Using Keyword Extraction for Web Site Clustering

Paolo Tonella, Filippo Ricca, Emanuele Pianta and Christian Girardi
ITC-irst
Centro per la Ricerca Scientifica e Tecnologica
38050 Povo (Trento), Italy
{tonella, ricca, pianta, cgirardi}@itc.it

Abstract

Reverse engineering techniques have the potential to support Web site understanding, by providing views that show the organization of a site and its navigational structure. However, representing each Web page as a node in the diagrams that are recovered from the source code of a Web site leads often to huge and unreadable graphs. Moreover, since the level of connectivity is typically high, the edges in such graphs make the overall result still less usable.

Clustering can be used to produce cohesive groups of pages that are displayed as a single node in reverse engineered diagrams. In this paper, we propose a clustering method based on the automatic extraction of the keywords of a Web page. The presence of common keywords is exploited to decide when it is appropriate to group pages together. A second usage of the keywords is in the automatic labeling of the recovered clusters of pages.

1 Introduction

Web sites often evolve from small and simple collections of purely HTML pages to big and complex applications, offering advanced transaction and data access facilities. The navigation structure is also subject to a similar trend. While initially only a few navigation facilities are needed, as the complexity grows, more advanced and intricate connections are provided. Lack of an established Web engineering practice is often the reason for a drift in the overall organization of the Web site. Cumulative maintenance interventions and successive radical changes may result in a “legacy” Web site, that can be hardly evolved in a safe and controlled way.

Tools and techniques are being developed to support understanding and restructuring of existing Web applications [4, 8, 11]. Software clustering [1, 6] aims at gathering software components into higher level groupings, thus providing the user with a more abstract, overall view of the system under analysis. It can be similarly adapted to the Web context, in order to produce a high level view of the Web site organization, in terms of cohesive groups (clusters) of pages and of the relationships among them. Such a view can be exploited in the process of Web site understanding, to gain knowledge about the organization of the entire site. More detailed information can be obtained by exploding each cluster of interest into subclusters, up to the individual pages that are the object of the change.

Web site clustering involves several non obvious decisions that profoundly affect the final result. First of all, the features used to describe each Web page have to be determined. These are the basic properties used to measure the similarity between pages, which in turn determines when two pages are clustered together. Several completely different choices are possible, ranging from the Web site connectivity [4], to the page structure [9], or to the content. The precise similarity measure, as well as the clustering algorithm in use, are other important parameters.

After computing the clusters for a Web site, a Web developer can access them to understand the overall organization of the system. However, to be meaningful, clusters should be properly labeled. A high level view showing just blank boxes (clusters) connected with each other, or, at the other extreme, labeled with all included pages, is by no means very informative and helpful. On the other side, a manual labeling process (concept assignment [4]) might be very difficult and time consuming. Some degree of automatic cluster labeling is thus another crucial feature for a practical usage of clustering in Web site evolution.

In this paper, we consider the page content as the basic feature used to cluster a Web site. Summary information about a page is obtained by means of keyword extraction. The same technique is exploited to attack the problem of cluster labeling. The keyword with the highest score within each cluster is used as cluster label. Preliminary experimental results confirm the feasibility of the approach.

The paper is organized as follows: the next section contrasts the existing literature with our proposal. Then, a brief summary on clustering methods is provided to make the paper self-contained (Section 3). In Section 4, the Natural Language Processing (NLP) method of keyword extraction is presented. The next section describes our clustering algorithm based on the keywords associated to each Web page, as well as our automatic cluster labeling technique. A case study is commented in Section 6. Conclusions and future work are given in the last section.

2 Related work

Clustering has several uses in program understanding and software reengineering [1, 6, 12], and has been recently applied to Web applications [4, 5, 9].

In [4] an approach to support the comprehension of Web applications by exploiting clustering techniques has been proposed. The approach is based on a conceptual model of a Web application and on a similarity measure between components that takes into account both the type and the topology of the links. This measure is exploited by a hierarchical clustering algorithm which produces a hierarchy of system partitions. Download and comprehension of the considered Web applications are conducted using the reverse engineering tool WARE.

Our approach introduces one major improvement over the technique described in [4]: automatic cluster labeling, and differs mainly in one respect: the basic feature exploited for clustering, which is the page content instead of its connectivity. In fact, content plays a very important role in Web applications and it can be hypothesized that it represents a good starting point for clustering. On the other side, the connectivity of a Web application suffers from a problem, highlighted also by the authors of [4]: purely navigational links (such as those leading to the home page) can hardly be distinguished from semantically richer ones. Labels in [4] (named concepts) are assigned to clusters manually. In our approach, labels assignment is handled in a completely automatic way.

In [5] an approach to identify duplicated pages (i.e., clones) in a Web application is proposed. Two different methods based on different similarity measures have been defined and experimented with: one exploiting the edit distance and the other one based on the frequency of the HTML tags in a page. The underlying descriptive feature (the HTML structure of a page) can be used as a further basic feature for clustering. This feature was actually exploited in [9], where an approach is presented for the identification of Web pages that can be migrated to a dynamic version, in that they share a similar structure.

3 Clustering

Clustering is a general technique aimed at gathering the entities that compose a system into cohesive groups (*clusters*). Given a system consisting of entities which are characterized by a vector of properties and are connected by mutual relationships, there are two main approaches to clustering [1]: the sibling link and the direct link approach. In the *sibling link* approach, entities are grouped together when they possess similar properties, while in the *direct link* approach they are grouped together when the mutual relationships form a highly interconnected sub-graph.

In the literature there exist several different clustering algorithms [12], with different properties. Hierarchical algorithms do not produce a single partition of the system. Their output is rather a tree, with the root consisting of one cluster enclosing all entities, and the leaves consisting of singleton clusters. At each intermediate level, a partition of the system is available, with the number of clusters increasing while moving downward in the tree. Divisive algorithms start from the whole system at the tree root, and then divide it into smaller clusters, attached as tree children. Alternatively, agglomerative algorithms start from singleton clusters and join them together incrementally.

3.1 Agglomerative hierarchical clustering

The agglomerative hierarchical clustering algorithm builds a hierarchy of clusterings starting from the bottom of the hierarchy, where each entity is in a different cluster. In each following step, the two most similar clusters are joined. After $N - 1$ steps (with N the number of entities), all entities are grouped into one cluster. Each level in the hierarchy defines a partition of clusters (i.e., a clustering). To select the resulting clustering, a *cut point* has to be determined.

We have adapted the agglomerative hierarchical algorithm known as Johnson's algorithm [12] to our purposes:

```
1. begin with $N$ clusters, each containing one Web page
   ($N$ is the number of Web pages in the Web site), and
   compute distance between pages (using the complement
   of the similarity measure).
2. **while** there are more than 1 cluster **do**
    (a) find the pair of clusters at least distance;
    (b) merge these clusters into a new cluster;
    (c) update the distance measures between each pair
        of clusters.
   **end while**
```

To update the distance measure between clusters, we have chosen the so called *complete linkage rule* [1]. This rule states that the distance measure between the already existing cluster C and a new cluster D, formed by joining clusters A and B, is the minimum between *dist*(C, A) and *dist*(C, B). It privileges cohesion over coupling [1].

4 Keyword extraction

To the aim of clustering Web pages having similar content, we need to characterize the content of a document in a way that is simple and computationally tractable. Also, the representation of the text content must be suitable for calculating efficiently content similarity measures between texts. A list of the most relevant keywords in the document can be used for this purpose. This approach is used for instance in [10].

The most simple approach to the extraction of the keywords of a text is based on finding the most frequent words in the text. The basic intuition underlying this approach is that the most important concepts in the texts are likely to be referred to repeatedly, or, at least, more frequently than minor concepts. Even if we think that this basic intuition is sensible, simply counting the most frequent words in a document is not enough to achieve our goal. The basic approach needs to be refined in a number of ways, that will be analyzed in the rest of this section.

Firstly, we need to consider lexical units that are wider than simple words. Not only can a concept be referred to with more than a word, that is a phrase, but often the concepts that characterize a text most precisely are the most specific ones, which are usually expressed with complex phrases. For this reason as a first attempt in this direction we considered as units both single words and bigrams, that is contiguous sequences of two words. Let us refer to one of these units (mono- or bigrams) as a term.

Secondly a term can occur frequently in a text without characterizing the text in contrast with other texts. This happens when all the texts that we are trying to cluster are about the same broad topic. For instance, if all the texts we are processing belong to the portal of a telephone company, we may find out that the terms ”telephone line” or, even worse, ”telephone” are not useful to characterize the meaning of a document even if they occur repeatedly. To avoid this problem the frequency of a term in a document must be confronted with the average frequency of that term in the whole Web site.

4.1 Bigrams selection

Not all bigrams in a text should be considered as terms to our purposes. We are mainly interested in three classes of complex terms: named entities, complex lexical units, and recurrent free phrases. Named entities are terms referring to individuals, locations, organizations, and dates. Complex lexical units are the kind of multi-word expressions that can be found in dictionaries (for instance phrasal verbs such as ”put on” and idiomatic expressions such as ”roller coaster”). Finally, the notion of recurrent free phrase was introduced by [2] to refer to a free combination of words which is recurrently used to refer to a concept. They are characterized by either high frequency in a reference corpus (e.g. ”American government”), high degree of association between words (e.g. ”first time”), or high salience (e.g. ”international summit”).

Selecting the bigrams belonging to each of these three classes is a challenging and resource demanding task. However the task can be appoximated by resorting to a combination of simple statistical measures and elementary linguistic knowledge. Our strategy consists in selecting as candidate keywords in a document the bigrams that where found in a list of frequent Italian bigrams. We built the list of frequent Italian bigrams with the following procedure: (1) select the topmost frequent bigrams in a reference corpus of 32 million words from an Italian newspaper; (2) cut off all bigrams occurring less than 4 times in the reference corpus; (3) apply a filter based on stop words. The aim of the filtering step is to get rid of bigrams like ”with the” or ”the only” which may occur frequently in texts but do not belong to any of the term classes mentioned above. To this extent we simply reject all bigrams including at least a word taken from a list of stop words. Note that stop words are in the great majority function words, that is words belonging to closed classes. Thus it makes sense to compile a list of stop words manually. The stop word list turned out to be very usuful to exclude frequent but irrelevant bigrams. Note that the words in the stopword list where excluded also from the topmost frequent monograms of the document.

4.2 Inverse document frequency

To better characterize each Web page to be clustered, we rank its keywords. A simple solution could be just counting the number of occurrences of the terms in the page (term frequency in Information Retrieval jargon). However this might not be an appropriate choice, as terms which occur uniformly in all the Web pages that we are clustering are not useful to distinguish documents with similar content as opposed to relatively unrelated documents. In alternative, we can rank the keywords on the basis of the inverse document frequency [7], defined as:

$$W[k] = N[k] \log \frac{D}{d[k]} \qquad (1)$$

where $N[k]$ is the absolute number of occurrences of the k-th keyword, $d[k]$ is the number of documents containing it, and D is the total number of documents.

High frequency keywords that are specific to a document ($d[k]$ small compared to D) have a high value of the inverse document frequency. On the contrary, unspecific keywords (i.e., keywords that occur uniformly in most documents) are given a small weight ($\log(D/d[k])$ is close to zero, since $d[k] \sim D$).

5 Web site clustering and labeling

Let KW be the vector of all keywords determined for a whole Web site (union of the keywords determined for each page), with each keyword uniquely represented by a single entry. A feature vector can be built for each page p, with $N[k]$, the number of occurrences of the k-th keyword $KW[k]$ in p, in position k. When a keyword is not present in a page p, the related entry in the feature vector is 0. Alternatively, the absolute number of occurrences $N[k]$ can be replaced by the inverse frequency in the document $W[k]$ (see equation (1)).

Given the description of each Web site page in terms of feature vectors, it is possible to exploit similarity or distance measures to agglomerate entities into clusters. Similarity/distance between clusters is generalized from the similarity/distance between entities by means of the complete linkage rule (see Section 3). In this work, we preferred a similarity measure over a distance measure, because the latter is prone to the well known problem of the sparse or empty vectors: distances become small not only when vectors are close to each other, but also when they are very sparse (or empty), thus leading to the formation of inappropriate clusters. The similarity measure used with the feature vectors described above is the normalized vector product, given by:

$$sim(p,q) = \frac{< V_p, V_q >}{||V_p|| \;\; ||V_q||} \qquad (2)$$

where V_p and V_q are the feature vectors of pages p and q respectively, angular brackets indicate the scalar product, which is normalized by the product of the norms, thus giving a similarity measure which ranges from 0 to 1.

After executing the agglomerative clustering algorithm, a proper cut point is manually selected. The possibility for the user to choose a given abstraction level (number of clusters or, equivalently, cut point), and then to adjust it toward the top of the hierarchy (less clusters with more pages inside) or toward the bottom (more clusters containing fewer pages) is an important interactive facility. In fact, the right abstraction level, appropriate for the ongoing program understanding task, is typically not known a priori, and can be determined empirically by moving upward and downward in the clustering hierarchy.

Then, labels are automatically assigned to each cluster. The keyword with the highest number of occurrences (resp., highest inverse document frequency) in a cluster C is assigned to C as its label. It should be noted that while the overall number of occurrences of a keyword $KW[k]$ in a cluster C is just the sum of the number of occurrences in each contained pages, the inverse document frequency is computed by applying equation (1) to entire clusters: D becomes the number of clusters, and $d[k]$ the number of clusters containing the keyword $KW[k]$.

6 Case Study

In order to illustrate the proposed technique, a small Web application (`www.promoturpejo.it`) has been analyzed. `Www.promoturpejo.it` is a bi-lingual (Italian and English) Web application that promotes the Pejo's Valley, a pleasant valley in Trentino (Italy). This dynamic Web application has been downloaded and analyzed by means of the tool **ReWeb** [8]. It consists of 240 HTML pages (57183 LOC), grouped into 9 directories and connected by 7107 hyperlinks.

The *graph representation* of a Web application (where nodes represent pages and edges represent hyperlinks among pages) can be abstracted by grouping pages according to the directory containing them. If no organization into directories is present, such a view (named *System View* in ReWeb), contains a single node representing the root directory (indicated with a dot). Otherwise, a node for each directory is added. An edge connects two nodes d_1 and d_2 of the System View if, in the graph representation of the site, there is a page in the directory associated to d_1 connected to a page in the directory of d_2.

Figure 1 shows the System View of `www.promoturpejo.it` as recovered by ReWeb. The proposed clustering technique has been applied to this Web site. The results of clustering have been assessed by measuring the distance from a reference grouping (*gold standard*) of the pages in the Web site, produced manually. The gold standard approach (expert criterion in [1]) is a general evaluation method that is used to measure the performance (for example, in terms of *precision/recall*) of an algorithm which approximates an "ideal solution" to a problem. The *gold standard* is such "ideal" solution to the problem.

To determine the *gold standard*, we have used the directory structure as a starting point, since in our example it is quite meaningful. Actually, two *gold standards* have been determined manually at different abstraction levels. One, labeled *High-level*, is composed of 13 groups, while the other (*Low-level*) is composed of 25 groups. The second is a specialization of the first, in that each group of pages in the first gold standard is possibly divided into subgroups. For example the pages contained in the directory `micologia` (mycology), a subdirectory of `estate` (summer) which forms a group in the *High-level* gold standard, are divided into two groups in the *Low-level* gold standard. One group contains mushroom cards, while the other group contains the pages that provide general mycology information.

The clustering algorithm has been applied to two different sets of vectors: 'promotour-vectors' and 'promotour-vectors-norm' (obtained by computing the document in-

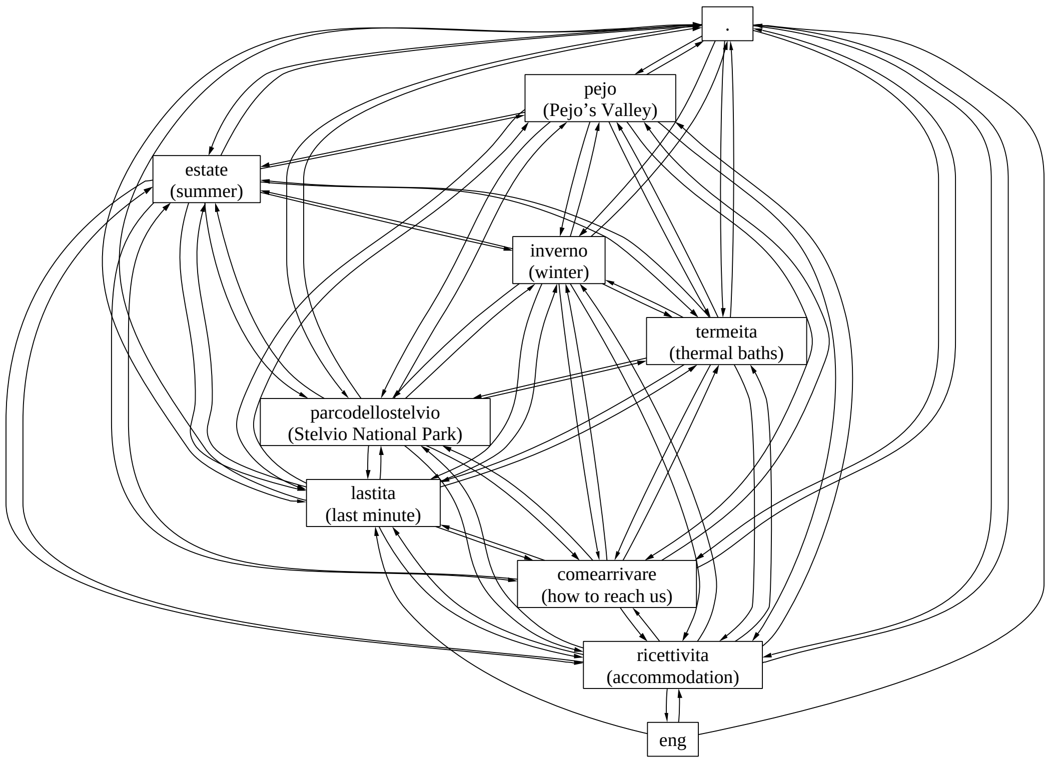

Figure 1. *System View of the Web application* `www.promoturpejo.it`*. The node with a dot represents the root directory. English translations of directory names are provided in brackets.*

verse frequency, see equation (1)). In both cases, 166 possible partitions, starting with a partition consisting of one cluster enclosing all pages, at the top of the hierarchy, down to a partition with singleton clusters only, at the bottom of the hierarchy, have been obtained.

By inspecting the steps of hierarchy formation, it is apparent that the technique works well. Pages of animals and pages promoting hotels/residences are respectively grouped together already at the cut point 20. At the cut point 60 (using 'promotour-vectors'), where 18 clusters are present, some other interesting groupings appear. In particular, the cluster of mushrooms cards, the cluster of hotels, the cluster of animals, and the cluster of flowers are emerging.

It can be noted that the clusterings most similar to the gold standard (both *Low-level* and *High-level* and using indifferently 'promotour-vectors' or 'promotour-vectors-norm') are those at a cut point between 115 and 150. With a cut point greater than 150, the algorithm groups too much, unifying clusters of pages with quite different contents. For example, the algorithm groups animals with countries at cut point 151, and sports with hotels at cut point 158.

For each possible partition, a measure of precision and recall with respect to the gold standard has been computed. Precision and recall are defined as in [1, 3], using the notion of *intra pair*. Intra pairs are pairs of pages in a same cluster. Precision and recall are defined by comparing the intra pairs in the gold standard and those in the clustering under test:

- **Precision**: Percentage of intra pairs in the test clustering that are also in the gold standard.
- **Recall**: Percentage of intra pairs in the gold standard that are also in the test clustering.

For each possible partition produced by our clustering method, we have plotted (recall, precision) couples at increasing cut points. Both *Low-level* (LL) and *High-level* (HL) gold standards have been considered (resp. Figure 2 and 3). The extremes of the curves are reached at the top (resp. bottom) of the clustering hierarchy. At one extreme, only one cluster contains all pages, so that all intra pairs of the gold standard are also intra pair in the only cluster (recall equal to 1), but many intra pairs in such cluster are not intra pairs in the gold standard (low precision). At the other extreme, the opposite is true: no error is in the singleton clusters, since they do not contain any intra pair (precision 1), but not correct intra pair is retrieved (recall 0).

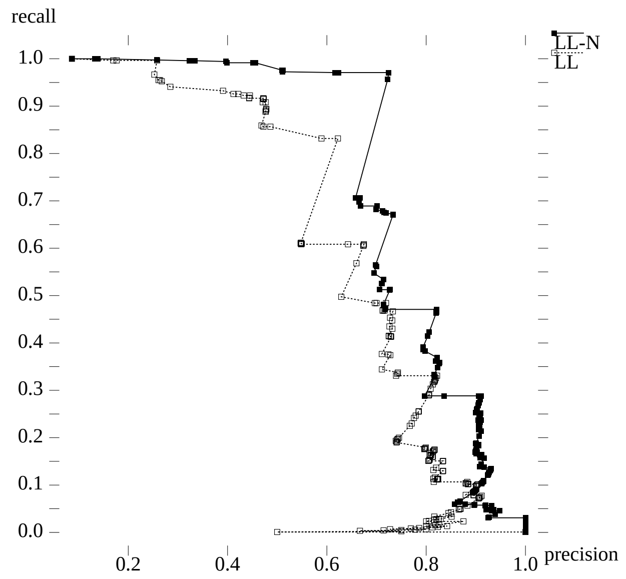

Figure 2. *Precision/recall at increasing cut points for the* Low-level *(LL) gold standard. Line LL is based on feature vectors with number of occurrences, while LL-N exploits the inverse document frequency.*

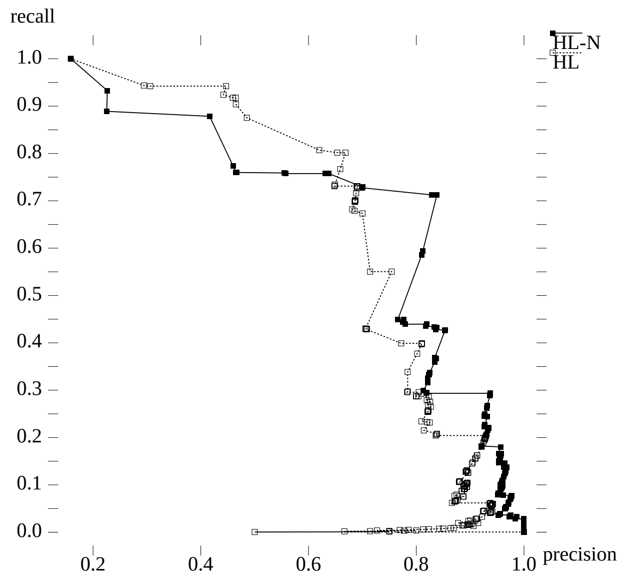

Figure 3. *Precision/recall at increasing cut points for the* High-level *(HL) gold standard. Line HL is based on feature vectors with number of occurrences, while HL-N exploits the inverse document frequency.*

Figure 2 contrasts the precision/recall plots obtained using the 'promotour-vectors' (dotted line) and the 'promotour-vectors-norm' (solid line), for the LL gold standard. It is quite clear that the clustering algorithm gives better results with input 'promotour-vectors-norm'. A good compromise of precision and recall in the LL-N curve is reached at the cut point 149 where the recall is 0.97 and precision is 0.72. This corresponds to the point near the top right corner of the figure.

Figure 3 is similar to Figure 2, but based on the HL gold standard. In this figure, the improvement deriving from the document inverse frequency computation is less evident. The HL-N curve remains under HL at high recall and low precision values (approximately, up to 0.75 of recall), while the trend is inverted afterwords.

Then, we have computed cluster labels for the clustering number 149, judged a good trade-off between precision and recall with respect to the LL gold standard, based on 'promotour-vectors-norm'. The result, restricted to the first 7 labels extracted and disregarding clusters composed of only one page, is shown in Table 1. In all cases, the cluster labeling algorithm reports labels that are related to the content of the respective clusters. In some cases (clusters 1, 3 and 9) the first automatically extracted label is the same as the label chosen by the expert. In other cases, the expert's label is present, but it is not at the top of the generated list (clusters 2, 4 and 8). In the remaining cases (clusters 5, 6 and 7), the expert's label does not appear directly in the list, but it is possible to find synonyms (cluster 6, *treatment*), or specific cases (cluster 7, *eurorafting, canyoning*) of general concepts (cluster 7, *water sports*).

Using the keyword with the highest score (the first in the list of extracted labels) as cluster label, it is possible to automatically produce the *Clustering View* in Figure 4. In the Clustering View each node represents a cluster and is labeled with the first label generated by the cluster labeling algorithm. An edge connects two nodes c_1 and c_2 of the clustering view if, in the complete view produced by ReWeb (Graph View), there is a page in the cluster associated to c_1 connected to a page in the cluster c_2. Not all labels produced automatically are completely satisfactory. A final manual refinement step is necessary to obtain a meaningful and usable view. However, even in the cases where an automatically produced label was changed, availability of an ordered list of automatically extracted labels was extremely useful in selecting the final label.

A comparison of the *Clustering View* in Figure 4 with the *System View* in Figure 1 (excluding nodes `.eng`, `lastita` and `comearrivare`) reveals that that there is a good agreement, although the former is more detailed than the latter. As noted above, the organization of this Web site into directories was judged a good and meaningful one, so that the ability of clustering to match it indicates that the algorithm is performing quite well. In other words, if the

Cluster	Expert's label	Automatic labeling (ordered)
1	appartamenti (apartments)	appartamenti prossimita' agenzia mt scheda cogolo situati (apartments proximity agency mt card *cogolo* located)
2	fauna (fauna)	settore fauna vive aquila mammiferi uccelli cervo (area fauna live eagle mammalians birds deer)
3	paesi, valle e tradizioni (towns, valley and traditions)	paese chiesa comasine cellentino strombiano valle celledizzo (town church *comasine cellentino strombiano* valley *celledizzo*)
4	hotels e inverno (hotels and winter)	sala camera servizi sci accesso apertura hotel (hall room services ski admittance opening hotel)
5	sport estivi e trekking (summer sports and trekking)	cima hotel rifugio sentiero malga scheda sport (peak hotel refuge path alpine-hut card sport)
6	terapie (therapies)	azione cura malattie acque proprieta' convenzioni specializzazione (action treatment illness waters property convention specialization)
7	sport d'acqua (water sports)	eurorafting discesa torrentismo idrospeed sport rafting ponte (eurorafting descent canyoning idrospeed sport rafting bridge)
8	funghi schede (mushrooms cards)	cappello gambo schede carne ricette funghi lamelle (cap-of-mushroom stem cards meat recipes mushrooms lamellae)
9	flora (flora)	flora fioritura fiori foglie primula specie semi (flora flowering flowers leaves primrose species seeds)

Table 1. *Labels extracted automatically for clustering 149. English translations are given under the corresponding Italian terms. Names of towns are in italic and not translated.*

same Web site had been organized as a flat collection of pages stored in one directory, clustering would be able to reconstruct a meaningful directory organization.

The node `estate` (summer) in the System View is decomposed, in the Clustering View, into three different clusters (corresponding to three subdirectories of `estate`), numbered 5, 8, and 9 in Table 1. Node `parcodellostelvio` is decomposed into two clusters: fauna (cluster 2) and flora (cluster 9), two subdirectories of `parcodellostelvio`. The node `pejo` corresponds to cluster 3 and `termeita` to cluster 6. In one case two nodes of the System View correspond to one cluster in the Clustering View: cluster 4 groups pages of the directories `inverno` and `ricettivita'`, although not all of the pages in the latter directory. Actually, Web pages promoting hotels and apartments (directory `ricettivita'`) advertise winter sports that are contained in the directory `inverno`. This is why they are inserted into a same cluster. Some of the pages associated with the node `ricettivita'` belong to cluster 1. These are the pages that deal with apartments, and are all contained in a subdirectory of `ricettivita'`.

7 Conclusions and future work

A preliminary exploration on the usage of Web clustering in support to program understanding has been conducted. A static Web site, containing various touristic information about a valley in the Alps, has been clustered, grouping together pages that are characterized by common keywords. Nodes in the resulting diagram (Clustering View) have been automatically labeled by the highest score keyword in each cluster.

A reference clustering (gold standard) was defined to assess the performances of our algorithm. In the clustering hierarchy, a cut point was selected giving the best compromise between precision and recall. The two performance values for such clustering are: precision = 0.72 and recall = 0.97. They are pretty high, thus indicating that the algorithm gets close to the expert's decomposition of the Web site.

Automatic labeling of the clusters gave also good results. In 3 cases the expert's label was the highest score keyword. In 3 cases it was among the keywords, although not the first one. In 3 cases it could be abstracted from the keywords. Although not always perfect, automatically extracted labels provide a remarkable support in interpreting the nodes of the Clustering View. While it is a very easy task to determine meaningful cluster names out of the keyword list, the same is not true if the names (or the entire contents) of the pages inserted into a cluster have to be inspected.

Our future work will be devoted to applying our method to more dynamic Web sites, trying to analyze the textual information that can be associated to comments and identifiers of script portions in dynamic Web pages. Moreover, we will consider the problem of contrasting the proposed clustering method against alternative ones.

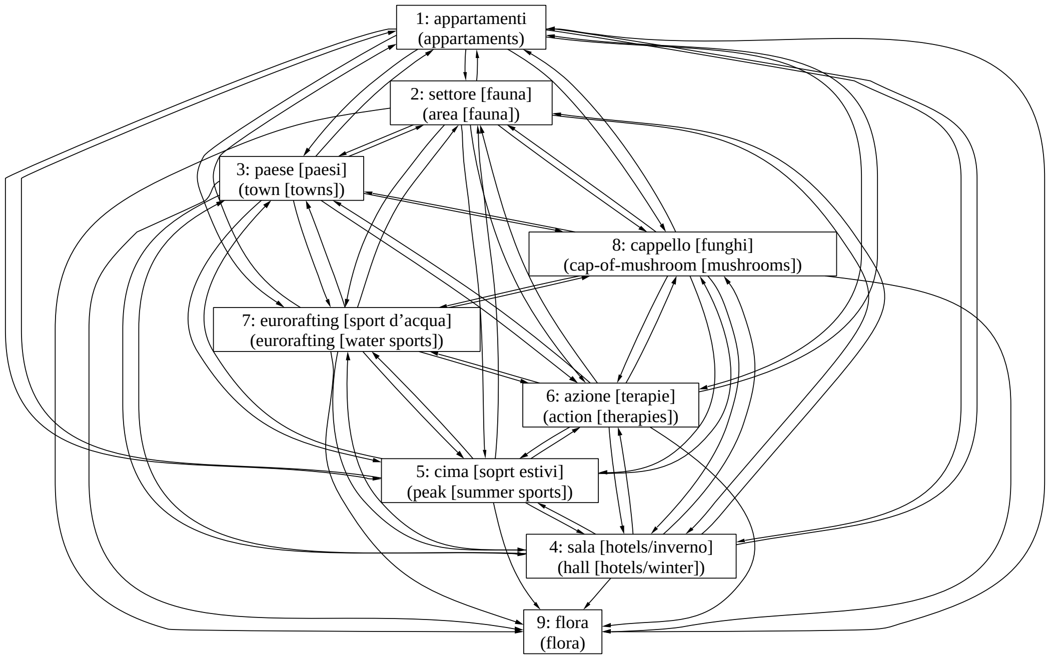
1: appartamenti (appartaments)
2: settore [fauna] (area [fauna])
3: paese [paesi] (town [towns])
8: cappello [funghi] (cap-of-mushroom [mushrooms])
7: eurorafting [sport d'acqua] (eurorafting [water sports])
6: azione [terapie] (action [therapies])
5: cima [soprt estivi] (peak [summer sports])
4: sala [hotels/inverno] (hall [hotels/winter])
9: flora (flora)

Figure 4. *The* Clustering View *of* Promoturpejo. *English translations are in brackets, manually refined labels in square brackets.*

References

[1] N. Anquetil and T. C. Lethbridge. Experiments with clustering as a software remodularization method. In *Proc. of the 6th Working Conference on Reverse Engineering (WCRE'99)*, pages 235–255, Atlanta, Georgia, USA, October 1999. IEEE Computer Society.

[2] L. Bentivogli and E. Pianta. Beyond lexical units: Enriching wordnets with phrasets. In *Proceedings of the Research Note Sessions of the 10th Conference of the European Chapter of the Association for Computational Linguistics (EACL'03)*, pages 67–70, Budapest, Hungary, April 2003.

[3] J. Davey and E. Burd. Evaluating the suitability of data clustering for software remodularization. In *Proc. of the Seventh Working Conference on Reverse Engineering (WCRE'00)*, pages 268–277, Brisbane, Australia, November 2000. IEEE Computer Society.

[4] G. A. D. Lucca, A. R. Fasolino, U. D. Carlini, F. Pace, and P. Tramontana. Comprehending web applications by a clustering based approach. In *Proc. of the 10th International Workshop on Program Comprehension (IWPC)*, pages 261–270, Paris, France, June 2002. IEEE Computer Society.

[5] G. A. D. Lucca, M. D. Penta, and A. R. Fasolino. An approach to identify duplicated web pages. In *Proc. of the 26th Annual International Computer Software and Applications Conference (COMPSAC)*, pages 481–486, Oxford, England, August 2002. IEEE Computer Society.

[6] S. Mancoridis, B. S. Mitchell, Y. Chen, and E. R. Gansner. Using automatic clustering to produce high-level system organizations of source code. In *Proc. of the International Workshop on Program Comprehension*, pages 45–52, Ischia, Italy, 1998.

[7] C. D. Manning and H. Schtze. *Foundations of Statistical Natural Language Processing*. The Mit Press, Cambridge MA, 1999.

[8] F. Ricca and P. Tonella. Analysis and testing of web applications. In *Proc. of ICSE 2001, International Conference on Software Engineering, Toronto, Ontario, Canada, May 12-19*, pages 25–34, 2001.

[9] F. Ricca and P. Tonella. Using clustering to support the migration from static to dynamic web pages. In *Proc. of the International Workshop on Program Comprehension (IWPC)*, pages 207–216, Portland, Oregon, USA, May 2003. IEEE Computer Society.

[10] J. F. Silva, J. Mexia, A. Coelho, and G. P. Lopes. Multilingual document clustering, cluster topic extraction and data transformation. *Lecture Notes in Artificial Intelligence (Progress in Artificial Intelligence)*, 2258:74–87, 2001.

[11] P. Warren, C. Boldyreff, and M. Munro. The evolution of websites. In *Proc. of the International Workshop on Program Comprehension*, pages 178–185, Pittsburgh, PA, USA, May 1999.

[12] T. Wiggerts. Using clustering algorithms in legacy systems remodularization. In *Proc. of the 4th Working Conference on Reverse Engineering (WCRE)*, pages 33–43. IEEE Computer Society, 1997.