \frac{1}{dist(k_i,l)}$$

Equação 3 – Frequência ponderada de termos do *InfoSpider* [25]

onde k_i é a i-ésima ocorrência do termo k na página P e $dist(k_i, l)$ é a contagem dos hiperlinks existentes entre k_i e l (incluindo l e até no máximo de p hiperlinks de distância). Este processo é ilustrado na Figura 12:

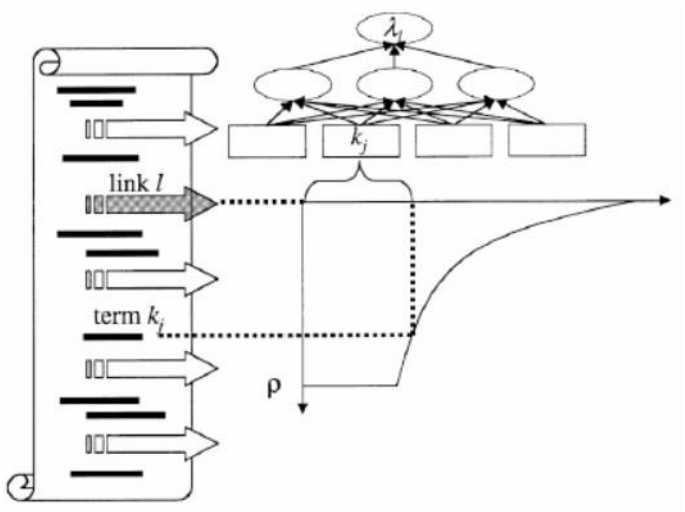

Figura 12 – Frequência ponderada dos termos do *InfoSpider* [25]

O pseudocódigo do *InfoSpider* é apresentado na Figura 13.

```
01  InfoSpider (consulta, paginas_semente) {
02    para-cada agente (1 .. #paginas_semente) {
03      inicializar(agente, consulta);
04      posicionar(agente, paginas_semente);
05      agente.energia = THETA / 2;
06    }
07    custo = #paginas_semente * THETA / (2 * MAX_PAGINAS);
08    enquanto (populacao > 0 e visitados < MAX_PAGINAS) {
09      agente = pegar_um_agente();
10      agente.url = pegar_url_corrente(agente);
11      agente.pagina = recuperar(agente);
12      agente.energia += similaridade(consulta, pagina) - custo;
13      treinar_rede_neural(agente, similaridade(consulta, pagina));
14      se (agente.energia >= THETA e populacao < MAX_BUFFER) {
15        descendente = mutar(clonar(agente));
16        descendente.energia = agente.energia / 2;
17        agente.energia -= descendente.energia;
18      } senao se (agente.energia <= 0) {
19        matar(agente);
20      }
21    }
22  }
```

Figura 13 – Pseudocódigo do rastreador *InfoSpider* [25]

3 METODOLOGIA

Neste trabalho foi projetado um rastreador web preferencial que efetua buscas heurísticas na Web por páginas relacionadas à área nuclear. O rastreador nuclear foi projetado sob o modelo de um **sistema especialista** [26] (também conhecido como **sistema baseado em conhecimento**). Tal sistema rastreador integra alguns componentes que lhe proporcionam o conhecimento nuclear especializado necessário para efetuar buscas na Web por informações nucleares, desta forma emulando a habilidade que um especialista nuclear tem de navegar na Web e verificar informações.

Em síntese, um sistema especialista é um sistema de computador que emula a habilidade de tomada de decisão de um especialista humano em um domínio específico de conhecimento por meio da aquisição, representação e raciocínio sobre este conhecimento, com o propósito de resolução de problemas complexos, suporte a decisões ou provimento de recomendações neste domínio de conhecimento [26].

3.1 Arcabouço metodológico

Sendo o rastreador nuclear um sistema especialista, ele compreende dois componentes principais: (1) uma **base de conhecimento**, construída como um **vocabulário nuclear** por um processo de **engenharia de conhecimento** e obtida a partir do conhecimento a priori dos especialistas nucleares, que contem **palavras-chave** cuidadosamente selecionadas e fortemente relacionadas a alguns **tópicos nucleares** predefinidos e (2) um **mecanismo de inferência** – uma rede neural artificial – que efetua a estimação de relevância dos hiperlinks e é utilizada pelo rastreador nuclear para guiar as suas buscas, realizando o papel de função de avaliação heurística.

O processo de engenharia de conhecimento (Figura 14) possui como objetivo integrar o conhecimento dos especialistas nucleares sobre o domínio nuclear na base de conhecimento [26] e, assim, prover o rastreador nuclear com o

conhecimento terminológico necessário sobre determinados tópicos do domínio nuclear. Este conhecimento é representado na forma de vocabulário que são relacionados a tópicos e assuntos nucleares pré-determinados e que são constituídos de palavras-chave de busca. As palavras-chave da base de conhecimento podem ser ponderadas para dar mais ou menos peso para aquelas que sejam mais ou menos relacionadas ao um determinado tópico ou assunto nuclear.

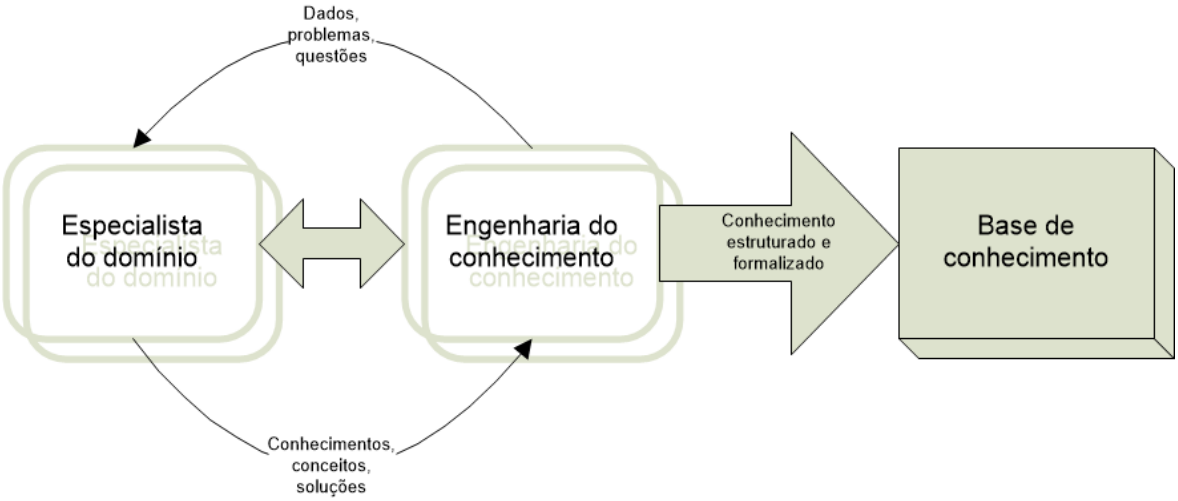

Figura 14 – Processo de engenharia de conhecimento

O mecanismo de inferência é composto por uma rede neural artificial perceptron multicamadas de alimentação adiante. Redes neurais artificiais são modelos matemáticos e computacionais inspirados nas estruturas neurais biológicas que efetuam processamento paralelizado e distribuído [27], com características de aprendizagem, memória, generalização, adaptação, etc. Elas são geralmente representadas em um grafo orientado (Figura 15) como sistemas de neurônios interconectados que computam funções complexas.

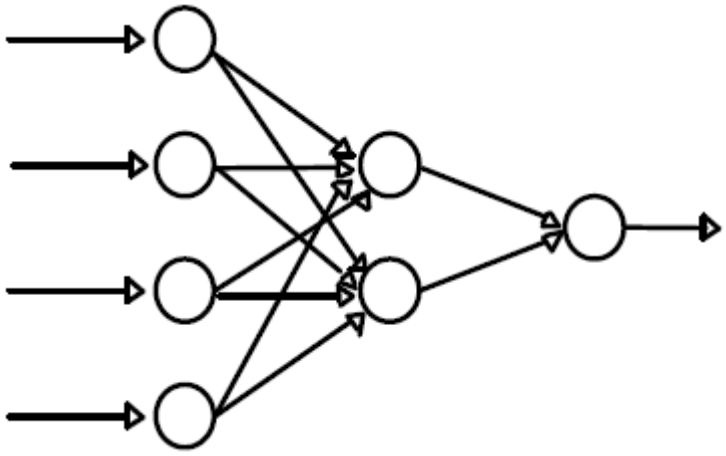

Figura 15 – Grafo orientado de uma rede neural artificial

O perceptron multicamadas é capaz de aprender superfícies de decisão não lineares utilizando funções diferenciáveis [27]. Neste trabalho foi utilizada a função logística como na Equação 4.

$$f(x) = \frac{1}{1 + e^{-x}}$$

Equação 4 – Função logística

O rastreador nuclear utiliza o vocabulário nuclear da base de conhecimento, a rede neural do mecanismo de inferência e implementa um algoritmo de busca por melhor escolha para efetuar buscas no ambiente hipertextual da Web por páginas cujo o conteúdo textual é similar a um determinado tópico nuclear selecionado. O tópico nuclear é especificado por uma configuração predefinida de ponderação das palavras-chave selecionadas a partir do vocabulário nuclear. Deste modo o rastreador nuclear é capaz de encontrar páginas relacionadas à área nuclear dentro do grande grafo da web, percorrendo os hiperlinks que as interconectam. Este projeto é ilustrado na Figura 16.

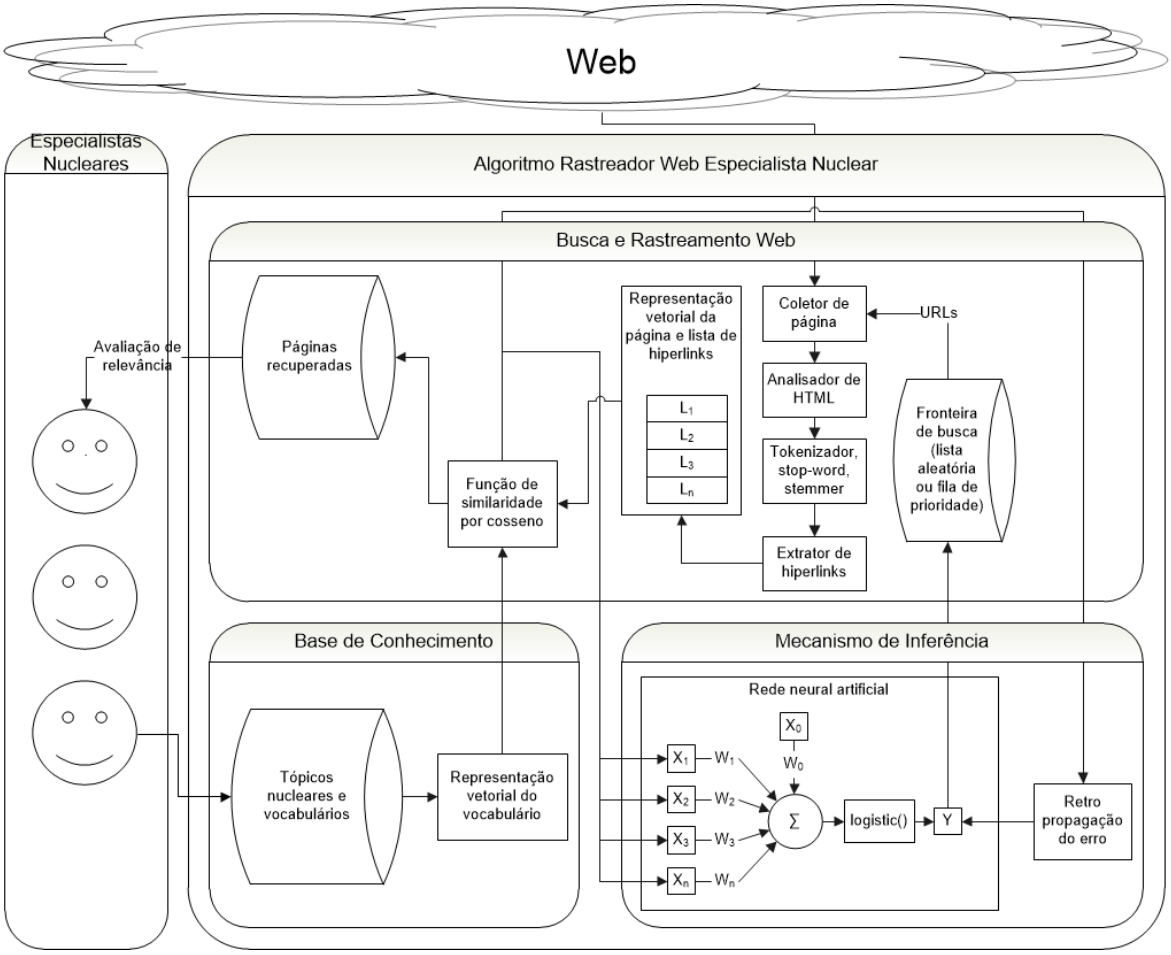

Figura 16 – Diagrama do projeto do rastreador especialista nuclear

Para identificar se o conteúdo textual de uma página (ou parte dela) é relacionado a algum vocabulário dos tópicos nucleares, o rastreador nuclear constrói uma representação vetorial da página e do vocabulário utilizando o **modelo de espaço vetorial** [9][28]. Neste modelo, a página e o vocabulário são representados por vetores em um grande espaço multidimensional onde cada uma das suas palavras corresponde a uma dimensão ou eixo no espaço vetorial (Figura 17). Nesta representação o texto é tratado como um **saco de palavras**[10], pois a ordem das palavras no texto é ignorada, mas a frequência delas é considerada.

[10] Modelo simplificado do texto onde este é representado como uma coleção não ordenada de palavras, considerando somente a sua frequência no texto.

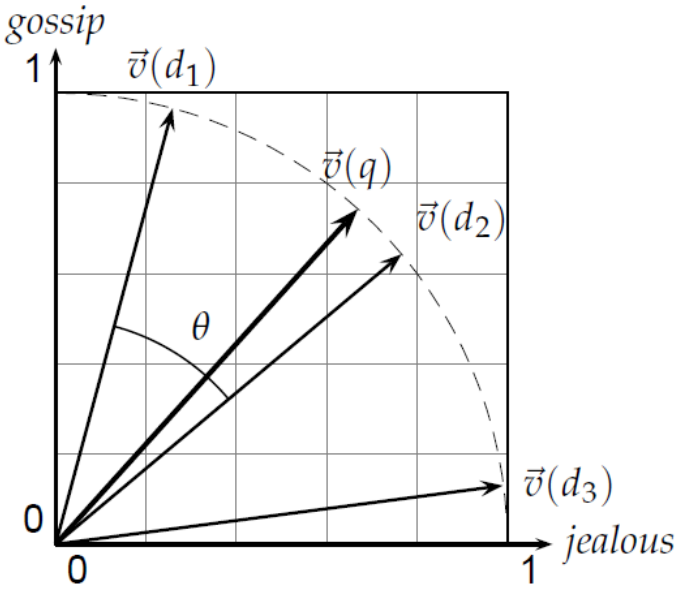

Figura 17 – Ilustração da similaridade por cosseno [9]

Após construir a representação vetorial dos textos da página e do vocabulário, o rastreador nuclear calcula a **função de similaridade por cosseno** (Equação 5) de ambos os vetores

$$sim(v,p) = \frac{\vec{v} \cdot \vec{p}}{|\vec{v}||\vec{p}|}$$

Equação 5 – Função de similaridade por cosseno

onde o numerador representa o produto escalar do vetor do vocabulário $\vec{v}$ e do vetor da página $\vec{p}$, e o denominador é o produto das suas normas euclidianas, assim normalizando o comprimento dos vetores para vetores unitários e tornando possível a comparação de textos similares mas de tamanhos diferentes. Altos valores de similaridade indicam que o conteúdo textual da página é mais similar ou mais relacionado ao vocabulário e respectivo tópico nuclear.

Enquanto o processo de busca é executado, a rede neural recebe como entrada os valores de similaridade entre o vocabulário nuclear e os segmentos de textos de um determinado hiperlink e sua respectiva página-fonte, calculados pela função de similaridade por cosseno, e retorna como saída o valor de relevância estimada para página-alvo – chamado de **pré-escore**. Duas principais heurísticas de busca na web, baseadas nas propriedades do grafo da web previamente apresentadas, são utilizadas para extrair informação a respeito

da página-alvo e para construir as entradas da rede neural: (1) o texto no entorno do hiperlink na página-fonte e (2) o texto dentro do hiperlink. O primeiro é baseado na propriedade que páginas conectadas são propensas a possuir conteúdo similar ou relacionado e o segundo é baseado na propriedade que o texto do hiperlink geralmente descreve a página-alvo a qual ele referencia.

Após a página-alvo ser recuperada da Web, o rastreador nuclear calcula a similaridade entre a sua representação vetorial e a do vocabulário nuclear. Este valor de similaridade é chamado de **pós-escore** e é uma estimativa de relevância mais precisa, fornecida como sendo um **sinal de reforço do ambiente** e utilizada para efetuar o treinamento por reforço da rede neural por meio do algoritmo de **retro propagação de erro**.

Estes métodos são combinados e utilizados em conjunto no algoritmo do rastreador nuclear.

3.2 Algoritmo desenvolvido

O algoritmo do rastreador nuclear é executado em dois modos ou etapas distintas: (1) **etapa de treinamento** e (2) **etapa de recuperação**. Na etapa de treinamento é efetuada uma busca parcialmente aleatória na Web para a construção de uma amostra de páginas diversificada que é utilizada para o treinamento da rede neural. A etapa de recuperação é utilizada para efetivamente recuperar páginas com conteúdo textual relacionado ao tópico nuclear, utilizando a rede neural previamente treinada.

Em ambas as etapas o rastreador nuclear recebe como parâmetros: (1) o vocabulário nuclear contendo as palavras-chave relacionadas ao tópico nuclear sendo buscado, (2) um conjunto de páginas-semente, (3) o valor da quantidade máxima de páginas para recuperar, (4) uma variável indicando se o algoritmo deve executar a etapa de treinamento ou etapa de recuperação, e (5) o número de épocas para o treinamento da rede neural. O pseudocódigo do rastreador especialista nuclear é apresentado na Figura 18.

```
01  Rastreador-Especialista-Nuclear (vocabulario,
02    paginas_semente, busca-maxima, treinar, epocas) {
03    se (treinar = verdadeiro) {
04      criar fronteira como lista_aleatoria;
05    } senao {
06      criar fronteira como fila_de_prioridade;
07    }
08    criar recuperadas como lista;
09    criar rede-neural como perceptron_multicamadas;
10    criar buscadas como inteiro;
11    para-cada (URL em paginas_semente) {
12      url.pre-escore = nulo;
13      fronteira.adicionar(URL, nulo);
14      enquanto (nao fronteira.vazia e buscadas <= busca-maxima) {
15        buscadas = buscadas + 1;
16        criar pagina como pagina;
17        hiperlink = fronteira.remover();
18        pagina.html = recuperar(hiperlink.url);
19        pagina.origem = hiperlink;
20        pagina.texto = extrair_texto(pagina.html);
21        pagina.texto = tokenizar_texto(pagina.texto);
22        pagina.texto = remover_stopword(pagina.texto);
23        pagina.texto = stemming_texto(pagina.texto);
24        pagina.pre-escore = hiperlink.url.pre-escore;
25        pagina.pos-escore = similaridade(pagina.texto, vocabulario);
26        recuperadas.adicionar(pagina);
27        hiperlinks = extrair_hiperlinks(pagina);
28        para-cada (hiperlink em hiperlinks) {
29          se (nao fronteira.contem(hiperlink.url)
30            e nao recuperadas.contem(hiperlink.url)) {
31            pre-escore = rede-neural.classificar(hiperlink);
32            hiperlink.pre-escore = pre-escore;
33            fronteira.adicionar(hiperlink, pre-escore);
34          }
35        }
36      }
37      fronteira.limpar;
38      buscadas = 0;
39    }
40    se (treinar = verdadeiro) {
41      para-cada (pagina em recuperadas) {
42        rede-neural.aprender(pagina);
43      }
44    }
45  }
```

Figura 18 – Pseudocódigo do rastreador nuclear especialista

O rastreador nuclear executa, em ambas as etapas, um **ciclo de busca** para cada URL das páginas-semente (Figura 18, linha 11). Em cada ciclo de busca, primeiro, o rastreador nuclear inicializa sua fronteira de busca pegando e removendo uma URL das páginas-semente e a adicionado à sua fronteira de busca (Figura 18, linha 13). Então, ele inicia uma iteração de busca (Figura 18, linha 14) pegando e removendo uma URL da fronteira de busca (Figura 18, linha 17), recuperando a página nesta URL (Figura 18, linha 18), processando e

analisando o texto da página (Figura 18, linhas 20 até 25) e extraindo e adicionando na fronteira de busca os hiperlinks existentes na página (Figura 18, linhas 27 até 35). As iterações de busca são repetidas até que o rastreador nuclear atinja o parâmetro de número máximo de páginas para recuperar e então inicie um novo ciclo de busca para cada URL restante nas páginas-semente.

Na etapa de treinamento, a fronteira de busca implementa uma lista ordenada aleatoriamente (Figura 18, linha 4), onde cada URL da fronteira de busca possui uma probabilidade uniforme de ser pega a cada iteração de busca, conforme uma amostragem aleatória simples. Isto torna a busca na etapa de treinamento menos seletiva e proporciona uma amostra de treinamento mais diversificada para o treinamento da rede neural. Na etapa de recuperação é utilizada uma fronteira de busca que implementa uma fila de prioridade (Figura 18, linha 6) para tornar o rastreador nuclear capaz de recuperar páginas em ordem descendente dos seus pré-escores e, assim, efetuando uma busca seletiva por páginas relacionadas ao tópico nuclear selecionado. No algoritmo rastreador nuclear não foi implementado o parâmetro N, logo a sua busca na etapa de recuperação tende a um comportamento de completa exploração do grafo da web.

Para cada hiperlink extraído da página-fonte na iteração de busca (Figura 18, linha 28), o rastreador nuclear calcula o pré-escore da página-alvo (Figura 18, linhas 31 e 32) analisando o texto do hiperlink da página-fonte utilizando suas representações vetoriais, a função de similaridade por cosseno e alimentando as entradas da rede neural com os valores de similaridade. Então, cada hiperlink é adicionado à fronteira de busca e ordenado de acordo com o pré-escore da página-alvo a qual ele referencia (Figura 18, linha 33).

As entradas da rede neural são computadas como segue (Figura 18, linha 31): a primeira entrada é a similaridade por cosseno entre os vetores do vocabulário nuclear e do texto do hiperlink; a segunda entrada até a sexta entrada são as similaridades por cosseno entre os vetores do vocabulário nuclear e dos blocos de texto da página-fonte existentes entre o hiperlink corrente e os outros hiperlinks intervenientes da página-fonte, até uma janela máxima de cinco hiperlinks de distância; a ultima entrada é a similaridade por cosseno entre os vetores do vocabulário nuclear e o texto completo da página-fonte.

O processamento e análise do texto da página são efetuados de acordo com os seguintes passos: (1) o código HTML da página é interpretado por um analisador sintático para a extração do conteúdo textual (Figura 18, linha 20); (2) a sequencia de texto extraída é **tokenizada**[11] para separá-la em palavras, símbolos e números individuais (Figura 18, linha 21); (3) ***stop-words***[12] são removidas do texto tokenizado (Figura 18, linha 22); (4) as palavras restantes são processadas pelo algoritmo Porter ***stemmer***[13] [29] (Figura 18, linha 23); (5) o pré-escore da página é atribuído copiando o pré-escore computado para o hiperlink que a referenciava (Figura 18, linha 24); (6) o pós-escore é calculado pela similaridade por cosseno entre os vetores do vocabulário nuclear e o vetor do texto processado da página (Figura 18, linha 25).

Quando está sendo executada a etapa de treinamento o rastreador nuclear, por fim, efetua o treinamento da rede neural executando o algoritmo de retro propagação do erro para cada página recuperada, computando o erro entre o pré-escore estimado para a página e o pós-escore observado. Então, a rede neural treinada é armazenada para ser utilizada pelo rastreador nuclear em etapas de recuperação futuras.

[11] Tokenização é o processo de decompor um texto em cada palavra, símbolo, número, etc. que o compõe, identificando os delimitadores existentes no texto (espaços em branco, quebras de linha, tabulações, entre outros).
[12] *Stop-words* são palavras extremamente comuns que normalmente não adicionam significado ao texto.
[13] *Stemming* é o processo de reduzir a palavra a sua forma raiz, desta forma tratando palavras com sufixos diferentes como iguais.

4 RESULTADOS E DISCUSSÃO

A fim de avaliar o algoritmo, foi elaborado um experimento para a recuperação de páginas na Web com o conteúdo textual relacionado a dois tópicos nucleares: (1) área nuclear em geral e (2) energia nuclear especificamente. Uma busca foi realizada ao longo de milhares de páginas na Web e métricas de avaliação foram computadas.

4.1 Procedimento experimental

Um vocabulário nuclear ponderado composto de dez palavras-chave foi utilizado como a base de conhecimento do rastreador nuclear na busca de ambos os tópicos nuclear especificados, como apresentado na Tabela 1. O vocabulário e as páginas-semente foram definidos em língua inglesa devido ao fato que o algoritmo de *stemming* utilizado [29] ser aplicável somente a esta língua.

Tabela 1 – Vocabulário nuclear

Palavra-chave	Ponderação
nuclear	10
energy	5
power	5
reactor	2
uranium	2
atomic	1
electric	1
technology	1
physics	1
fuel	1

Seis páginas-sementes foram selecionadas de *web sites* relacionados aos dois tópicos nucleares definidos: *Wikipedia*[14], *World Nuclear Association*[15] e *International Atomic Energy Agency*[16], como apresentado na Tabela 2.

[14] http://en.wikipedia.org
[15] http://world-nuclear.org
[16] http://www.iaea.org

Tabela 2 – Páginas-semente

Página-semente	Tópico nuclear
http://en.wikipedia.org/wiki/Nuclear_power	Energia nuclear
http://en.wikipedia.org/wiki/Nuclear_technology	Área nuclear em geral
http://world-nuclear.org/info/Current-and-Future-Generation	Energia nuclear
http://world-nuclear.org/info/Non-Power-Nuclear-Applications	Área nuclear em geral
http://www.iaea.org/NuclearPower	Energia nuclear
http://www-naweb.iaea.org/na	Área nuclear em geral

Os parâmetros de busca e da rede neural foram definidos como apresentado na Tabela 3.

Tabela 3 – Parâmetros de busca e da rede neural

Parâmetro	Valor
Unidades de entrada da rede neural	7
Camadas escondidas da rede neural	1
Neurônios da camada escondida da rede neural	4
Neurônios de saída da rede neural	1
Função de transferência da rede neural	logística
Taxa de aprendizado da rede neural	0,1
Taxa de momentum da rede neural	0,5
Épocas de treinamento da rede neural	1000
Ciclos de busca (igual ao número de URLs nas páginas-semente)	6
Iterações de busca (número de páginas para recuperar)	1000
Número máximo de URLs na fronteira de busca	ilimitado

Dadas estas especificações, primeiro foi executada uma etapa de treinamento do algoritmo e efetuado o treinamento da rede neural e, após, foi executada uma etapa de recuperação, onde foram computadas as métricas de avaliação. Em resumo, em ambas as etapas foram efetuados seis ciclos de busca, um para cada URL das páginas-semente, com 1000 iterações de busca para cada ciclo, recuperando assim um total de 12000 páginas.

4.2 Avaliações de desempenho

A soma do erro quadrático na etapa de treinamento da rede neural no decorrer das épocas de treinamento é apresentada na Figura 19. A Equação 6 apresenta a função de soma do erro quadrático da rede neural, onde N é o número total de páginas na amostra de treinamento, C é o número total de

neurônios na camada de saída, t_{ij} é a saída esperada do i-ésimo neurônio da camada de saída para a j-ésima página, y_{ij} é a saída computada pelo i-ésimo neurônio da camada de saída para a j-ésima página.

$$erro = \frac{1}{2}\sum_{i=1}^{N}\sum_{j=1}^{C}\left(t_{ij} - y_{ij}\right)^2$$

Equação 6 – Função de soma do erro quadrático

No treinamento da rede neural era esperado que o valor da soma do erro quadrático diminuísse gradualmente no decorrer das épocas sem apresentar grandes oscilações, sendo o único critério para a finalização do treinamento a execução de 1000 épocas e, assim, não sendo estabelecido um valor mínimo para a soma do erro quadrático. Deste modo, o resultado atingido no treinamento da rede neural é satisfatório, pois a soma do erro quadrático inicia em 46,6 na primeira época de treinamento e atinge 34,2 na milésima época, sendo suficiente para ajustar os pesos da rede neural e melhorar a sua capacidade de classificação das páginas, conforme será apresentado a seguir.

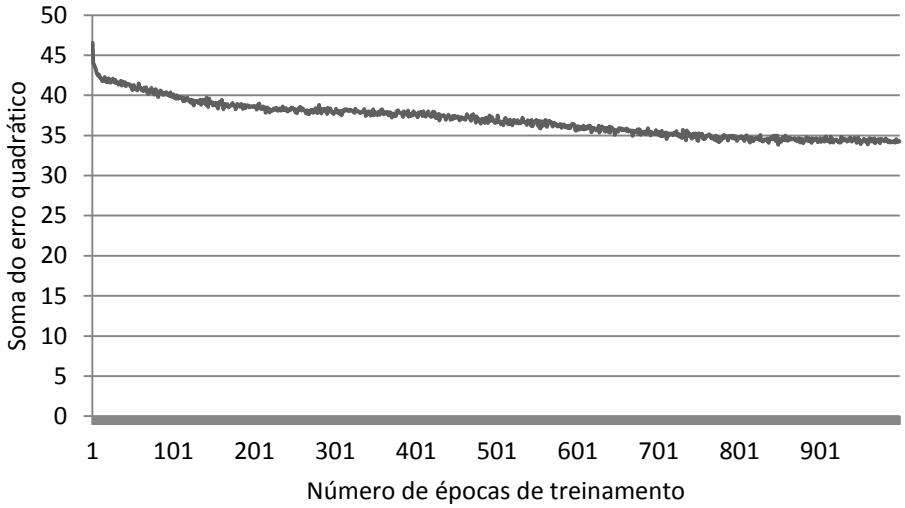

Figura 19 – Erro de treinamento da rede neural

A Figura 20 apresenta o gráfico de comparação do pré-escore, computado pela rede neural, e o pós-escore, computado pela função de similaridade por cosseno, nas etapas de treinamento e recuperação. A rede neural inicialmente destreinada durante a execução etapa de treinamento calcula

altos valores de pré-escore para as páginas-alvo, isto porque é utilizada uma fronteira de busca ordenada aleatoriamente, e os pós-escores das páginas são baixos, causando assim um alto erro de classificação como observado na Figura 21. Na etapa de recuperação, a rede neural já treinada estima com maior precisão os pós-escore das páginas-alvo, diminuindo o erro de classificação e tornando o rastreador nuclear mais eficiente na busca.

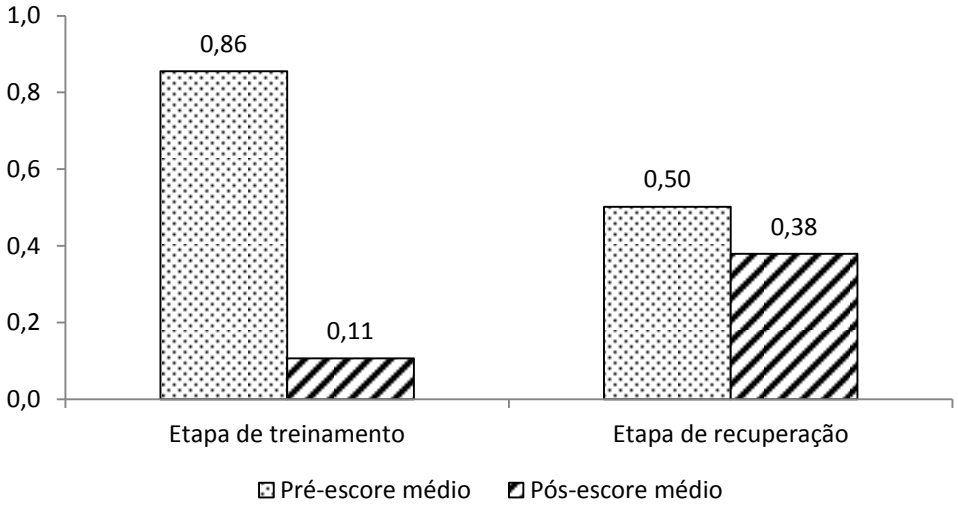

Figura 20 – Pré vs. pós-escore nas etapas de treinamento e recuperação

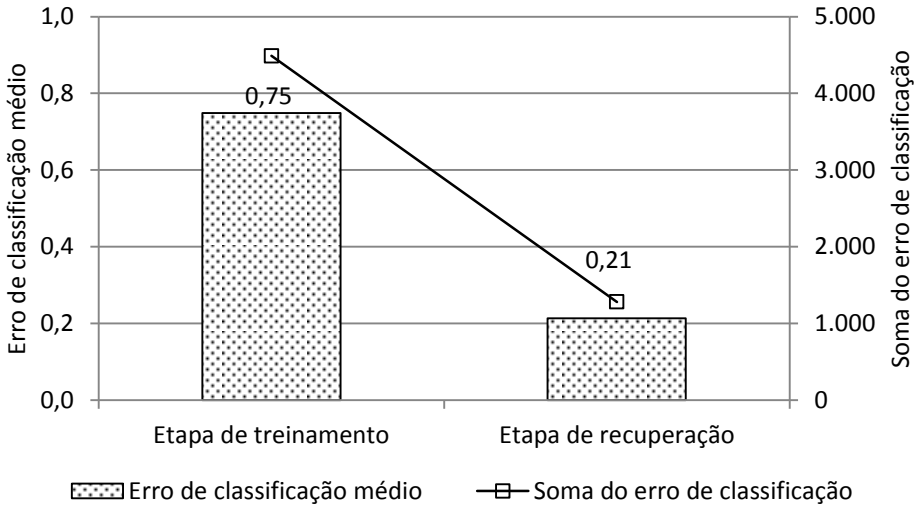

Figura 21 – Erro de classificação nas etapas de treinamento e recuperação

A Figura 22 e Figura 23 apresentam o detalhamento da Figura 20 e Figura 21, respectivamente, para cada ciclo de busca efetuado a partir de cada página-semente na etapa de treinamento. Pode-se observar que para todas as

páginas-semente houve uma tendência de a rede neural superestimar o escore da página-alvo.

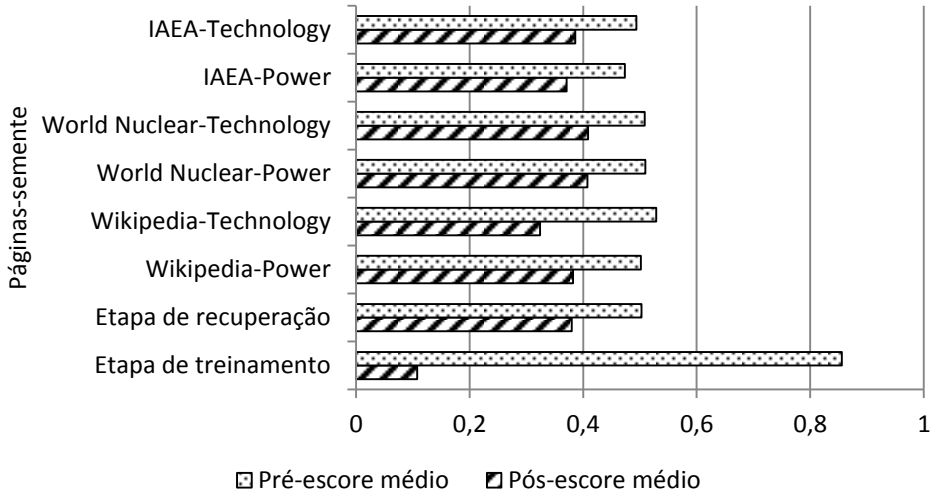

Figura 22 – Pré vs. pós-escore na etapa de recuperação por página-semente

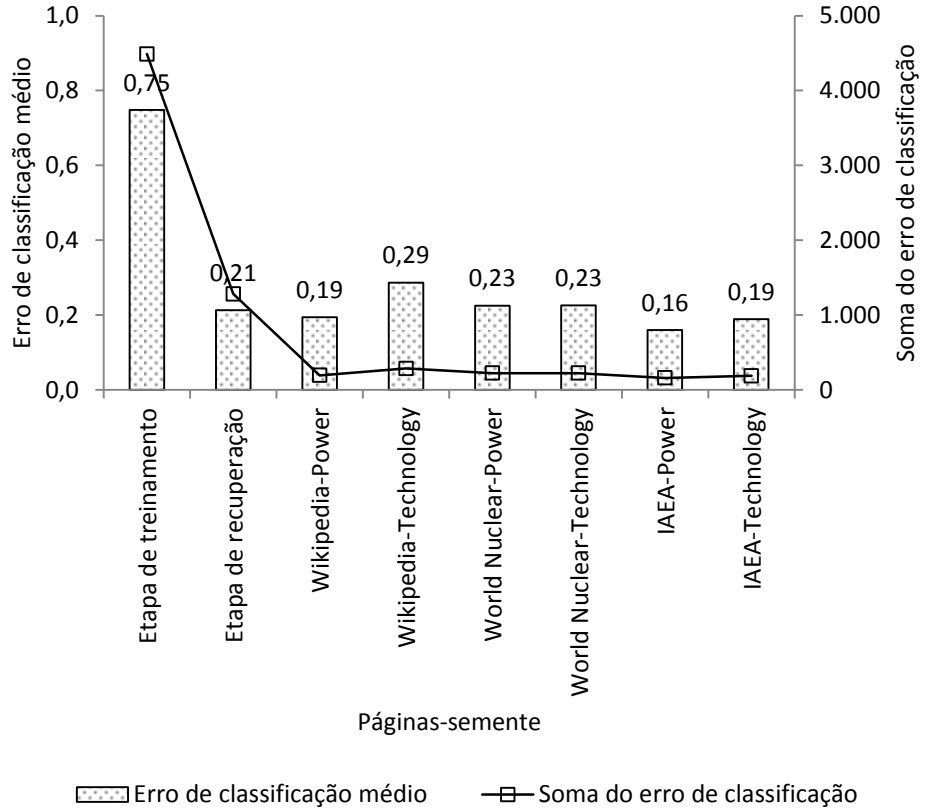

Figura 23 – Erro de classificação na etapa de recuperação por página-semente

Para efetuar uma comparação de desempenho, foi executada uma busca utilizando o algoritmo *Naïve Best-First* apresentado na Figura 5. Como

parâmetros do algoritmo, foi utilizado o mesmo vocabulário nuclear (Tabela 1) e o mesmo conjunto de páginas-semente (Tabela 2). Foi utilizada como função de avaliação heurística a função de similaridade por cosseno (Equação 5). A Figura 24 e Figura 25 apresentam os resultados da busca para este algoritmo.

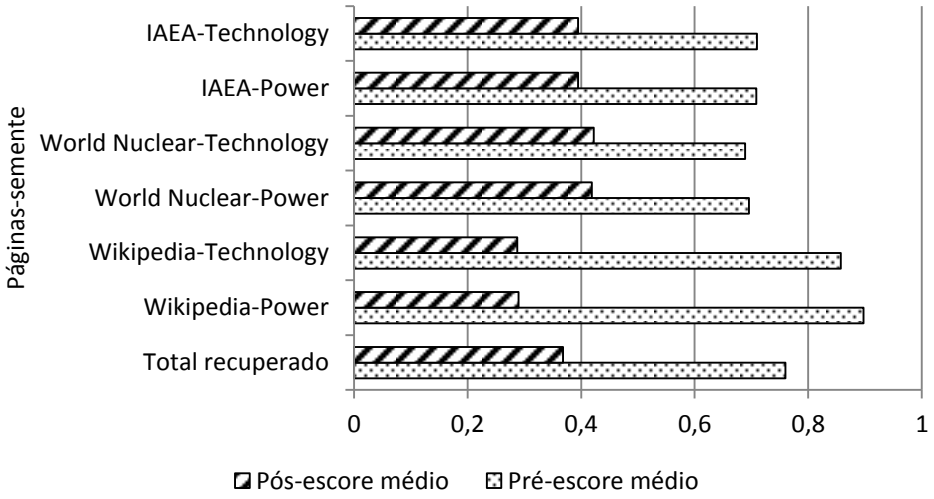

Figura 24 – Pré vs. pós-escore para o rastreador *Naïve Best-First*

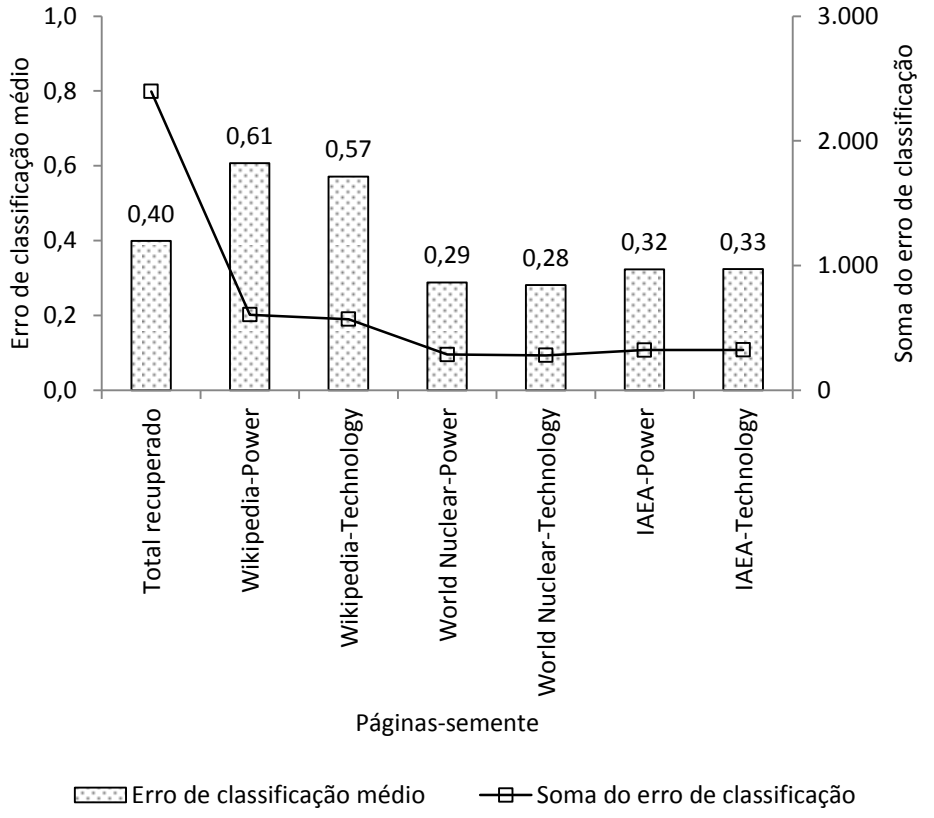

Figura 25 – Erro de classificação para o rastreador *Naïve Best-First*

A Figura 26 apresenta a taxa percentual de quanto o pré-escore médio das páginas recuperadas pelo rastreador nuclear é maior em relação ao *Naïve Best-First*. Pode-se observar que, para o conjunto total, o rastreador nuclear apresentou um desempenho ligeiramente melhor que o rastreador *Naïve Best-First*, onde o pós-escore médio das páginas recuperadas por ele foi 3,2% maior. Para as páginas-semente da *Wikipedia*, o desempenho do rastreador nuclear foi consideravelmente melhor, obtendo um pós-escore médio 12,7% e 31,8% maior. Entretanto, para as páginas-semente da *World Nuclear Association* e *International Atomic Energy Agency*, o seu desempenho foi um pouco pior, obtendo um pós-escore médio variando de 2,1% à 6,1% menor. Uma possível explicação para isto é devido a *Wikipedia* ser uma enciclopédia *online* e, por tanto, suas páginas possuem um conteúdo textual mais verboso e estão mais interconectadas, sendo mais representativas que as outras páginas-semente e assim propiciando ao rastreador nuclear mais evidências a respeito da possível relevância de uma determinada página-alvo e suportando melhor as suas heurísticas de busca. Uma segunda possibilidade seria devido a um eventual sobre ajuste da rede neural às páginas da *Wikipedia*.

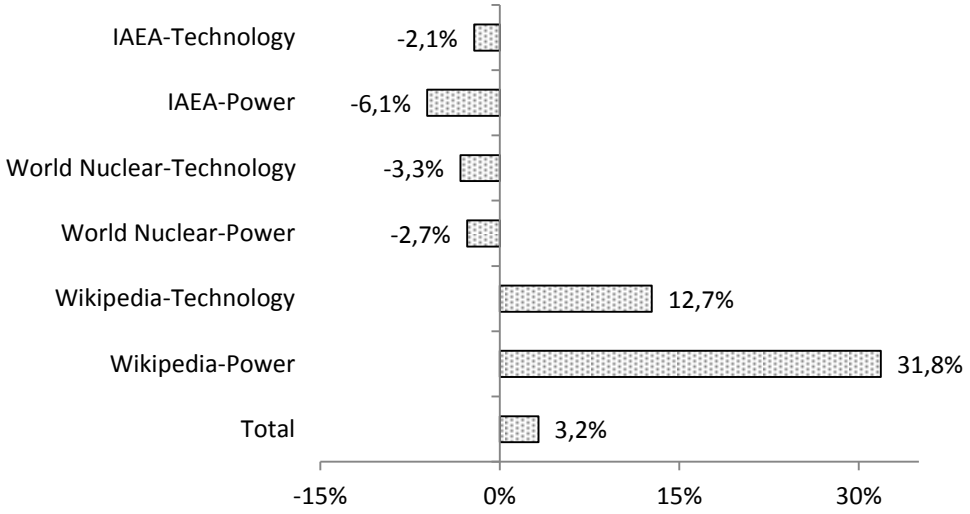

Figura 26 – Relação do pós-escore médio entre ambos os algoritmos

Um pequeno subconjunto de 60 páginas foi aleatoriamente selecionado, do total de 6000 páginas recuperadas na etapa de recuperação, e utilizado como uma amostra para efetuar a avaliação de relevância pelos especialistas nucleares. Dois especialistas nucleares manualmente e

independentemente classificaram cada página da amostra em relação aos tópicos nucleares definidos, marcando “sim” se a página possuir conteúdo textual relacionado ao tópico nuclear ou “não” caso contrário. Uma tabela de contingência como mostrado na Tabela 4 foi utilizada para organizar os resultados das classificações dos especialistas nucleares.

Tabela 4 – Tabela de contingência de avaliação de relevância

		Especialista nuclear 2		
		Sim	Não	Total
Especialista nuclear 1	Sim	a	b	q_1
	Não	c	d	q_0
	Total	p_1	p_0	n

Duas métricas de avaliação foram computadas a partir da tabela de contingência: (1) precisão (Equação 7), que é a fração de páginas recuperadas que são relevantes para os especialistas nucleares [9] e (2) estatística kappa (Equação 10), que é uma medida da magnitude da concordância da classificação dos especialistas nucleares, corrigida pela chance [9].

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

Equação 7 – Precisão

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

Equação 8 – Concordância observada

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

Equação 9 – Concordância esperada

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

Equação 10 – Estatística kappa

A Tabela 5 exibe os resultados da avaliação de relevância dos especialistas nucleares para o tópico “área nuclear em geral”.

Tabela 5 – Avaliação de relevância para “área nuclear em geral”

		Especialista nuclear 2		
		Sim	Não	Total
Especialista nuclear 1	Sim	48	1	49
	Não	6	5	11
	Total	54	6	60

A Tabela 6 exibe os resultados da avaliação de relevância dos especialistas nucleares para o tópico “energia nuclear especificamente”.

Tabela 6 – Avaliação de relevância para “energia nuclear especificamente”

		Especialista nuclear 2		
		Sim	Não	Total
Especialista nuclear 1	Sim	43	1	44
	Não	8	8	16
	Total	51	9	60

A Tabela 7 exibe as métricas de avaliação computadas a partir das avaliações dos especialistas nucleares. Em resumo, o rastreador nuclear obteve 80% de precisão e um coeficiente kappa de 0,53 para o tópico “área nuclear em geral” que representa uma boa precisão com uma concordância moderada da avaliação dos especialistas. Para o tópico “energia nuclear especificamente”, o rastreador nuclear obteve 72% de precisão e um coeficiente kappa de 0,55 que também representa uma boa precisão com uma concordância moderada.

Tabela 7 – Resumo das métricas de avaliação

Métrica	**Área nuclear em geral**	**Energia nuclear especificamente**
Precisão	0,80	0,72
Estatística kappa	0,53	0,55
Concordância observada	0,88	0,85
Concordância esperada	0,75	0,66

5 CONCLUSÕES

Uma grande parte do tempo disponível para a realização deste trabalho foi utilizado na implementação dos algoritmos e estruturas de dados necessárias para o funcionamento do rastreador nuclear. Como exemplos, alguns dos algoritmos e estruturas de dados que foram implementados são: rede neural artificial perceptron multicamadas, algoritmo de treinamento de retro propagação do erro, função de similaridade por cosseno, estruturas de listas, estruturas de grafos, fronteira de busca, tokenizador, *stemmer*, analisador HTML, normalizador de URLs, coletor de páginas, algoritmo de busca por melhor escolha e o próprio algoritmo do rastreador nuclear. Além da razoável quantidade de algoritmos implementados foi necessário um esforço e atenção substancial no tocante a eficiência deles, pois eles foram executados em um computador pessoal de pequeno porte e deveriam ser suficientemente eficientes para conseguir recuperar um volume considerável de páginas utilizando recursos computacionais limitados. Deste modo, o tempo para a realização de testes mais exaustivos e refinamento do vocabulário nuclear e parâmetros gerais foi escasso. Assim, como uma extensão futura e complementação deste trabalho podem ser realizadas as seguintes tarefas:

- expansão e refinamentos do vocabulário nuclear;
- adição de novos tópicos nucleares para busca;
- pesquisa e adição de um algoritmo de *stemming* para língua portuguesa (se existir) e adaptação do vocabulário nuclear;
- avaliações dos especialistas nucleares mais detalhadas;
- utilização de outras páginas-semente;
- variações nos valores dos parâmetros de busca e da rede neural;
- avaliação detalhada e entendimento das diferenças de desempenho em relação ao algoritmo *naïve best-first*.

No tocante a avaliação dos resultados, embora o procedimento de avaliação tenha sido elaborado experimentalmente, concentrando-se nas avaliações de relevância dos especialistas nucleares, ao invés de ser elaborado formalmente sobre métodos estatísticos, consideramos estes resultados um indicativo preliminar positivo que o algoritmo rastreador nuclear é efetivo e eficiente para a busca e recuperação de páginas relacionadas ao domínio nuclear de forma autônoma e massiva, assim atingindo os objetivos propostos.

Uma importante perspectiva futura para extensão deste trabalho é a pesquisa e desenvolvimento de métodos e técnicas capazes de identificar e extrair informações subjetivas (ou, em outras palavras, opiniões) das páginas recuperadas, classificando-as de acordo coma sua polaridade (negativa, positiva ou neutra) com o objetivo de prover um meio de análise e monitoramento da aceitação pública da área nuclear e seus temas na Web, conforme já delineado em [5].

Por fim, é esperado como benefício deste trabalho desenvolver e fornecer um método potencialmente útil para alguns propósitos nucleares, como: recuperação de páginas na Web; buscas executadas com recursos computacionais limitados; recuperação de páginas recorrentes para detecção de mudanças; buscas na Web em tempo real; construção de grandes bases de conhecimento, corpus, e repositórios de informação para análises pós recuperação; descoberta de informações na Web; etc.

GLOSSÁRIO

Algoritmo de retro propagação do erro: algoritmo de treinamento de redes neurais artificias.

Aprendizagem por reforço: uma área da aprendizagem computacional interessada em como um agente deve tomar suas ações em um ambiente de forma tal que ele maximiza algum critério de recompensa cumulativa.

Base de conhecimento: um banco de dados que armazena conhecimento em uma forma interpretável por computadores, geralmente com o propósito de ter raciocínio dedutível aplicado a ele.

Busca e rastreamento web: programa de computador que busca e recupera informações da Web de forma automática, “visitando” suas páginas e “movendo-se” através dos hiperlinks que as conectam para acessar novas informações.

Busca em amplitude: busca que recupera as páginas nível por nível, utilizando uma fronteira de busca ordenada como uma fila para adicionar e remover as URLs em ordem FIFO (first-in, first-out).

Busca exaustiva ou busca não informada: procura exaustiva de soluções do problema no espaço de busca, por meio da verificação de todas as soluções possíveis.

Busca heurística ou busca informada: procura de soluções aproximadas do problema no espaço de busca, por meio da verificação de algumas soluções possíveis definidas de acordo com regras.

Busca por melhor escolha: busca que recupera as páginas por ordem de maior prioridade definida por uma função de avaliação heurística, utilizando uma fronteira de busca ordenada como uma fila de prioridade para adicionar e remover as URLs em ordem de maior importância.

Ciclo de busca: busca iniciada a partir de uma página-semente e executada até o atingimento de algum critério de parada.

Engenharia de conhecimento: disciplina que envolve a integração de conhecimento em sistemas de computação com o propósito de solucionar problemas complexos que normalmente requerem um alto nível de perícia humana.

Espaço de busca: conjunto de todas as soluções possíveis de um problema.

Estimativa de relevância: uma predição da relevância de uma página a um determinado tópico nuclear.

Etapa de treinamento: busca efetuada pelo rastreador nuclear parcialmente aleatória na Web para a construção de amostra de páginas diversificada que é utilizada para o treinamento da rede neural.

Etapa de recuperação: busca efetuada pelo rastreador nuclear para efetivamente recuperar páginas com conteúdo textual relacionado ao tópico nuclear, utilizando a rede neural previamente treinada.

Fronteira de busca: estrutura de dados na forma de uma lista de URLs das páginas que estão aguardando para serem recuperadas.

Função de similaridade por cosseno: a medida do cosseno do ângulo de dois vetores utilizada para comparar a similaridade textual de documentos, normalmente utilizada nas áreas de recuperação de informação e mineração de texto.

Grafo da web: modelo que representa as páginas da Web e seus hiperlinks por meio de um grafo direcionado, conexo e esparso.

Hiperlink: uma referência existente em uma página para outra página.

Hipertexto: ver página.

HTML: linguagem de marcação utilizada para produzir páginas na Web.

Iteração de busca: cada repetição do processo de busca compreendendo a recuperação de uma página.

Mecanismo de inferência: é o “cérebro” que o sistema especialista utiliza para raciocinar sobre a informação existente na base de conhecimento com o propósito de solucionar problemas complexos.

Modelo de espaço vetorial: modelo matemático para representação de documentos de texto como vetores em um grande espaço multidimensional onde cada uma das suas palavras corresponde a uma dimensão ou eixo no espaço vetorial.

Palavra-chave: termo com alta importância semântica para o domínio nuclear.

Pré-escore: uma estimativa de relevância efetuada antes da recuperação da página-alvo, computada utilizando somente os dados textuais e estrutura de hiperlinks da página-fonte a qual o hiperlink que referência a página-alvo está contido.

Página: um documento apropriado para a Web, escrito em HTML com capacidade hipertextual.

Página-alvo: página qual é referenciada por um hiperlink existente em outra página (página-fonte).

Página-fonte: página qual contem um hiperlink que referência outra página (página-alvo).

Páginas-semente: conjunto de páginas inicialmente informadas ao algoritmo rastreador, as quais são utilizadas como pontos iniciais da busca.

Pós-escore: uma estimativa de relevância efetuada após a recuperação da página-alvo, computada utilizando o conteúdo textual existente nela.

Rede neural artificial: modelo matemático inspirado nas redes neurais biológicas e que consiste de grupos de neurônios artificiais interconectados e processa informações utilizando uma abordagem conexionista para a computação. Em muitos casos, a rede neural artificial é um sistema adaptativo que altera sua estrutura durante a etapa de aprendizado. Elas são utilizadas para modelar relações complexas entre entradas e saídas ou para encontrar padrões nos dados.

Saco de palavras: modelo simplificado do texto onde este é representado como uma coleção não ordenada de palavras, considerando somente a sua frequência no texto.

Sistema especialista: sistema de computador que emula a habilidade de tomada de decisão de um especialista humano em um domínio específico de conhecimento, por meio da aquisição, representação e raciocínio sobre este conhecimento, com o propósito de resolução de problemas complexos, suporte a decisões ou provimento de recomendações neste domino de conhecimento.

Tópico nuclear: um assunto ou tema de interesse nuclear.

URL: endereço na Web da página-alvo referenciada pelo hiperlink.

Vocabulário nuclear: um catálogo de palavras relacionadas ao domínio nuclear.

World Wide Web, Web: um sistema de hipertextos interconectados acessados via Internet.

REFERÊNCIAS BIBLIOGRÁFICAS

1 GULLI, A.; SIGNORINI, A. **The Indexable Web is More than 11.5 Billion Pages**. In: INTERNATIONAL WORLD WIDE WEB CONFERENCE (WWW2005), 14, May 10-14, 2005, Chiba, Japan. ***Proceedings*...** p. 902-903.

2 RAMACHANDRAN, S. **Web metrics: Size and number of resources – Make the web faster**. 2010. Disponível em: <https://developers.google.com/speed/articles/web-metrics> Acesso em: 11 de Agosto de 2013.

3 SAGAN, C. **Cosmos**. New York, N.Y.: Ballantine Books, 1985.

4 LIU, B. **Web Data Mining: Exploring Hyperlinks, Contents, and Usage Data**. New York, N.Y.: Springer, 2007.

5 REIS, T.; BARROSO, A. C. O.; IMAKUMA, K. **Monitoring and Analysis of Nuclear Acceptance by Information Retrieval and Opinion Extraction on the Web**. In: 2011 INTERNATIONAL NUCLEAR ATLANTIC CONFERENCE (INAC 2011), October 24-28, 2011, Belo Horizonte, Brazil. ***Proceedings*...** v. ENIN, p. 16-29.

6 BRODERA, A.; KUMARB, R.; MAGHOULA, F.; RAGHAVANB, P.; RAJAGOPALANB, S.; STATAC, R.; TOMKINSB, A.; WIENERC, J. **Graph Structure in the Web**. **Computer Networks**, v. 33, p. 309-320, 2000.

7 COOPER, C.; FRIEZE, A. **A General Model of Web Graphs**. In: EUROPEAN SYMPOSIUM ON ALGORITHMS (ESA), 9, August 28-31, 2001, Aarhus, Denmark. ***Proceedings*...** p. 500-511.

8 COOPER, C.; FRIEZE, A. **Crawling on Web Graphs**. ACM SYMPOSIUM ON THEORY OF COMPUTING (ACM STOC 2002), 34, May 19-21, 2002, Montréal, Québec, Canada. ***Proceedings*...** p. 419-427.

9 MANNING, C. D.; RAGHAVAN, P.; SCHÜTZE, H. **An Introduction to Information Retrieval**. Cambridge, United Kingdom: Cambridge University Press, 2008.

10 DAVISON, B. D. **Topical Locality in the Web**. INTERNATIONAL ACM CONFERENCE ON RESEARCH AND DEVELOPMENT IN INFORMATION RETRIEVAL (ACM SIGIR 2000). 23, July 24-28, 2000, Athens, Greece. ***Proceedings*...** p. 272-279.

11 DAVISON, B. D. **Topical Locality in the Web: Experiments and Observations**. Technical Report DCS-TR-414, Department of Computer Science, Rutgers University 2000.

12 CRASWELL, N.; HAWKING, D.; ROBERTSON, S. **Effective Site Finding using Link Anchor Information**. INTERNATIONAL ACM CONFERENCE ON RESEARCH AND DEVELOPMENT IN INFORMATION RETRIEVAL (ACM SIGIR 2000), 24, September 9-12, New Orleans, Louisiana, USA. ***Proceedings*...** p. 250-257.

13 PANT, G.; SRINIVASAN, P.; MENCZER, F. **Crawling the Web**. **Web Dynamics**, v. 2, p. 153-177, 2004.

14 MENCZER, F.; PANT, G.; SRINIVASAN, P. **Topical Web Crawlers: Evaluating Adaptive Algorithms**. **ACM Transactions on Internet Technology**, v. 4, n. 4, p. 378-419, 2004.

15 PANT, G.; SRINIVASAN, P.; MENCZER, F. **Exploration versus exploitation in topic driven crawlers**. In: WWW02 WORKSHOP ON WEB DYNAMICS, 2002.

16 PINKERTON, B. **Finding what people want: Experiences with the WebCrawler**. In: INTERNATIONAL WORLD WIDE WEB CONFERENCE (WWW1), 1, May 25-27, 1994, CERN, Geneva, Switzerland. ***Proceedings*...** p. 821-829.

17 CHO, J.; GARCIA-MOLINA, H.; PAGE, L. **Efficient Crawling Through URL Ordering**. In: INTERNATIONAL WORLD WIDE WEB CONFERENCE (WWW7), 7, April 14-18, 1998, Brisbane, Australia. ***Proceedings*...** p. 161-172.

18 NAJORK, M.; WIENER, J. L. **Breadth-first search crawling yields high-quality pages**. In: INTERNATIONAL WORLD WIDE WEB CONFERENCE (WWW10), 10, May 1-5, 2001, Hong Kong. ***Proceedings*...** p. 114-118.

19 HERSOVICI, M.; JACOVI, M.; MAAREK, Y. S.; PELLEG, D.; SHTALHAIM, M.; UR, S. **The Shark-Search Algorithm - An Application: Tailored Web Site Mapping**. In: INTERNATIONAL WORLD WIDE WEB CONFERENCE (WWW7), 7, April 14-18, 1998, Brisbane, Australia. ***Proceedings*...** p. 317-326.

20 MENCZER, F.; PANT, G.; SRINIVASAN, P. **Evaluating Topic-driven Web Crawlers**. In: INTERNATIONAL ACM CONFERENCE ON RESEARCH AND DEVELOPMENT IN INFORMATION RETRIEVAL (ACM SIGIR 2000), 24, September 9-12, New Orleans, Louisiana, USA. ***Proceedings*...** p. 241-249.

21 CHAKRABARTI, S.; BERG, M. VAN DEN; DOM, B. **Focused crawling: A new approach to topic-specific Web resource discovery**. In: INTERNATIONAL WORLD WIDE WEB CONFERENCE (WWW8), 8, May 11-14, 1999, Toronto, Canada. ***Proceedings*...** p. 1623-1640.

22 DILIGENTI, M.; COETZEE, F.; LAWRENCE, S.; GILES, C. L.; GORI M. **Focused crawling using context graphs**. In: INTERNATIONAL CONFERENCE ON VERY LARGE DATABASES (VLDB 2000), 26, September 10-14, 2000, Cairo, Egypt. ***Proceedings*...** p. 527-534.

23 BRIN, S.; PAGE, L. **The anatomy of a large-scale hypertextual Web search engine**. **Computer Networks**, v. 30, p. 107-117, 1998.

24 DE BRA, P.; POST, R. **Information retrieval in the World Wide Web: Making Client-based Searching Feasible**. In: INTERNATIONAL WORLD WIDE WEB CONFERENCE (WWW1), 1, May 25-27, 1994, CERN, Geneva, Switzerland. ***Proceedings*...** p. 183-192.

25 MENCZER, F. **Complementing search engines with online Web mining agents**. **Decision Support Systems**, v. 35, p. 195-212, 2003.

26 JACKSON, P. **Introduction to Expert Systems**. Boston, USA: Addison-Wesley, 1998.

27 HAYKIN, S. **Neural Networks: A Comprehensive Foundation**. New York, USA: Prentice Hall, 1998.

28 SALTON, G.; MCGILL, M. J. **An Introduction to Modern Information Retrieval**. New York, USA: McGraw-Hill, 1983.

29 PORTER, M. F. **An Algorithm for Suffix Stripping**. **Program: electronic library and information systems**, v. 14, p. 130-137, 1980.