ISSN 2007-9737

Determining the More Adequate Web Page Node for Advertising Placement

J. Guadalupe Ramos[1], Jéssica López Espejel[2,3], Rogelio Ferreira Escutia[1], Heberto Ferreira Medina[1,4]

[1] Tecnológico Nacional de México,
Instituto Tecnológico de Morelia,
Mexico

[2] Université Paris 13,
France

[3] CEA LIST NANO-INNOV,
France

[4]Universidad Nacional Autónoma de México,
Instituto de Investigaciones en Ecosistemas y Sustentabilidad,
Mexico

{jguadaluperamos, rogelio}@itmorelia.edu.mx, jessica.lopez-espejel@cea.fr,
hferreir@iies.unam.mx

Abstract. For many web sites, money earning is crucial for keeping the content production. Advertising is one of the most common strategies for web content monetization. To determine where is the more appropriate place for ad location is an essential task to get a gentle introduction of the commercial information. There are algorithms based on keywords appearing within text; however we consider that implicit meaning is more adequate for a better harmony between content advertising. In this work, we present a formal method that determines the best place for advertising location. For this, we explore the underlying tree-like structure of a web page, we extract the text from each (X)HTML node and compute the semantic similarity (by employing latent semantic analysis) w.r.t., the advertising source text. We introduce a unique formula for the numerical calculation of the web page node relevance. We think it could be used for measuring the concordance among web page nodes and the commercial information and for the design of dynamic ads insertion methods.

Keywords. Semantic similarity, latent semantic analysis, dynamic advertising.

1 Introduction

Advertising aggregation into web pages is a common strategy for monetization. Too many web sites deliver free content and get income using advertising payment. This kind of monetization allows them to operate and to keep information production. Sometimes advertising causes displeasure for diverse reasons, for instance, when the amount of publicity is excessive and others when the commercial message is discordant for the information read.

Advertising aggregation is an important issue that must be analyzed from different points of view. On one side, commercial information is necessary for guaranteeing the economic survival of web sites. On the other hand, users read information about their interest, i.e., they browse and consume web content according to a specific motivation. We consider that methods that preserve the *thematic sense* between web content and an advertising message can reduce the reader's displeasure when publicity is shown.

In this work, we consider the thematic sense as a set of topics which are correlated. For instance, a football match, which is transmitted by TV, regularly is related to the consuming of snacks and beer and, perhaps, within a friend's meeting. They are different facts, a football match, snacks, and beer drinking and friends meeting. However, they are correlated. A formal approach that is useful to find out correlations into data is the technique of Latent Semantic Analysis (LSA [4]).

Here we can state the proposal for this work. We think that a gentle introduction to the publicity can be reached if the sense of the web content concerning the advertising message can be assured. For this reason, we introduce a formal method, to measure the level of correlation among both messages, expressed within text fragments.

Our proposal is motivated, too, by the behavior of new styles of inRead advertising, which dynamically presents commercial information when a user browses a web page. In this way, web page fragments could be semantically compared with the advertising message, and then, the more similar web page fragment could be determined for the publicity location.

Now an important issue must be faced, which web page fragments should be considered for the analysis?

We could treat the text source as plain documents. However, web text is formatted using (X)HTML, i.e., the information is organized in formatting nodes which frequently provide certain implicit unity (all the information in a node is related) imposed by the web designer.

In this way, the main path for this work is going to be established in the context of approaches that exploit the text fragments within a web page and written in natural language. For instance„ those devoted to web filtering such as [1, 10, 17]. Regardless of this focus, formulas here presented for calculus of semantic similarity among text fragments can be applied in a seamless way to any pair of text fragments, and thereby we could analyze sentences of paragraphs instead of web page text fragments.

We consider that methods preserving the unity of (X)HTML nodes are more acceptable. Some works following this approach are [1, 5, 14, 16]. For instance, in [1], a method for information extraction from web pages considering the distance between (X)HTML nodes as measurement of analysis is introduced. However, standard tests of similarity, in the setting of natural language processing, as a basis for producing text measurements, are not employed. We believe that it is necessary to test classical techniques of natural language processing to establish an adequate comparison framework.

In this work, we present a formal method for automatic measuring of semantic similarity among an advertising text and a web page node, based on techniques of similarity employed in natural language processing and inspired on the notion of latent semantic analysis. Our formal method requires only once the calculations of LSA, and then it returns the more relevant web page node for advertising placement.

One of the main contributions of this paper is the definition of one formula to determine the semantic similarity of one text excerpt w.r.t. an advertising text. The formula is not affected by the size of text fragments.

The rest of the paper is organized as follows. In Section 2, Theoretical foundations of our work. In Section 3, we introduce a formal technique for relevance calculation based on semantic similarity and tree-like structure of web pages. Then, in Section 5, we describe a prototype and a set of experiments. Finally, we present related work and conclusions in Section 6.

2 Theoretical Foundations

In this section standard theoretical basis of our work are introduced.

Salton et al. originally introduced the vector space model for automatic indexing in [15] and it is considered a standard representation technique in information retrieval setting where stored entities (documents) are compared with each other. Given a text document d, a dictionary of *terms* is a set whose elements are the different *words* in the document d. $\overrightarrow{V}(d)$ denotes the vector associated to document d, whose components are the weights for each element in the dictionary.

In a vectorial representation, typographic symbols such as "," or "-" are ignored. The well-known stop words are treated in the same way. For web pages, formatting labels are removed.

2.1 Latent Semantic Analysis

Latent Semantic Analysis (LSA) is a theory and technical method for extracting and representing the contextual-usage meaning of words by means of statistical computations applied to a large corpus of text [7]. Hence, the underlying idea is that the aggregate of all the word contexts in which a given word does and does not appear provide a set of mutual constraints that largely determines the similarity of meaning of words and sets of words to each other [8].

The first step of LSA consists of the construction of a matrix representation of text, i.e., the matrix M, in which columns are employed for modeling documents and rows for terms (words). Each row i represents a specific term as well as each column j represents a document. Thus, each cell $M_{i,j}$ stands for the frequency in which every term i appears in the document j. Term frequency tf can be substituted by some other weighting scheme as for instance $tf.idf$ [11].

Next, LSA performs the *Singular Value Decomposition* process (SVD) on the matrix M. In the original matrix M, terms and documents are mutually dependent between them. In SVD, a rectangular matrix M is decomposed in the product of the other three matrixes, i.e., $M = USV^T$. New matrixes will be formed by *singular vectors* or *singular values*. Resultant matrix U will contain a vector representation of the terms, which will have linear independency w.r.t. the relationship with the documents, while V will contain the vector representation of the documents whose components will be linearly independent w.r.t. the relationships with terms in M.

Finally, S is a diagonal matrix in which singular values are found in descending order, and they represent the relationships between the other matrixes. The highest values in S represent the relations with major variance among terms and documents.

After SVD decomposition, the original matrix M can be rebuilt as of the matrix product of the resultant three matrixes. When a reconstruction over matrixes is performed it is possible to choose only the first k elements of the matrixes, i.e., $M' = U_k S_k {V_k}^T$, with this, a new matrix M' is obtained, in which the noise introduced by irrelevant relations is eliminated. Thus, the new values $M'_{i,j}$ unveil latent relationships among terms and documents, the so called human cognitive relations in [8]. SVD is implemented in many different mathematical tools, we use JAMA, a basic linear algebra package for Java [12].

Example 1 *Let us consider the documents:*

$d0 =$ *My computer. It has branded software.*

$d1 =$ *A PC is useful. Only with branded software.*

$d2 =$ *A PC (as computer) hardware. It can be generic.*

$d3 =$ *Branded software and generic hardware. Both, go well with my computer.*

The dictionary of the document collection is {computer, software, branded, PC, hardware, generic}. According to previous speech, the first row in M is for the representation of the term $computer$ (second one for software, and so on w.r.t. the dictionary) and the column 0 will be for the first document, then $M_{0,3}$ stands for the number of times that computer appears in document 3, and so on. By applying the technique SVD, $M = USV^T$ is obtained:

$$M = \begin{bmatrix} 1 & 0 & 1 & 1 \\ 1 & 1 & 0 & 1 \\ 1 & 1 & 0 & 1 \\ 0 & 1 & 1 & 0 \\ 0 & 0 & 1 & 1 \\ 0 & 0 & 1 & 1 \end{bmatrix}, U = \begin{bmatrix} 0.49 & 0.21 & 0.35 & 0.77 \\ 0.47 & -0.50 & 0.02 & -0.17 \\ 0.47 & -0.50 & 0.02 & -0.17 \\ 0.26 & 0.14 & -0.93 & 0.22 \\ 0.35 & 0.47 & 0.08 & -0.39 \\ 0.35 & 0.47 & 0.08 & -0.39 \end{bmatrix},$$

$$S = \begin{bmatrix} 3.19 & 0.0 & 0.0 & 0.0 \\ 0.0 & 1.74 & 0.0 & 0.0 \\ 0.0 & 0.0 & 1.19 & 0.0 \\ 0.0 & 0.0 & 0.0 & 0.6 \end{bmatrix}, V^T = \begin{bmatrix} 0.45 & 0.38 & 0.46 & 0.67 \\ -0.46 & -0.50 & 0.73 & 0.08 \\ 0.32 & -0.75 & -0.36 & 0.45 \\ 0.70 & -0.22 & 0.35 & -0.58 \end{bmatrix}.$$

Considering $k = 2$ and reconstructing M:

$$M' = \begin{bmatrix} 0.542 & 0.413 & 0.986 & 1.085 \\ 1.067 & 0.993 & 0.044 & 0.929 \\ 1.067 & 0.993 & 0.044 & 0.929 \\ 0.265 & 0.195 & 0.557 & 0.577 \\ 0.133 & 0.019 & 1.115 & 0.821 \\ 0.133 & 0.019 & 1.115 & 0.821 \end{bmatrix}.$$

ISSN 2007-9737

If the similarity between vectors representing the rows 0 and 3 of M is calculated, i.e., the comparison of similarity among terms "PC" and "computer," the result is 0.4, while, 0.99 in M'. Although the coincidence of both terms is given only in one document, correlations in the rest of documents allow to unveil the major latent *similarity. This insights of relationships will be exploited later by the algorithms exposed in this document.*

3 Calculating the More Relevant Web Page Node for Advertising

Now, we describe the main contribution of this work, i.e., a formal technique for calculating the more relevant web page node for advertising.

The goal of the technique is to find out the web page node (of a specific URL) that presents the best semantic similarity w.r.t. a given advertising text. An *advertising text* is a fragment of text in natural language, which is regularly inserted in a web page to be exposed.

3.1 Semantic Similarity based on LSA

Measuring the similarity of meanings between two blocks of text, on the one hand, the text of a fragment on a web page and the other, the advertising text, can be defined as an act of measuring the semantic similarity between texts.

LSA is fully documented for comparison of similarity among documents of terms. However, advertising placement requires being determined by analyzing several text pieces from a web page. In this setting, a procedure for calculation of similarity among text excerpts (of variable size) instead of documents is needed.

Our approach for application of LSA through M' and the discrimination of a set of text excerpts of different size implies several steps. We follow the same phases of [13] and some definitions, to define a unique formula for semantic computing similarity and its application towards web page fragments.

We require to incorporate the semantic LSA information towards distinct sized texts and also, for the aggregation of the LSA semantic information upon the similarity calculation.

In this way, our first definition for semantic similarity measurement is called *relative similarity*: If a text fragment f_d from document d is being analyzed, for each term i in collection we put in vector $\overrightarrow{V}(f_d)$ value 0 if that word does not appear in f_d, and we put value $M'[i][d]$ if the word appears in f_d, i.e, the corresponding values in vector $\overrightarrow{V}(f_d)$ are mapped from column d in M'.

For instance, the vector for the text fragment f_{d_0} "It has branded software" from d_0 in the Example 1 is $\overrightarrow{V}(f_{d_0}) = \langle 0, 1.067, 1.067, 0, 0, 0\rangle$. According to the dictionary {computer, software, branded, PC, hardware, generic}, terms "software" and "branded" appear in d_0 and their values are taken from $M'[0][i]$

Let us remember that we are computing similarities among text fragments of several sizes, this is the reason why we apply a mapping of values from the document in M' to $\overrightarrow{V}(f_d)$, i.e., we do not take the whole document. We could use relative similarity for comparing text fragments; however, when there are not common terms in vectors, then the result produced is 0. Hence, the strategy of relative similarity needs to be improved.

We must consider that LSA gathers contextual information w.r.t. the terms in a collection. A certain kind of measurement of correlations should be conveniently incorporated into a new scheme of comparisons among text fragments. This is going to be examined in the following.

Definition 2 (Mutual similarity matrix $\mathcal{M}$) *Given the reconstructed matrix M', where m is the number of rows and n is the number of columns in M'. Let the squared $m \times m$ $\mathcal{M}$ be the mutual similarity matrix. Such that $\mathcal{M}_{i,j}$ contains the similarity value $sim(\overrightarrow{V}(t_i), \overrightarrow{V}(t_j))$ where $\overrightarrow{V}(t_i) = \langle M'_{i,0}, \ldots, M'_{i,n-1}\rangle$, $i \in \{0, \ldots, m-1\}$ and $\overrightarrow{V}(t_j) = \langle M'_{j,0}, \ldots, M'_{j,n-1}\rangle$ with $j \in \{0, \ldots, m-1\}$.*

Intuitively speaking $\mathcal{M}$ is a data structure which contains the cosine similarity values between all the terms in the reconstructed matrix M'.

Example 3 *Let us consider the Example 1, the mutual similarity matrix of M' is as follows:*

$$\mathcal{M} = \begin{bmatrix} 1.000 & 0.730 & 0.730 & 0.999 & 0.920 & 0.920 \\ 0.730 & 1.000 & 1.000 & 0.692 & 0.405 & 0.405 \\ 0.730 & 1.000 & 1.000 & 0.692 & 0.405 & 0.405 \\ 0.999 & 0.692 & 0.692 & 1.000 & 0.940 & 0.940 \\ 0.920 & 0.405 & 0.405 & 0.940 & 1.000 & 1.000 \\ 0.920 & 0.405 & 0.405 & 0.940 & 1.000 & 1.000 \end{bmatrix},$$

Here $\mathcal{M}[4][5] = 1.000$ which is the result of similarity among term 4 and 5 in M', i.e., the similarity between hardware*, in the row 4, and* generic*, in the row 5 of M'. This is due to both terms appear always in pair in the documents of Example 1. Similarities between* computer *and* PC *are more obvious. Consequently diagonal matrix is composed by only ones.*

In the matrix of mutual similarities the information that shows explicitly the numerical similarities among terms in a collection of documents is located. Now, it is necessary to exploit such measures from $\mathcal{M}$ to compute the calculus of similarity of small fragments of text and guaranteeing that term correlations are taken into account.

For this, we apply a simple idea: if a term $t0$ is highly related (closer to one) with term $t4$ according to $\mathcal{M}$ and, if we have a text fragment f which contains $t0$ and does not contain $t4$ then, in order to execute the calculation of similarity of f we will aggregate to the vector $\overrightarrow{V}(f)$ the values for $t0$ and also for $t4$, i.e., $\overrightarrow{V}(f) = \langle 1, 0, 0, 0, threshold \rangle$ where $threshold$ is an arbitrary value in order to identify the limit for considering a meaningful relation between terms. Formally:

Definition 4 (threshold, meaningful relation) *Let the real number μ be threshold such that $0 < \mu \leq 1$, and $\mathcal{M}_{i,j}$ is a meaningful relation between terms t_i and t_j in M' $\iff$ $\mathcal{M}_{i,j} \geq threshold$, where $i, j \in \{0, \ldots, m-1\}$ and m is the number of rows in M'.*

The threshold defines the least numerical limit of similarity measure for considering a relation between terms as important. Each row in M' represents a term in the collection of documents, and the numerical relation between the terms t_i, t_j is joined in $\mathcal{M}_{i,j}$

To incorporate main correlations in the vectors of text fragments, we define the concept of augmented vector.

Definition 5 (augmented vector) *Let $\overrightarrow{V\triangle}(f)$ be the augmented vector of a text fragment f in the setting of a collection of documents, such that $\overrightarrow{V\triangle}(f) = \langle v_0, \ldots, v_{m-1} \rangle$ where v_i is the corresponding weight of $t_i \in dict(collection)$ in the fragment f, which is obtained from:*

$$v_i = \begin{cases} \textit{weight}(t_i, f) & \textit{if } t_i \in dict(f) \\ \textit{threshold} & \textit{if } t_i \notin dict(f) \textit{ but there is a} \\ & \textit{meaningful relation with} \\ & \textit{some } t_j \in dict(f) \\ \textit{0} & \textit{otherwise}, \end{cases}$$

where $i, j \in \{0, \ldots, m-1\}$, m is the number of rows in M' and $weight(t_i, f)$ is a function that returns the corresponding weight of t_i in f.

When a vector is augmented, not only appears the weight of those terms present in a fragment of text; also, the threshold value is incorporated in the position corresponding for terms which are not present in the fragment but, they maintain a correlation (meaningful relation) with some other terms present in the fragment. Roughly speaking, in an augmented vector, there are values for its terms and their correlated terms. Now, for computing similarity we overload standard cosine similarity.

Definition 6 (augmented similarity) *The augmented similarity between fragments of text t_1, t_2 is defined by:*

$$augsim(t_1, t_2) = \frac{\overrightarrow{V\triangle}(t_1) \cdot \overrightarrow{V\triangle}(t_2)}{|\overrightarrow{V\triangle}(t_1)||\overrightarrow{V\triangle}(t_2)|},$$

where the numerator represents the dot product of the augmented vectors $\overrightarrow{V\triangle}(t_1)$ and $\overrightarrow{V\triangle}(t_2)$, and the denominator is the product of their Euclidean lengths.

Here, augmented similarity returns the similarity calculus of two vectors, which were augmented from correlations among terms.

Finally, intending to harvest the best properties of relative similarity and augmented similarity, we define semantic similarity.

ISSN 2007-9737

Definition 7 (semantic similarity) *Given $t1$ and $t2$, and $compsim(t_1, t_2) = log(relsim(t_1, t_2) + 0.001) + log(augsim(t_1, t_2) + 0.001)$, let:*

$$semsim(t_1, t_2) = \frac{1}{1 + e^{-compsim(t_1,t_2)}},$$

be the semantic similarity between text fragments t_1 and t_2.

Fundamentally we apply the product of relative similarity with augmented similarity, which we call $compsim$. Relative similarity has two graceful properties. First, it takes advantage of LSA correlations, and on the other hand, it computes greater values in the case of the presence of common terms. Then, an initial idea is to compute the product of $relsim$ and $augsim$; however, eventually, the calculus could produce 0, to redress this, we apply a logarithmic addition. Finally, in order to normalize the values of calculations in the range of 0 and 1, we introduce the well known sigmoid function i.e., $f(x) = 1/(1+e^{-x})$ where x is our $compsim$.

In this way, semantic similarity combines the properties of both measurements. Relative similarity is more affected by the coincidence of terms in the *ad text* w.r.t. the analyzed fragment, while augmented similarity, offers greater values when a text fragment brings more relationships with its terms. In other words, it privileges the number of correlations of its terms.

At this point, a useful formal scheme for computing semantic similarity between text fragments has been completely defined. Next, a set of rules for text fragmentation to determine the more relevant text from a web page is required.

4 Extraction of Text Fragments from a Web Page

To define a strategy to extract text fragments from a web page, we could suggest many criteria. However, there is the standard scheme DOM, which organizes the content of a web page in hierarchical nodes.

Example 8 *Let us consider the following web page:*

```
<html>
 <head>
    <title>Semantics within web pages</title>
 </head>
 <body>
   Semantics is related with meaning.
   <div>
     Semantics can be extracted in two ways:
     <span>
       a) Driven by metadata
     </span>
     <span>
       b) By using mathematical techniques.
          One of them is:
          <strong>
                  latent semantic analysis.
          </strong>
     </span>
   </div>
 </body>
</html>
```

DOM model is an adequate scheme. It stands for Document Object Model, which is a W3C standard platform—and language-neutral interface that allows programs and scripts to dynamically access and update the content, structure, and style of documents [9].

Its corresponding DOM representation is visualized in a graphical mode in Figure 1. We can see a web page as a tree-like data structure where each node is an (X)HTML element, i.e., a (X)HTML tag with its contained text and its attributes, furthermore, its children are the embedded (X)HTML labels.

In a DOM data structure, if a node contains nested (X)HTML labels, there is a relation of *embedding* between the node and its children. For instance in Example 8, `div` node embeds two `span` node children.

Definition 9 (web page tree) *A web page tree $(\mathcal{V}, \mathcal{E})$ is a directed, acyclic graph whose vertices $\mathcal{V}$ are (X)HTML elements and $\mathcal{E} \subseteq \mathcal{V} \times \mathcal{V}$ is a set of ordered pairs $(v \to v') \in \mathcal{E}$ called edges, where v embeds v', and there is a $root(\mathcal{V}, \mathcal{E})$ node which is not embedded.*

ISSN 2007-9737

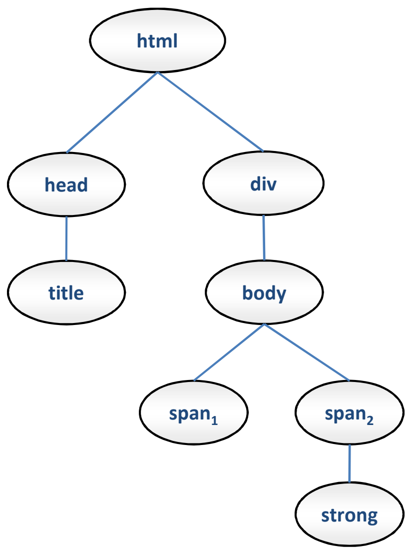

Fig. 1. The DOM tree of Example 8, visualized in a graphical way

Given a web page tree $P = (\mathcal{V}, \mathcal{E})$ and two nodes $v_1, v_n \in \mathcal{V}$, if there is a sequence $v_1, v_2, \ldots, v_n$ of nodes in P where $(v_i, v_{i+1}) \in \mathcal{E}$ for $1 \leq i \leq n-1$, then we say that there is a *path* from v_1 to v_n in P. Given $u, v \in \mathcal{V}$ we say that the node v is *reachable* from u if there is a path from *u* to *v*. We use $\rightarrow^*$ to denote the reflexive and transitive closure of $\rightarrow$.

Now we define candidate branch of a web page tree in order to compute semantic similarity tests w.r.t. the web user query.

Definition 10 (candidate branch) *Let* $P = (\mathcal{V}, \mathcal{E})$ *be a web page tree, then* $B = (\mathcal{V}', \mathcal{E}')$ *is a candidate branch where* $\mathcal{V}' = \{v \in \mathcal{V} | v \neq root(P)\} \cup \{w | w$ *is reachable from* $v\}$ *and* $\mathcal{E}' = \{(u_1, u_2) | (u_1, u_2) \in \mathcal{E}$, *and* $u_1, u_2 \in \mathcal{V}'\}$

Hence, each node (except the root) of a web page tree and its children is a candidate branch.

Definition 11 (web text) *Let* $B = (\mathcal{V}, \mathcal{E})$ *be a candidate branch, then* $text(B)$ *is its web text.*

Web text is necessary to get text excerpts for applying semantic similarity calculations.

Example 12 *Let us consider the web page tree of Example 8, then there are 7 candidate branches, i.e., those which are compound by the following (X)HTML elements:*

```
B1={head, title}
B2={title}
B3={body, div, span1, span2, strong}
B4={div, span1, span2, strong}
B5={span1}
B6={span2, strong}
B7={strong}
```

Their contained texts were not drawn. Hence, the set of candidate branches is $\{B_1, B_2, B_3, B_4, B_5, B_6, B_7\}$, *and with respect to the function* $text$, *for instance, text(B_6) returns the text excerpt "By using mathematical techniques. One of them is: latent semantic analysis." and text(B_7) returns the string: "latent semantic analysis.", i.e., we extract the text without (X)HTML labels.*

Definition 13 (semantically relevant nodes) *Given* $L = \{< B, s > \mid B$ *a candidate branch,* A *an advertising text and* $s = semsim(text(B), A)$ $\}$ *be the ordered set of semantically relevant nodes induced by the total order* $\succeq$ *where* $< B_x, s_x > \succeq < B_y, s_y >$ *if* $s_x \geq s_y$.

Example 14 *Here the shape of the semantically relevant nodes is basically an ordered list composed by candidate branches and their corresponding semantic similarity w.r.t. an* ad text, *for instance:* $L = \{< \{span_2, \ldots\}, 0.53 >, < \{div, \ldots\}, 0.5 >, < \{body, \ldots\}, 0.42 >, < \{span_1\}, 0.28 >, < \{strong\}, 0.28 >, < \{head, \ldots\}, 0.0 >, \ldots\}$, *would show the branches (without text drawing) and their semantic similarity.*

Once semantically relevant nodes of a web page were computed, we are able to offer the most significant text excerpts as a result of an *ad text*.

Next, we present a definition of *relevant web page node* for advertising placement. For this reason, we require some auxiliary functions: $highest(L)$ returns the web page node with greater semantic similarity found in a set of semantically relevant nodes. In the above example, the function would return $< \{span_2, \ldots\}, 0.53 >$. Formally,

ISSN 2007-9737

Definition 15 (relevant web page node) *Given a web page url and an advertising text A, let $node = highest(L)$ be the more relevant web page node for advertising placement, where L are the semantically relevant nodes of url with respect to A.*

In Figure 2 we present an algorithm to determine the more relevant web page node for advertising location from a given web page.

The first step of the algorithm prepares a collection from contextual documents. Then, it constructs the matrix M by indexing text from the collection. After, it computes singular value decomposition, reconstructs M', and calculates the mutual similarity matrix $\mathcal{M}$, once this was done, it has all elements for computing semantic similarity between nodes and advertising text. There, we introduce a function: $semantically\_relevant\_nodes(w, A)$ which computes the list of semantically relevant nodes from a particular web page and by considering A as text advertising.

The last part of the algorithm extracts the highest semantically related web page node w.r.t. the advertising text and returns it. In other words, the algorithm determines which web page node has more semantic similarity w.r.t. an advertising text.

5 Prototype and Experiments

In this section, some aspects of the practical approach for web page node relevance calculation are presented, a relevant node is the more semantically related w.r.t. a text advertising. The computational tool is shown in Figure 3.

Once the target web page is defined, the web page is downloaded, for this, a parser is employed. Jericho [6] allows us to retrieve and to filter web pages and their text. Inclusive, it is possible to walk through their tree structures, visiting each node, and recovering their text. DOM node tree is visited to take one node, each time, for analysis. Then, DOM node and advertising text are augmented, and finally, a value of semantic similarity is computed. The node that presents the highest semantic similarity is chosen as the more appropriate place for particular commercial information.

Figure 4 is devoted to the natural language processing phase explanation. The first step refers to text preparation. Naturally, the tool requires a set of documents whose content must be in the same context as text advertising. It is crucial because LSA will calculate correlations between terms, which is useful for measuring semantically related texts, and to be exploited in our relevance analysis.

Then, the standard actions of natural language processing are shown: text cleaning, stop words deleting, and so on. For that phase, a proper library has been developed. Each document is treated, next, matrix M is formed by indexing text from the collection of documents. SVD calculus is performed by using a library of the third part, M' is reconstructed, and the mutual similarity matrix is computed. Roughly speaking, this process performs latent semantic analysis.

5.1 Experiments

In this section, we describe an experiment performed upon the prototype, and correspondingly upon the formal technique. The collection was composed of text documents from a set of web pages whose links are the following.

1. `http://lsa.colorado.edu/whatis.html`
2. `https://en.wikipedia.org/wiki/Latent_semantic_analysis`
3. `http://recommender-systems.org/latent-semantic-indexing/`

In this experiment, all web pages compose the collection of documents for LSA feeding. Next, we employ a little paragraph of text advertising to determine the more semantically related web page node (in each URL), i.e., the more relevant node for advertising placement.

Each web page was downloaded, and their DOM nodes were extracted by means of our DOM parser (based on Jericho [6]), then a text file with the set of nodes from each URL was prepared. Next, the whole process of the formal technique introduced in Section 3 was computed to determine the semantic similarity of every node and as a consequence, the best adequate place for commercial information.

ISSN 2007-9737

Input: U a set of contextual documents; w the target web page for advertising; A, an advertising text; k dimensions for matrix reconstruction
Output: $node$, the relevant web page node
Initialization: $Collection := \{\}$
Begin
  For each $u \in U$
    let $d_u := text(u)$
    $Collection := Collection \cup \{d_u\}$
  let M := **indexing as of** $Collection$
  let USV^T := $SVD(M)$
  let $M' := U_k S_k V_k^T$
  let $\mathcal{M}$:= **mutual similarity matrix of** M'
  let $L_w := semantically\_relevant\_nodes(w, A)$
  let $node := highest(L_w)$
End
Return: $node$

Fig. 2. An algorithm for computing the more semantically related web page node for advertising location

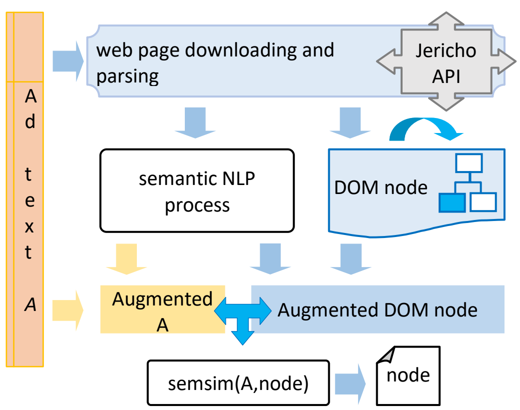

Fig. 3. A tool for web page node relevance calculation

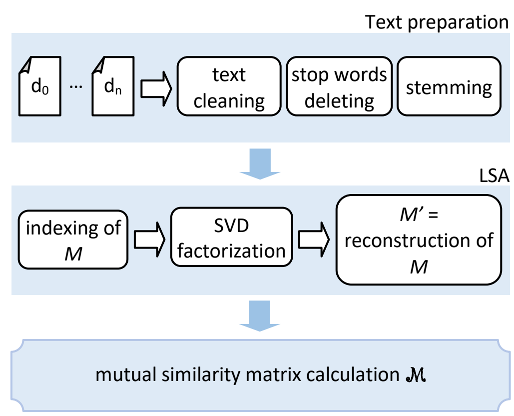

Fig. 4. Semantic NLP process

The advertising text is: `singular value decomposition (SVD) to the matrix`; for this reason, we developed a tool that receives a text and returns each text fragment and its corresponding measurements. Thereby the decision is taken by considering the web page node with the highest semantic similarity.

In the Table 1 a set of measurements are presented, there, a series of 26 nodes f from `http://lsa. colorado.edu/whatis.html` are put through testing.

The first calculus shown is `sim`, which represents the standard cosine similarity between the ad text an the analyzed node. The second one is the result of the relative similarity `relsim`. The third one represents the augmented similarity `augsem`. Next column presents results of composed `compsim` similarity, i.e., the product of `relative` and `augsem`, then `semsim` is calculated by using the logarithmic approach. `semsim` unveils the result of the query, i.e., fragment 22 from the `http://lsa.colorado.edu/whatis.html` URL. For each node of the web

ISSN 2007-9737

Table 1. Measurements for `http://lsa.colorado.edu/whatis.html` nodes

f	sim	$relsim$	$augsim$	$compsim$	***semsim***	terms	% terms	text of f
1	0.14	0.137	0.336	0.046	0.2090	508	30.71	What is LSA? What is ...
2	0	0	0.091	0	0.0173	1	0.06	What is LSA?
3	0	0	0.862	0	0.0446	17	1.02	Note: If you linked ...
4	0	0	0.892	0	0.0452	9	0.54	click here to open ...
5	0	0	0.997	0	0.0473	3	0.18	The information on this page is based
6	0	0	0.867	0	0.0447	9	0.54	Landauer, T. K., Foltz, ...
7	0	0	0.888	0	0.0451	4	0.24	which is available for ...
8	0	0	0.780	0	0.0428	68	4.11	Latent Semantic Analysis (LSA) is a ...
9	0	0	0.995	0	0.0473	3	0.18	Latent Semantic Analysis
10	0	0	0.091	0	0.0174	1	0.06	(LSA)
11	0.21	0.154	0.804	0.124	0.2882	70	4.23	Research reported in, and ...
12	0	0	0.997	0	0.0474	2	0.12	semantic space
13	0	0	0.833	0	0.0440	33	1.99	LSA can be construed ...
14	0	0	0.753	0	0.0422	72	4.35	As a practical method ...
15	0	0	0.735	0	0.0417	76	4.59	Of course, LSA, as ...
16	0	0	0.853	0	0.0444	24	1.45	However, LSA as currently...
17	0	0	0.750	0	0.0421	98	5.92	LSA differs from other...
18	0	0	0.729	0	0.0416	105	6.34	However, as stated above...
19	0	0	0.874	0	0.0449	6	0.36	Preliminary Details about ...
20	0	0	0.842	0	0.0442	47	2.84	Latent Semantic Analysis is ...
21	0.05	0.163	0.822	0.134	0.2955	29	1.75	The first step is to ...
22	0.45	0.234	0.754	0.176	**0.3204**	62	3.74	**Next, LSA applies singular ...**
23	0	0	0.859	0	0.0445	17	1.02	Landauer, T. K., ...
24	0	0	0.867	0	0.0447	4	0.24	Basic and applied memory...
25	0	0	0.863	0	0.0446	15	0.90	Landauer, T. K., & Dumais...
26	0	0	0.182	0	0.0232	2	0.12	Psychological Review, ...

page, the maximum semantic similarity w.r.t. the ad text is calculated, and then the best location for advertising is discovered.

Now let us focus in a series of properties of the technique and let us observe its behavior through the results in the Table 1.

The technique is independent of the size of the text node analyzed. Initially, it is more straightforward computing calculations in simple documents than complete documents (or web pages). DOM nodes present the phenomenon of embedding. For instance, node `body` embeds all the paragraphs on a web page. Fragment 1 constitutes the biggest node in `http://lsa.colorado.edu/ whatis.html` and, nevertheless, it does not bring the highest `semsim`.

The technique offers maximum qualification for keyword coincidences. In column `semsim`, maximum values are those that present `relsim` different of 0, i.e., there is keyword shared in analyzed node and ad text. If there are not common keywords, the discriminating values will be those from `augsim`. And some interesting, we can obtain results inclusive when nodes do not contain common terms.

The technique exploits LSA results. `augsim` determines a nice measurement among text fragments. Each vector is enriched with meaningful relationships of terms. Hence `augsim` calculates the quality of correlations between terms. Let us observe the fragment number 5. Its text is short. However, the word `information` appears more than 40 times in `https://en.wikipedia.org/wiki/ Latent_semantic_analysis`, this is the reason why its measurements are higher in Table 1.

6 Related Work and Conclusions

To the best of our knowledge, there are many research works focused on measuring the success of advertising campaigns [2, 3]. However, there are not for measuring the best place for publicity location on a web page. It may be that this concern is more attractive for companies and, the solutions they reach are implemented instead of being published. However, from an academic point of view, it is interesting to measure the meaning of

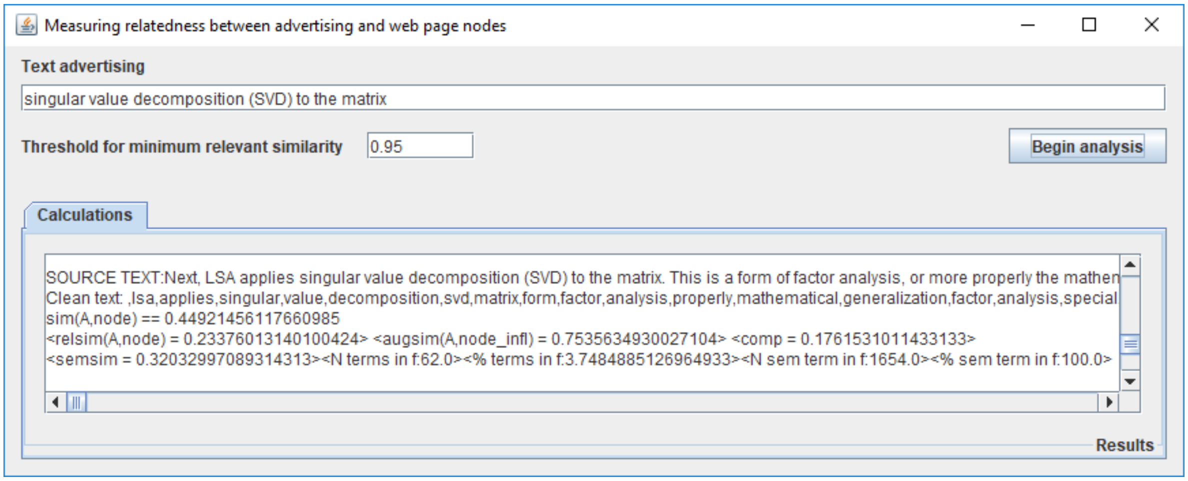

Fig. 5. A prototype for automatic relatedness calculation among advertising text and web page nodes

advertising w.r.t. the meaning of the web content which a user reads. In this work, we proposed a method for the numerical comparison. Other future work will be devoted to analyzing the success of this criterion for publicity location.

In summary, we have introduced a formal technique for semantic similarity between web page nodes and advertising messages, which presents several advantages: always returns a value, is independent of the size of text fragments, privileges (numerically) the existence of common words in text node and ad text, outperforms results of cosine similarity, only once calculation of LSA is required to produce any number of comparisons from an ad text.

The exposed formula of semantic similarity is based on that presented in [13], some formalizations are included here. And finally, the technique is described in a technological style (fully procedural), which is convenient to replicate the tool.

We have developed a pair of tools, one for web page (Jericho [6] based) parsing and other for text fragment ranking (Figure 5).

For future work, we are testing deep methods based on word embedding, to produce a parametric framework for web filtering and web indexing.

The potential applications of the technique are the following: analysis of similarity among content and advertising, filtering of web pages (since we rank DOM nodes with semantic similarity), the transformation of web pages, determining of hot sections in a web page, production of industrial tools, and other more.

Acknowledgements

This work was partially supported by Tecnológico Nacional de México / Instituto Tecnológico de Morelia.

We want to thank the University of Paris 13 for funding to attend this event and CEA for the support given.

Also we thank the IIES-UNAM professors: Atzimba López, Alberto Valencia and Hugo Zavala, who introduced us to the topic of Java libraries for HTML management.

References

1. **Castillo, C., Valero, H., Ramos, J. G., & Silva, J. (2012).** Information extraction from webpages based on DOM distances. *CICLing (2)*, pp. 181–193.
2. **Corner, M. D., Levine, B. N., Ismail, O., & Upreti, A. (2017).** Advertising-based measurement: A platform of 7 billion mobile devices. *Proceedings of the 23rd Annual International Conference on Mobile Computing and Networking*, MobiCom '17, ACM, pp. 435–447.

3. **Englehardt, S. & Narayanan, A. (2016).** Online tracking: A 1-million-site measurement and analysis. *Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security*, CCS '16, ACM, pp. 1388–1401.
4. **Furnas, G. W., Deerwester, S. C., Dumais, S. T., Landauer, T. K., Harshman, R. A., Streeter, L. A., & Lochbaum, K. E. (2017).** Information retrieval using a singular value decomposition model of latent semantic structure. *SIGIR Forum*, Vol. 51, No. 2, pp. 90–105.
5. **Gupta, S., Kaiser, G., Neistadt, D., & Grimm, P. (2003).** DOM-based content extraction of HTML documents. *Proceedings of the 12th International Conference on World Wide Web*, WWW '03, ACM, pp. 207–214.
6. **Jericho (2018).** Jericho HTML Parser. Available at: `http://jericho.htmlparser.net/docs/index.html/`.
7. **Landauer, T. & Dumais, S. (1997).** A solution to Plato's problem: The latent semantic analysis theory of acquisition, induction, and representation of knowledge. *Psychological review*, Vol. 104, No. 2, pp. 211–240.
8. **Landauer, T., Foltz, P., & Laham, D. (1998).** An introduction to latent semantic analysis. *Discourse processes*, Vol. 25, pp. 259–284.
9. **Le Hégaret, P. (2018).** W3C DOM IG: Document object model (DOM). Available at: `http://www.w3.org/DOM/`.
10. **López, S. & Silva, J. (2009).** A new information filtering method for webpages. *Database and Expert Systems Applications, DEXA, International Workshops, 2010*, pp. 32–36.
11. **Manning, C., Raghavan, P., & Schütze, H. (2008).** *An introduction to information retrieval*. Cambridge University Press.
12. **MathWorks & NIST (2013).** National institute of standards and technology, JAMA: A Java matrix package. `https://math.nist.gov/javanumerics/`.
13. **Ramos, J. G., Navarro-Alatorre, I., Flores Becerra, G., & Flores-Sanchez, O. (2019).** A formal technique for text summarization from web pages by using latent semantic analysis. *Research in Computing Science*, Vol. 148, No. 3, pp. 11–22.
14. **Ramos, J. G., Silva, J., Arroyo, G., & Solori, J. (2010).** A technique for information retrieval from microformatted websites. *Lecture Notes in Computer Science*, Vol. 5947/2010, pp. 344–351.
15. **Salton, G., Wong, A., & Yang, C. S. (1975).** A vector space model for automatic indexing. *Commun. ACM*, Vol. 18, No. 11, pp. 613–620.
16. **Silva, J. (2007).** A program slicing based method to filter XML/DTD documents. *SOFSEM (1)*, pp. 771–782.
17. **Silva, J. (2009).** Information filtering and information retrieval with the web filtering toolbar. *Electr. Notes Theor. Comput. Sci.*, Vol. 235, pp. 125–136.

Article received on 29/10/2019; accepted on 05/03/2020.
Corresponding author is J. Guadalupe Ramos Díaz.