Extending Web Content Management Systems Navigation Capabilities with Semantic Navigation Maps

Damiano Distante
Faculty of Economics
Unitelma Sapienza University
Rome, Italy
damiano.distante@unitelma.it

Michele Risi
Dept. of Maths and Computer Science
University of Salerno
Fisciano, Italy
mrisi@unisa.it

Giuseppe Scanniello
Dept. of Maths and Computer Science
University of Basilicata
Potenza, Italy
giuseppe.scanniello@unibas.it

Abstract— **This paper presents an automatic approach built on information retrieval and clustering techniques to enhance the navigation capabilities of modern Web Content Management Systems (WCMSs). The approach uses Latent Semantic Indexing to discover correlations between the contents published through these systems, and a fuzzy clustering algorithm to form groups of related contents. For each page of the developed website, a set of navigation links towards pages showing similar or related content and a measure of such similarity is proposed. An implementation of the approach for the Joomla! Open Source WCMS and the results from a case study on a real world website are also presented.**

Keywords- Web content management systems, Joomla!, navigation structure, semantic navigation maps, latent semantic indexing, clustering

I. INTRODUCTION

A Web Content Management System (WCMS) is a web system enabling its users to quickly develop a website and easily create, edit, and publish contents in it, with little training and little or no need for programming knowledge. Thanks to the significant benefits of using this kind of systems and to the availability of robust and reliable open source options for them, the number of websites relying on WCMSs is continuously increasing[1].

The list of features offered by a WCMS usually includes:

- Automatic generation of website navigation structure and content organization.
- Rich text WYSIWYG content editing.
- Templating, i.e., easily switching between different prebuilt or customized presentation themes.
- Security management, including session management and role-based access control to contents/operations.
- Modularity, enabling a WCMS to be extended with additional modules and plug-ins.

More advanced features of a WCMS may include: version control and archiving; granular editing privileges; search engine optimization; global and site-specific content sharing; automated content approval workflow; multilingual website management; page caching technology; user profiling and customization.

Focusing on navigation capabilities, visitors of websites relying on WCMSs are usually offered two possibilities to find contents of their interest: (*i*) perform keywords searches through a full-text search engine natively provided by the system or integrated in the website as an external service, such as Google Site Search[2]; (*ii*) surfing the website by using links and access structures forming its navigation structure.

Access structures automatically built and dynamically managed by WCMSs are usually of the following kinds [1]:

- Hierarchical navigation indexes enabling navigation to contents, based on their classification into categories and sub-categories.
- Breadcrumb trails showing the path to the current page all the way from the home page.
- Sitemap showing the list of pages forming the website in a hierarchical organization.

Additional access structures to support specific navigation goals have to be purposely created by the user of a WCMS. Examples of such navigation structures are represented by "guided tours" enabling the ordered navigation between the members of a selection of contents (aka, collection) sharing some property, and links connecting related contents.

The work presented in this paper is aimed at extending the above synthesized navigation capabilities provided by most WCMSs. To this aim, we propose an approach that extends the navigation structure of websites generated with WCMSs by means of Semantic Navigation Maps (SNMs) [2], a complimentary and automatically built navigation structure which enables navigating between the contents of the site, based on their similarity. The paper describes the approach with its underlying techniques, and an implementation for Joomla! (www.joomla.org), one of the most popular open source WCMSs. We also report on the results of a case study conducted on a real world website developed with this WCMS.

The paper is built on the work presented in [2] and extends the application of SNMs to the realm of websites generated and managed with WCSMs. In particular, compared to our previous work, this paper provides the following main new contributions:

[1] A rich list of commercial and open source WCMSs can be found in Wikipedia. A review for many of them can be found in CMS Critic (www.cmscritic.com), while surveys on popularity and market share trends can be found at CMSWire (www.cmswire.com) .

[2] http://www.google.com/sitesearch/

- A new approach based on the fuzzy c-means clustering algorithm to group related contents.
- An implementation as a module for the Joomla! WCMS which enables introducing SNMs into websites developed with this system.

The rest of the paper is organized as follows. Section II synthesizes our approach to group related contents and then to recover SNMs, its underlying techniques and describes the architecture of the implemented module that we have developed for the Joomla! WCMS. Section III highlights the results of a case study conducted on a real world website, while Section IV concludes the paper and draws possible future directions for our work.

II. The Proposed Approach and the Supporting Joomla! Module

A. An Overall Description

As discussed in Section I, navigation natively supported by most WCMSs is usually limited to category-based access to contents (often called "articles"). Additional navigation structures for specific information access goals have to be manually defined and kept up to date by the user. This is also due to the lack of support offered by these systems for well-known web design methods, such as those referenced in [3]. In particular, navigation between related contents belonging to different categories has to be supported with links expressly defined and kept up to date by the user.

Our work is intended to automatically discover relations between the articles published within a WCMS and to increment the navigation structure of the site with SNMs, i.e., with a set of links that connect each article to others which are found to be similar at content level. We have defined an approach to recover SNMs that has proven validity in a number of case studies involving real world websites [2]. On the other hand, we here propose a revised version of the approach and its implementation for integration with WCMSs.

The approach uses the LSI technique [5][6] to compute a similarity measure between the articles of the website, based on their textual content, and then it uses a fuzzy clustering algorithm [8] to identify clusters (groups) of articles with similar or related content. Links connecting a given article to others within the same cluster are then proposed to the user visiting the site as additional navigation paths. Each of the links is also accompanied by a measure of the similarity between the current article and the pointed ones.

The rationale for using a fuzzy clustering algorithm to group articles relies on the fact that an article could be associated to several concepts of the application domain, and for this reason it should belong to more clusters. The use of a graph-theoretic algorithm, as we did in [2], causes each article to belong to a maximum of one cluster, and thus to be associated with only one concept of the application domain.

In case new articles are added to the WCMS, the process incrementally performs the indexing, thus reducing the time needed to perform it. This has however a drawback as the navigation capability of the WCMS may be reduced. In fact, new terms of an inserted or modified article may not contribute to the enhancement of the existing latent-semantic space and then links among the new content and existing ones may be missed. The improvement in term of time required to incrementally insert a new document is 45-55%. In case the number of missed links becomes significant, the new latent-semantic space can be re-computed on all the contents within the WCMS. Conversely, the new latent-semantic space may be computed every week/month to update the complementary navigation structure of the WCMS according to the new added contents.

Figure 1 shows the UML Package Diagram of the logical architecture of the module that we have developed to implement the approach for Joomla!. As it might be clear, the architecture does not depend on the specific WCMS considered for our case study, and the module can be adapted to different WCMSs. The Data component represents the database of the WCMS. This component is also in charge of managing the persistence of the links between pairs of articles. The Computing Dissimilarity component computes similarities between the articles using an LSI technique. The software component in charge of computing similarities among articles is LSI Engine, which uses the library referenced in [12]. The component to group articles according to their similarity is Grouping Articles. This component implements the fuzzy clustering algorithm originally proposed by Kaufman and Rousseeuw in [8]. Finally, the GUI component is mainly in charge of starting the clustering-based process and enabling the user to manually refine (if required) the identified groups of articles through the Modifying Articles sub-component. This component also enables the selection of articles and websites to consider in the clustering. It is worth mentioning that the GUI component is integrated within Joomla!.

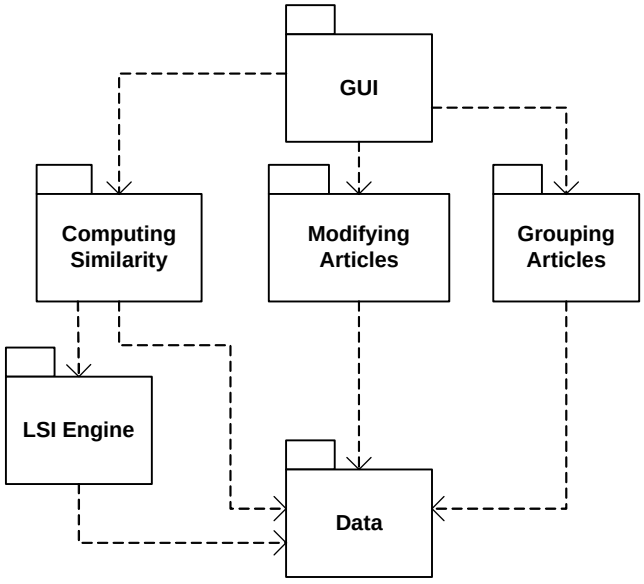

Figure 1. The software architecture.

B. The Recovery Process and the Underlying Techniques

Figure 2 depicts the overall process to recover links from a website developed with a WCMS, which is composed of two subsequent phases: Computing Similarity and Grouping Articles. The rounded rectangles represent process phases, whilst the rectangles represent the intermediate artifacts produced at the end of each phase.

The first phase is in charge of computing dissimilarity between pairs of articles present in the WCMS according to a properly defined measure. Successively, the phase Grouping Articles is performed to identify a suitable grouping of these articles. In the following we detail each of these phases and the underlying techniques.

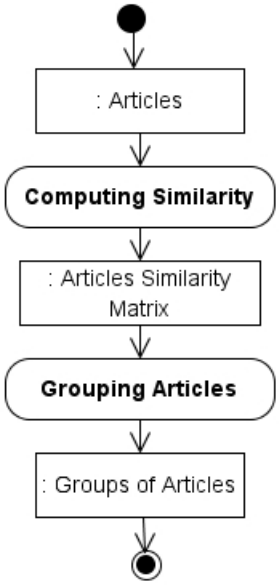

Figure 2. The overall process.

1) Computing Similarity

This phase first extracts the textual content of each article available in the WCMS. The content has then to undergo a normalization subphase in which non-textual tokens are eliminated (i.e., operators, special symbols, numbers, html tags, etc.), terms composed of two or more words are split (e.g., "mail_address" is turned into "mail" and "address") and terms with a length less than two characters are not considered. A stemming algorithm [9] is also used to reduce inflected or derived terms to their stem. The terms contained in a stop word list are removed as well.

The preprocessed text is then used to get the *A* term-by-document matrix. A generic entry $a_{i,j}$ of this matrix denotes the number of times that the i^{th} term in the j^{th} document appears. For the weight associated to each pair *<term, document>* we used the *term frequency–inverse document frequency*, also known as *tf-idf*, according to the chosen local (i.e. *lw*) and global (i.e. *gw*) weighting scheme, which is defined as:

$$tf\_idf(A) = lw\_\log tf(A) \cdot gw\_idf(A) \quad (1)$$

In particular for every term t_i and document d_j in A the term frequency–inverse document frequency is defined as:

$$tf\_idf(A[t_i, d_j]) = \log(A[t_i, d_j] + 1) \cdot \ln\left(\frac{|d|}{\sum_{i,j} A[t_i, d_j] > 0}\right) \quad (2)$$

The global weighting scheme in the previous equation is forwarded to the weighting step of the folding phase.

The normalized and weighted content is then used to compute the concept space by adopting an LSI technique that has been originally developed to overcome the synonymy and polysemy problem occurring with the Vector Space Model (VSM) [6]. In particular, LSI explicitly considers dependencies among terms and among documents (articles in our case), in addition to the associations between terms and documents. This technique assumes that there is a latent structure in word usage that may be partially obscured by the used words in a document.

LSI is applied on a term-by-document matrix A, which is built on the content of the articles. This matrix is $m \times n$, where m is the overall number of different terms appearing in the pages of the site and n is the number of articles. An entry $a_{i,j}$ of the term-by-document matrix *A* (with rank *r*) represents a measure of the weight of the *i*-th term in the *j*-th article. On this matrix a Singular Value Decomposition (SVD) [6] is applied to decompose it in the product of three matrices, $T \cdot S \cdot D^T$. The matrix *S* is an $r \times r$ diagonal matrix of singular values and *T* and *D* have orthogonal columns. SVD also provides a simple strategy for optimal approximate fit using a subset of *k* concepts (the space of the underlying concepts) corresponding to the largest singular values in *S*. The selection of a "good" value of *k* (i.e., the singular values of the dimensionality reduction of the concept space) is an open issue and a number of strategies have been proposed in the past (e.g., percentage of number of terms, fixed number of factors, etc.). In our approach, we calculate the number of singular values according to the Guttman-Kaiser criterion [7].

Terms and articles are graphically represented by vectors in the k space (i.e., Latent-Semantic Space) and the cosine between each pair of vectors indicates the article similarity. This value ranges from -1 (when the two articles are different) to 1 (when the content is the same). In our case we defined a cosine based dissimilarity measure to get a quantitative indication of how the content of the articles is different:

$$d(i, j) = \frac{1 - \cos(V_i, V_j)}{\max_{\forall V_x, V_y \in W}(1 - \cos(V_x, V_y))} \quad (3)$$

where *i* and *j* represent the articles and V_i and V_j the corresponding vectors in the k space *W* of a given website. This measure assumes values ranging from 0 (when the content of two articles is the same) to 1 (when they have a different content). Let us note that this measure cannot be considered a distance, as it does not obey the triangle inequality rule. However, this does not influence the possibility of using clustering algorithms as Oudshoff *et al.* shows in [10].

To incrementally add new articles (see Section II.A) and the corresponding contents to preexisting latent-semantic space, we used the fold-in method [12][4]. This method is used to avoid re-computing SVD each time a change is made to the term-by-document matrix. If on one side, the fold-in method can be computed very fast, on the other side, its accuracy may quickly degrade. In the latter case the term-by-document matrix has to be calculated considering all the documents to index. In fact, folding-in terms or documents is a much simpler alternative that uses an existing SVD to represent new information.

In case additional documents are to be folded into the existing latent-semantic space, again a new text-by-document matrix is constructed re-using the vocabulary of the first term-by-document matrix on the additional articles. Moreover, the resulting matrix *F* can be weighted re-using the global weights of the first matrix *A*. In particular the term frequency–inverse document frequency used to weight the new matrix is:

$$tf\_idf(F) = lw\_\log tf(F) \cdot gw\_idf(A) \quad (4)$$

The matrices *T*, *S* and *D* obtained by applying SVD can be reduced to *k*, thus resulting into the truncated matrices T_k, S_k and D_k. To fold-in a new document (i.e., a vector *d* whose dimension is $m \times 1$, where m is the number of terms of the term-by-document matrix *A*) into an existing latent-semantic space, a projection *d'*, of *d* onto the span of the current term vectors (columns of T_k) is computed by $d' = dT \cdot T_k \cdot S_k\text{-}1$.

Finally, the latent space of the new document will be added to the original latent space by computing $T_k \cdot S_k \cdot d'$. The interested reader can find further details in [4] and [11].

2) Grouping Articles

We adopted the fuzzy c-means clustering algorithm to group articles into *c* clusters. Similarly to the fuzzy logic, an article has a degree of belonging or membership to clusters (i.e., one or more clusters) rather than completely belonging to a single cluster. Compared to other fuzzy clustering methods, we used an extended version that accepts a dissimilarity matrix as input.

The membership (i.e., u_{iv}) of the article *i* to the cluster *v* is non negative. Moreover, the sum of all the memberships of a given article *i* is 1. The clustering is carried out through an iterative optimization of the objective function:

$$\sum_{v=1}^{c} \frac{\sum_{i=1}^{n}\sum_{j=1}^{n} u_{iv}^{r} u_{jv}^{r} d(i,j)}{2\sum_{j=1}^{n} u_{jv}^{r}} \quad (5)$$

where *n* is the number of articles and *i* and *j* are all the possible pairs that can be obtained on these articles. On the other hand, the number of cluster to identify is c (in our case is equal to $\lfloor n/2 \rfloor - 1$), while *r* is the membership exponent used in the fit criterion and it is larger than 1 (i.e., it assumes value between 1 and ∞). In case *r* is close to 1, the clustering algorithm behavior is comparable with the crisper version of this algorithm (i.e., the standard k-means clustering algorithm), whereas, for larger value of *r*, the fuzziness level of the clusters is higher. Finally, *d(i, j)* is the dissimilarity between the articles *i* and *j*. The iterations will stop when:

$$\max_{i,v=1..k} \left| u_{iv}^{(t+1)} - u_{iv}^{(t)} \right| < \varepsilon \quad (6)$$

The ε value ranges from 0 and 1 and represents a termination criterion, whereas *t* indicates the iteration step. It is worth mentioning that the tuning values of the adopted clustering algorithm should be chosen according to the content of the articles to be grouped.

The output of the iterations is a matrix of membership $u_{i=1..n,v=1..c}$. The clusters are defined using a threshold $th=1/c$ on the membership values. Successively, the algorithm groups the articles starting from each given article identifies a list containing articles with similar content. In particular, this list is computed through the union of the clusters to which the article participates.

The SNM associated to a particular article is built by linking the article to all the articles included in the clusters to which the given article belongs. Each link is associated with the measure of similarity calculated between the source and target articles.

C. The Joomla! Module

To support our approach in the context of WCMS, we have implemented an extension module for Joomla!, a well known and widely employed open source WCMS for publishing contents on the Web and intranets. Joomla! is written in PHP, uses MySQL as DBMS, and includes features such as page caching, RSS feeds, printable versions of pages, and support for language internationalization.

Joomla! easily allows modular extensions and integrations through plug-ins, modules, and components. Accordingly, new functionalities can be added to a website without hacking the core code of Joomla!. Among the extensions that Joomla! supports, modules complement the content contained in a page. This was the rationale for developing our proposal as a Joomla! module. In particular, to generate the SNMs using all the articles present in a website a graphical user interface enabling to change the properties of the module has been developed. SNMs can be also generated across all the websites hosted on a given server.

III. CASE STUDY

To assess the validity and feasibility of our proposal we experimented it on a real world website built using Joomla!. We chose for this purpose the website of "Balletto del Sud"[3], an Italian dancing company. At the date of the study, the site counted 339 articles which were all involved in the analysis.

In particular, the developed Joomla! module integrated in the bottom left-hand side of each page of the site the associated SNM recovered with the proposed process. Each SNM presents a set of links connecting the current article to other articles of the site that have been found similar to it. In addition, each link

[3] http://www.ballettodelsud.it/

reports the associated rank value obtained by computing the cosine between the current article and the pointed one.

TABLE I. DESCRIPTIVE STATISTICS OF THE ANALYZED WEBSITE.

Number of analyzed pages	339
Number of clusters	136
Number of iterations	185
Number of pages within the largest cluster	18
Number of pages within the smallest cluster	2
Number of clusters with one pages	32
Mean number of pages within the clusters	10.23

By analyzing the results of the conducted case study, we have observed that the 339 published articles were grouped into 136 different clusters, with the largest cluster containing 18 articles. On the other hand, the mean number of articles within the clusters was 10.23. Note that the number of iterations needed by the fuzzy c-means clustering algorithm to group the articles was 185. These and other descriptive statistics on the case study are summarized in Table I.

By examining the SNMs of a representative sample of articles of the analyzed website we observed that most of the proposed links featuring a similarity measure > 33% were in fact related to the main topic of the examined article. This threshold could be used to filter links to be shown by each SNM.

IV. CONCLUSION AND FUTURE WORK

In this paper we have proposed an approach to extend the navigation capabilities of current WCMSs (which usually natively support only category-based access to contents) with automatically built SNMs. We presented the approach, the underlying techniques, and an implementation for the Joomla! open source WCMS with a case study on a real world website developed with this system. The results of the case study have been not deeply presented and discussed for space reasons.

A SNM offers a user the possibility of navigating from each page to others with similar or related content. The approach relies on a process that first computes a dissimilarity measure between the articles published in the site using LSI, and then uses a fuzzy-clustering algorithm to group articles with similar or related content into clusters. The navigation structure of the website is then extended by including into each page the corresponding recovered SNM which shows links to related pages the user may be interested to visit and the associated measure of similarity.

SNMs are not intended to replace the navigation structure of a website, but to complement it in order to make explicit the correlations that might exist between the different contents published in the site.

Automatically recovered SNMs can be also useful to the editor of a website. In fact, s/he could use the retrieved SNMs to add explicit links between a given article and one or more found related to it with our approach. We aim to extend the proposal we have developed for Joomla! to include this feature.

Other future work will be devoted to quantitatively assess the correctness and completeness of the links provided by the recovered SNMs. Finally, we also plan to conduct empirical studies to evaluate the benefits and the impact on navigability of introducing SNMs in WCMSs, both in their front-ends and back-ends.

V. ACKNOWLEDGMENT

The authors would like to thank Lucio Santoro, who implemented some software components of the Joomla! module presented in this paper.

REFERENCES

[1] Cramer, R.: CMS Navigation Design - the architecture of dynamic content. http://www.ryancramer.com/journal/entries/cms-design-navigation/ (2008)

[2] Scanniello, G., Distante, D., and Risi, M.: An Approach and an Eclipse Based Environment for Enhancing the Navigation Structure of Web Sites. In Journal on Software Tools for Technology Transfer (STTT). vol. 11, no. 6, pp. 469-484. Springer Berlin / Heidelber (2009).

[3] Casteleyn, S., Daniel, F., Dolog, P., and Matera, M. (Eds.): Engineering Web Applications, ISBN: 978-3-540-92200-1, Springer Berlin Heidelberg (2009).

[4] Berry, M., Dumais, S., and O'Brien, G: Using Linear Algebra for Intelligent Information Retrieval. In SIAM Review, vol. 37, no. 4, pp. 573–595 (1995).

[5] Deerwester S., Dumais S. T., Furnas G. W., Landauer T. K., and Harshman R.: Indexing by Latent Semantic Analysis. In Journal of the American Society for Information Science, no. 41, 391-407 (1990).

[6] Harman. D.: Ranking Algorithms, In Information Retrieval: Data Structures and Algorithms, Prentice-Hall, Englewood Cliffs, NJ, 363–392, (1992).

[7] Guttman. L.: Some necessary conditions for common factor analysis. In Psychometrika, vol. 19, pp. 149-61 (1954).

[8] Kaufman, L. and Rousseeuw, P.J.: Finding Groups in Data: An Introduction to Cluster Analysis. Wiley, New York (1990).

[9] Manning, C.D., Raghavan, P., and Schutze, H.: Introduction to Information Retrieval. Cambridge University Press, (2008).

[10] Oudshoff, A.M., Bosloper, I.E., Klos, T. B., and Spaanenburg, L.: Knowledge Discovery in Virtual Community Texts: Clustering Virtual Communities, Journal of Intelligent and Fuzzy Systems, vol. 14, no. 1, pp. 13-24 (2003).

[11] Wang, X., Jin, X.: Understanding and Enhancing the Folding-In Method in Latent Semantic Indexing. In 17th International Conference on Database and Expert Systems Applications (DEXA 2006) Lecture Notes in Computer Science pp.104-113. Springer Berlin / Heidelberg (2006).

[12] Wild, F: An LSA package for R. In Proc. of the 1st International Conference on Latent Semantic Analysis in Technology Enhanced Learning, pp. 11-12 (2007).