A Cooperating Hybrid Neural-CBR Classifiers for Building On-line Communities

Maria Malek\(^1\), Rushed Kanawati\(^2\)

\(^1\)LAPI, EISTI, Av. du Parc E-95011 Cergy
maria.malek@eisti.fr
\(^2\)LIPN-CNRS UMR Q 7030
University of Paris 13,
99 AV. J.B. Clément
rushed.kanawati@lipn.univ-paris13.fr

Abstract

This paper reports on an ongoing development of an application that aims at identifying communities of use in the context of an organized group of people. The described application, called CoWing, consists of mining users' bookmark files to identify communities that share the same information interests. The system is composed of a set of assistant agents, called WINGS, and a central agent that manages the users organization. A Wing agent observes the user behavior to learn the user's bookmark classification strategy. A hybrid neural case-based reasoning incremental classifier is used for this purpose. Classification knowledge learned by an agent is then used to identify, for each folder in the local bookmark hierarchy, a community of users that share the same interests in the theme of that folder.

Introduction

One effective way to find relevant information is to locate people that are likely to know where such information can be found. Usenet news is one example of the application of this information searching strategy. Different World Wide Web (the web hereafter) searching systems apply the above-cited method. Different strategies are applied by the different systems in order to build and maintain communities of use. Authors of (Gibson, Kleinberg and Raghaven, 1998) propose a system that infers communities by mining the structure of the web. The ComMentor system, described in (Röscheisen, Mogensen and Winograd, 1995) provides a basis for community building by sharing annotations of web pages. The KnowledgePump system (Glance, Arregui and Dardenne, 1999) and Pharos (Bothors and Didieu, 1999) provide an infrastructure for defining communities. Both systems apply a collaborative filtering approach for supporting dynamic membership to predefined communities. Yenta (Foner, 1999) is a distributed matchmaking system where communities are managed by a multi-agent system.

In this paper we present a new system, called CoWing, for building and identifying on-line communities of use by mining users' bookmark repositories. Almost all web browsers available today provide users with some bookmarking facility. A bookmark system enables users to save addresses of web sites that the user needs to memorize. Typically a bookmark is a record that holds the following information: the web page address (the site URL), the page title and some other usage data such as the bookmark creation date, last visit date and user-provided page annotation text. In order to enhance the access time to the bookmark repository, users are provided with the facility of organizing their bookmarks in a hierarchy of folders.

The CoWING system provides an infrastructure that enables an organized group of users to share their bookmarks in an implicit way. By implicit, we mean that users are not required to do extra work to share their experiences. The only additional work is to define other's access rules to their own repositories. A role-based access control service is provided in order to ease this extra task. The basic idea of the system is the following: a personal agent, called a Wing observes the user behavior in managing her/his bookmark repository. Each Wing agent implements a hybrid neural/CBR classifier that learns how the user classifies her/his bookmarks. Wing agents can exchange parts of their own hierarchy of repositories. Each agent uses these informations to in order to establish relationship between each local folder and other’s (remote) bookmark folders. Two folders are related if they contain bookmarks treating the same information theme.

The reminder of the paper is organized as follows. The proposed system is studied in section 2. First the overall system architecture is described. The classification learning strategy, using the hybrid classifier is then
exposed. The section ends with detailing communities formation algorithm. Related work is briefly discussed in section 3. Finally we conclude in section 4.

The CoWing System

System overview

The COWING system provides a group of organized users with a computerized-support that enable users to share their experiences in managing bookmark lists. Figure 1 illustrates the overall architecture of the COWING system.

![Figure 1: The COWING system architecture involving three users: A, B and C.](image)

The system is composed of a central COWING agent and a WING agent per registered user. The COWING agent acts as WING agent registry. It provides WINGS with each other addresses. In addition it provides Wing agents with a description of the users organization. Each user manages her/his own hierarchy of bookmarks just as in single-user settings. However, users are required to set access rules that define which personal bookmarks or bookmark folders to share with whom. An easy to use role-based access control system is provided for that purpose. The access control model will not be detailed in this paper. Readers interested in this aspect can consult (Kanawati and Malek, 2001).

The community identification algorithm functions as follows: each Wing agent asks peer agents to feed him with new bookmarks. When a WING B receives a request from a WING A, the former computes the view A has on its own repository. An agent view is composed of the set of bookmark folders and bookmarks for which the agent has the read access right. The agent B sends back to A bookmark folders that constitute A's view on B's repository. For each received folder f, A uses its classifier, switched to the classification mode, in order to classify bookmarks contained in f. If the majority of these bookmarks are classified in a same local folder fi, then A recommends to add all bookmarks contained in f into fi.

When the user consults the bookmark folder fi s/he can confirm or reject the agent proposition. Depending on the user decision (i.e. confirm or reject) recommended bookmarks will be treated either as positive or as negative examples. Next, we introduce some notations that will be used in describing the functioning of the COWING System. Then we detail each of the three main services implemented in COWING: the access control service, the bookmark classifier and the bookmark recommendation mechanism.

Learning to Organize

Each WING agent uses a case-based reasoning (CBR) classifier in order to learn the user's bookmark classification strategy (Malek, 2000). A case is classically composed of two parts: the problem part and the solution part. The problem part contains the following indices:

1. Adresse: this is the page URL.
2. Content: this is a keyword vector that describes the bookmarked page content. Actually, this vector is taken to be the list of words defined in the meta section in the page HTML code.

Similarity over bookmarks is defined by an aggregation function of two simple similarity functions over the two indices. The similarity function between two URLs a and b is given by:

$$Sim(a,b) = 1 - \frac{h(a, MSCA(a, b)) + h(b, MSCA(a, b))}{h(a, root) + h(b, root)}$$

Where the function h() given the number of links between two nodes and the function MCSA() gives the most specific common abstraction of two nodes in the URL hierarchy.

The content similarity function is the following:

$$Sim_c(u,v) = \frac{Card(u \cap v)}{Card(u \cup v)}$$

Where card is the cardinality function.

The solution part is the folder identifier in which the bookmark is saved by the user.

The used classifier memory model, called PROBIS, is based on the integration of a prototype-based neural network and a flat memory devised into many groups, each of them is represented by a prototype. PROBIS contains two memory levels (see figure 2), the first level contains prototypes and the second one contains examples.
Atypical zone
Prototype level
Memory level

Fig. 2. The memory is composed of two levels: prototypes and stored examples.

The first memory level is composed of the hidden layer of the prototype-based neural network. A prototype is characterised by:

1. The prototype's co-ordinates in the $m$-dimensional space (each dimension corresponding to one parameter), these co-ordinates are the centre of the prototype.
2. The prototype's influence region, which is determined by the region of the space containing all the examples represented by this prototype.
3. The class to which belongs the prototype (i.e. a bookmark folder).

The second memory level is a simple flat memory in which examples are organised into different zones of similar examples.

These two levels are linked together, so that a memory zone is associated with each prototype. The memory zone contains all examples belonging to this prototype. A special memory zone is reserved for atypical examples. These are examples that do not belong to any prototype.

The classifier system operates either in learning mode or in classification mode. The system can switch from one mode to another at any moment. Before the first learning phase, the system contains neither prototypes nor zones of examples. Examples for training are placed initially in the atypical zone. Prototypes and associated zones are then automatically constructed. An incremental prototype-based neural network is used to construct the upper memory level. Particular and isolated examples are kept in the atypical zone whereas typical examples are transferred to the relevant typical zones. This memory organisation helps to accelerate the classification task as well as to increase the system's generalisation capabilities.

In addition, adding a new example is a simple task; the example is added in the appropriate memory zone and the associated prototype is modified. The learning procedure is the following:

1. If the new example does not belong to any of the existing prototypes, a new prototype is created (this operation is called assimilation). This operation is accomplished by adding a new hidden unit to the neural network. The co-ordinates of this prototype and the radius of the influence region is initialised to a maximal value (this is a system parameter). A new memory zone is also created and linked to the prototype. The new example is added to the new memory zone.
2. If the new example belongs to a prototype whose class value is the same as the example, the example is added to the associated zone of the second level memory. The prototype co-ordinates are modified according to the Grossberg learning law (Grossberg, 1987) to fit better the new example (this operation is called accommodation). The vector representing the prototype co-ordinates and memorised in the weights of the links going from the input layer to this prototype is modified according: $W_{prot}(t+1) = W_{prot}(t) + g(t) \cdot \text{Sim}(b_i - W_{prot}(t))$ where $b_i$ is the vector representing the bookmark to classify, $g(t)$ is a decreasing series which tends to 0, and $\text{Sim}$ is the bookmark similarity function.
3. If the new example belongs to a prototype whose class value is not the same as the example, the radius of this prototype is decreased in order to exclude the new example of this prototype (this operation is called differentiation). The new example is introduced again to the neural network and the most similar prototype (if there is any) is activated again and one of the three previous conditions is right.

Prototypes we obtain approximate the folders in the bookmark repository. Atypical examples correspond to bookmarks that can be classified in more than one folder.

**Building Communities**

The community formation consists on identifying remote folders that are similar to local ones. Bookmarks contained in remote similar folders will then recommended to the user to adding them to the local folder. The recommendation computation is performed as follows. Each WING agent maintains locally two data structures: an agenda and a folder correlation matrix (FCM). The agenda is a dictionary structure where keys represent identifiers of peer WING agents to contact and values are next contact dates. Hence Agenda[i][t] gives the next contact date with agent $i$. The FCM is a $m \times n$ matrix where $m$ is the number of folders in the local repository and $n$ the number of peer agents known to the local agent.
An entry $FCM[i,j]$ is a couple $<f',\text{cor}_i>$ where $f'$ is a folder identifier maintained by user $u_i$ and $\text{cor}_i$ is the correlation degree between the folder $f'$ and the folder $f_j$ maintained by local agent. Correlation between two folders $f_j$ and $f_i$ is given by the number of bookmarks contained in folder $f_i$ that are classified in folder $f_j$ divided by the total number of bookmarks in $f_j$. In the $FCM$ matrix, an entry $FCM[i,j]=<f',\text{cor}_i>$ is computed by taking the folder $f_i$ from the agent $j$ repository that have the maximum correlation value with folder $i$ belonging to the local repository. Given a WING agent, the booking recommendation process is made by executing the following algorithm:

For each B agent in Agenda do
If Agenda[B] is over then
    send B a bookmark request receive from B: V and ND
For each $f$ in V
    $<i,c>=\text{computeCorrelation}(f)$
    If $FCM[i,B].\text{cor} < c$ then
        $FCM[i,B]=<f,c>$
    If $FCM[i,B].\text{cor} > \delta$ then
        recommend to add bookmarks in $f$ to the local folder $i$

Figure 3 illustrates the interaction protocol between two Wing agents.

![Interaction protocol between WING agents](image)

The function $\text{computeCorrelation}$ (line 7 in above algorithm) finds the folder $i$ in the local repository that have the highest correlation value with a folder $f$ as defined above. The function proceeds as follows. For each bookmark $b_i$ in $f$ the local neural/CRB classifier is applied. For each bookmark, the classifier responds by the identifier of a local folder. The folder that has been selected the most will be the returned folder. Notice that the correlation relation is not symmetric since correlation is computed by using local classifiers (the classifier is different from one agent to another) and by using information contained in the local agent view on the repository of the other agent.

**Related Work**

Few systems are proposed in the literature to cope with the problem of collaborative bookmark management. Almost all-commercial systems are based on implementing a central shared URL repository that allows users to store and retrieve URLs. Some shared URL repositories, such as MyLynx.com allow user to define a private section and a public section.

Examples of shared bookmark systems are the GAB system (Wittenburg et. al. 1995) KnowledgePump (Glance, Arregui and Dardenne 1999). Pharos (Bouthor, 1998). The GAB system offers a service that allows merging different user bookmark repository in a virtual centralized bookmark. However no recommendation mechanism is implemented. It is up to the users to navigate in the merged repository to find bookmarks they are interested in. A comparable approach is also implemented in the PowerBookmarks systems (Li et. al. 1999). Both KnowledgePump and Pharos provide users with the possibility to share a centralized bookmark repository. The hierarchy of the repository is defined by the system administrator. Both systems provide also customization service in order to recommend users with bookmarks that are more interesting for them in given folder. Recommendation computation is made by applying a collaborative filtering mechanism that is base on matching the characteristics of bookmarks added and accessed by each user.

Most similar to our work is the RAAP system (Delgado, Ishii and Ura, 1998). In RAAP the system also learns by using a classical classifier how users classify bookmarks and use this information to recommend people with new bookmarks. However, RAAP has the disadvantage of being built on a centralized repository. It provides a poor access control model.

**Conclusion**

In this paper we have presented Cowing: a new full distributed collaborative bookmark management system. The COWING system addresses mainly the resources discovery problem. It provides a mean that allow users to share their bookmarks, in a personalized way without asking users to do extra task except for defining others access control on their own repositories. Each user is assisted by a personal agent, the Cowing agent that uses a hybrid neural/CRB classifier that learns the user strategy in classifying bookmarks. The learned classification strategy is used to construct associations between bookmark folders belonging to other users.

Experiments made on *synthetic data* show that our approach is valid. However, we believe that some enhancements should in order to make the system operational in real work settings. One important issue concerns the cold start problem (Kanawati and Malek, 2000). The applied recommendation computation
approach makes the hypothesis that users have organized their bookmarks in a hierarchy of folders. Each folder has some semantic sense. While lots of users do use hierarchical bookmark structures, some still using flat organization structures (Abrams 1997). Another related problem is the witnessed low overlapping between different user’s bookmark repositories (Cockburn and McKenzie 2000). We are working on proposing solutions to these two problems. Future work concerns also the extension of the system to handle the two other problems of bookmark maintenance and organization.

References


