Learning User Communities for Improving the Services of Information Providers

Georgios Paliouras,* Christos Papatheodorou,+ Vangelis Karkaletsis,*
Costantine Spyropoulos,* Victoria Malaveta+

* Institute of Informatics and Telecommunications,
+ Division of Applied Technologies,
National Centre for Scientific Research (NCSR) "Demokritos",
15310, Aghia Paraskevi, Attikis, GREECE
*Tel: +301-6503196, Fax: +301-6532175,
*E-mail:{paliourg, vangelis, costass}@iit.demokritos.gr
+ Tel:+301-6503287, E-mail:{papatheodor,victoria}@leucippus.nrcps.ariadne-t.gr

Abstract. In this paper we propose a methodology for organising the users of an information providing system into groups with common interests (communities). The communities are built using unsupervised learning techniques on data collected from the users (user models). We examine a system that filters news on the Internet, according to the interests of the registered users. Each user model contains the user's interests on the news categories covered by the information providing system. Two learning algorithms are evaluated: COBWEB and ITERATE. Our main concern is whether meaningful communities can be constructed. We specify a metric to decide which news categories are representative for each community. The construction of meaningful communities can be used for improving the structure of the information providing system as well as for suggesting extensions to individual user models. Encouraging results on a large data-set lead us to consider this work as a first step towards a method that can easily be integrated in a variety of information systems.

1 Introduction

The separation of "wheat from hay" on the Internet is becoming increasingly important as the amount of information available electronically becomes unmanageable for its non-expert receivers. As a result, a number of service providers have appeared recently in the market to help Internet users separate the information they need out of the plethora of information on the net. The simplest form of such a service is a search engine, which matches user-specified keywords to documents related to these keywords through word indices. The capabilities of such a search engine are restricted both in terms of the complexity of the search pattern and also in terms of the relevance of the information that gets retrieved. The overflow of information demands more advanced techniques. It is not the amount of information that gives the value, but access to the required information at the right time and in the most suitable form.

In this paper we examine the exploitation of user modelling techniques for the customisation of information to the needs of individual users. More specifically, we propose a methodology for organising the users of an information providing system into groups with common interests (*communities*). The communities are built from data collected from the users (*user models*). Each model contains the user's interests on the news categories covered by the information providing system. These interests are expressed by the user during his/her registration to the information system. The user model can be modified by the user at any point in time, reflecting changes of interest on the news categories.

The task of building user communities from user models can be seen as a data mining task, since we are looking for interesting patterns within a database, i.e., the user models. For this reason the type of method that is applicable to this problem is the method used mainly in data mining: *unsupervised learning*. In this paper we examine two unsupervised learning techniques, COBWEB [10] and ITERATE [3] and evaluate them on the task of building user communities. Both learning algorithms belong in the conceptual clustering family which is particularly suitable to symbolic training data, as it is the case here, where the users express their preferences on the news categories covered by the information system.

The resulting communities can be used to improve the services provided by the information system. However, this can be done effectively only when the generated communities are meaningful, that is if they contain those sets of preferences that are representative for the participating users. Ideally we would like to be able to construct a prototypical model for each community, which is representative of the participating user models and significantly different from the models of other communities. Such communities can be used to:

- re-organise the structure of the information system, e.g. re-define the hierarchy of news categories,
- suggest extensions to individual user models, e.g. news categories that are not selected by a user, but which are popular within his/her community,
- support the expansion strategy for the service, e.g. include new sources of information for popular news categories.

A sound and objective method for characterising communities is highly desirable, in order for the system designer to use the clustering results. This is a major problem in statistical clustering methods. That’s why we examine the use of a metric to decide which categories are representative for the community. This metric was used to evaluate the results of the application of the two learning algorithms in a large dataset of 1078 user models which was given to us by an Internet information provider.

Section 2 of the paper takes a broader view of user modelling in order to position our problem within this research domain, explains how machine learning techniques can be exploited in user modelling and describes the two learning algorithms and their applicability to our problem. Section 3 discusses the problem of constructing meaningful communities and introduces a metric for the characterisation of the generated community descriptions. Section 4 presents the setting of an experiment for the application of the two learning algorithms and discusses the experiment results. Section 5 presents briefly related work and section 6 describes ongoing work and introduces our plans for future work.

2 Research Background

2.1 User Modelling

User Modelling technology aims to make information systems really user-friendly, by adapting the behaviour of the system to the needs of the individual. The importance of adding this capability to information systems is proven by the variety of areas in which user modelling has already been applied: information retrieval, filtering and extraction systems, adaptive user interfaces, student modelling.

Information retrieval and filtering systems aim to deliver to the user those documents that are relevant to his/her information requirements, whereas information extraction systems aim to deliver specific facts from those documents. *NewT* is a system that helps the user filter Usenet Netnews [13] according to his interests. Brajnik and Tasso [5,6] present the exploitation of the user modelling shell UMT in the prototype information-providing *system Tourist Advisor (TA).* Kay [12] describes the *Movie Advisor* project, which operates on a database of movie descriptions, suggesting movies that should appeal to a specific user. *Doppelgänger* [15] is a user modelling system that gathers sensor data about users, makes inferences on those data and makes the results available to applications. *Firefly*’s agent software (see *http://www.firefly.com*) groups users preferences based on their similarities in order to suggest buying opportunities to specific customers based on their similarity to other customers. *Fab* [1] is a filtering system that recommends items similar to those a given user has liked in the past, and items liked by users whose interests are similar to those of a given user. *UMIE* [2] is a Web-based prototype user modelling component *(*see *http://www.iit.demokritos.gr/UMIE),* that filters the data extracted from an information extraction system according to the users models.

Adaptive user interfaces are implemented exploiting user models which contain information about the typical use of the system by each user. These user models differ from those in information filtering and retrieval, because they contain mainly procedural information, specifying how the system is used by each user. Substantial effort has been made to automate the adaptation of user interfaces to the user. This is usually achieved by monitoring the use of the system by the user, e.g. [7, 8, 12, 15].

This paper focuses on information retrieval and filtering systems, but the methodology that we propose is directly applicable to other systems that incorporate user models.

A user model consists mainly of the individual preferences of the user. Furthermore, it may contain personal information about the user, such as his/her age, occupation, etc. The latter type of information is not directly necessary for the adaptation of the system to the user, but may be used to categorise the user into a *stereotype*, which in turn allows the system to anticipate some of the user’s behaviour. Stereotypes have been introduced in [19], as a means of organising the users of the system into meaningful groups. For instance, a stereotype might state that “young people are interested in sports news.” Thus, a stereotype characterises groups of users, with common preferences. The characterisation of the group is based on personal information included in the models of the participating users.

Personal information about the users of a system is not always available and therefore the construction of user stereotypes may not be possible. In that case, the organisation of users into groups with common interests can still be useful. Such a group of users is termed a *user community* and corresponds to a stereotype missing personal information. Clearly, the loss of information in the transition from stereotypes to communities is not without cost. A stereotype can be used to predict the preferences of a user, even when he/she has not explicitly stated any of them. This is not possible with communities, which can only be used to extend/modify an existing user model. Despite that, user communities can be used in several ways to improve the quality of service provided by the information system.

2.2 Learning from user models

Machine learning methods have been applied to user modelling problems, mainly for acquiring models of individual users interacting with an information system, e.g. [4, 9, 17]. In such situations, the use of the system by an individual is monitored and the collected data are used to construct the model of the user, i.e., his/her individual requirements.

We are concerned with a higher level of generalisation of the users' interests: the construction of user communities. This task requires the application of learning techniques to user models, which are assumed to have been constructed by a separate process, either manual or automatic. Fig. 1 illustrates the two different levels of learning in user modelling. In the lower part of the graph, user models are constructed from individual user queries and at the higher level, the user models are used to build communities.

The choice of learning method depends largely on the type of training data that are available. The main distinction in machine learning research is between *supervised* and *unsupervised* learning methods. Supervised learning requires the training data to be preclassified. This usually means that each training item (*example*) is associated with a unique label, signifying the category in which the item belongs. In our case, this would mean that each user model must be associated with a category label out of a set of possible categories that have been defined beforehand. Given these data, the learning algorithm builds a characteristic description for each category, covering the examples of this category, i.e., the users belonging to the category, and only them, i.e., none of the users of other categories. The important feature of this approach is that the category descriptions are built conditional to the preclassification of the examples in the training set. In contrast, unsupervised learning methods do not require preclassification of the training examples. These methods form clusters of examples, which share common characteristics. When the cohesion of a cluster is high, i.e., the examples in it are very similar, it is labelled as a category.

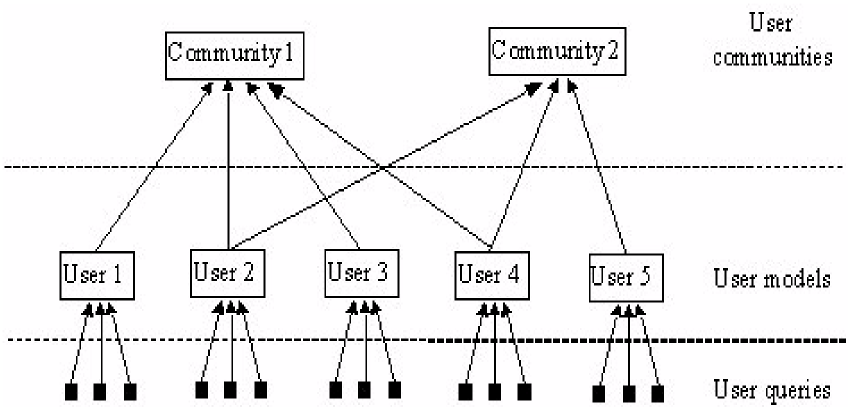

Fig. 1. Two levels of user-model construction

Supervised learning seems most appropriate for the construction of user stereotypes, i.e., the characteristic description of user groups based on personal information about the users. In contrast, the construction of user communities is a typical unsupervised learning task. In our work, we have chosen to concentrate on the construction of user communities, since the collection of personal information about an information system on the Internet is problematic. The construction of stereotypical descriptions for the communities, if personal information is available, is a straightforward task for the common supervised learning algorithms, e.g. C4.5 [16] and AQ15 [14].

Unsupervised learning tasks have been approached by a variety of methods, ranging from statistical clustering techniques to neural networks and symbolic machine learning. In this work, we have opted for the symbolic learning methods, because we are interested in the comprehensibility of the results. The branch of symbolic machine learning that deals with unsupervised learning is called *conceptual clustering* and two popular representatives of this approach are the algorithms COBWEB and ITERATE. These two algorithms are briefly described in the following subsection.

2.3 Learning algorithms

The two algorithms that we use in this work perform *conceptual clustering*. Conceptual clustering is a type of learning by observation that is particularly suitable for summarising and explaining data. *Summarisation* is achieved through the discovery of appropriate clusters, which involves determining useful subsets of an object set. In unsupervised learning, the object set is the set of training examples, i.e. each object is an example. *Explanation* involves *concept characterisation*, i.e., determining a useful concept description for each class.

COBWEB is an incremental algorithm that uses hill-climbing search to obtain a concept (cluster) hierarchy, partitioning the object space. The term incremental means that objects are incorporated into the concept structure as they are observed. We consider an object as a vector of feature-value pairs. In our case, objects are the user models and features are the news categories and take the values true and false for each user. Each concept in the hierarchy produced by COBWEB, is a probabilistic structure that summarises the objects classified under that concept. The probabilistic nature of concepts expresses the predictive associations between the feature and their values.

In order to construct the clusters, COBWEB uses *category utility* [11], which is a probabilistic measure of the usefulness of a cluster.

In the following we denote the j-th value of a feature A_i by V_{ij} and the k-th cluster by C_k. Category utility is a trade-off between the conditional probabilities of intra-cluster similarity, $P(A_i=V_{ij}|C_k)$, and inter-cluster dissimilarity, $P(C_k|A_i=V_{ij})$. Formally the category utility for a cluster k is defined as:

$$CU_k = P(C_k)\left[{}_{i}\ {}_{j}\ P(A_i = V_{ij}|C_k)^2 - {}_{i}\ {}_{j}\ P(A_i = V_{ij})^2 \right] \qquad (1)$$

and for the whole set of objects:

$$CU = \frac{{}^{n}_{k=1}\ \mathrm{CU}_k}{n}, \qquad (2)$$

where n is the number of clusters in a partition.

CU represents the increase in the expected number of feature values that can be correctly guessed:

$$P(C_k)\ {}_{i}\ {}_{j}\ P(A_i = V_{ij} \mid C_k)^2 \qquad (3)$$

given a partition (C_1,C_2,...,C_n), over the expected number of correct guesses with no such knowledge:

$${}_{i}\ {}_{j}\ P(A_i = V_{ij})^2 \qquad (4)$$

COBWEB incorporates objects into the concept hierarchy using the following four clustering operators:

- placing the object in an existing cluster,
- creating a new cluster,
- combining two clusters into a new one (merging) and
- dividing a cluster (splitting).

Given a new object, the algorithm applies each of the previous operators and selects the hierarchy that maximises category utility. COBWEB is an efficient and flexible algorithm. The complexity of the incorporation of an object in a hierarchy is quadratic to the nodes of the derived hierarchy. Since the size of the hierarchy is log-linearly related to the size of the object set, COBWEB is scalable to large training sets. The algorithm can also deal with missing feature values. However, COBWEB depends on its incremental character, i.e. it is dependent on the order of the observed objects.

In ITERATE the problem of the order-dependence is dealt by an object ordering strategy called ADO (Anchored Dissimilarity Ordering). ITERATE is not incremental and does not produce a concept hierarchy, but a flat partition of the objects. It starts by generating a tree, which is then flattened into a set of cohesive and maximally distinct clusters. However ITERATE is a descendant of COBWEB and inherits several of its features, e.g. the probabilistic structure of the concepts, and the category utility measure.

ITERATE has three primary steps:

- classification tree generation from a set of objects,
- partition of the tree to extract a set of clusters and
- iterative redistribution of the objects among the clusters to achieve a new set of maximally separable clusters.

During the first step, ITERATE produces a classification tree using the category utility measure similarly to COBWEB. The differences between the two methods are the following:

- ITERATE does not utilise the merging and splitting operators of COBWEB and
- if a node of the tree contains one or more objects, it sorts them using ADO scheme in order to obtain a maximally dissimilar ordering among the objects.

In the second step, ITERATE extracts an initial partition structure by comparing the category utility of the clusters, CU_k along a path in the classification tree. For every path from the root to a leaf of the tree, the algorithm computes the category utility of each cluster. While this measure increases, the algorithm goes on to the next cluster of the path. At the cluster for which the measure value drops, the algorithm inserts its previous cluster to the initial partition list. In this manner ITERATE ensures that no concept subsumes another and maximises cluster cohesion.

In the final step, ITERATE redistributes the objects to the clusters of the initial partition (derived in step 2), until the *category match* measure is maximised for each object. During this process, some clusters may be merged and thus the initial partition is improved. The category match metric measures the increase in the expected predictability of cluster C_k for the feature values present in data object *d*. Formally category match is defined as:

$$CM_{dk} = P(C_k) \underset{i,j\in\{A_i\}_d}{} \left(P(A_i = V_{ij} | C_k)^2 - P(A_i = V_{ij})^2\right) \qquad (5)$$

The category utility measure of a given cluster, is a sum of the category match measures of the objects in the cluster.

3 Construction of meaningful communities

The clusters generated by the application of the two learning algorithms to the data collected from the users (user models), represent the user communities. The question that arises is whether there is any meaning in the derived communities. Since there is no personal information available about the users, the construction of stereotypical descriptions for the communities is not possible. The only information available is the set of preferences of the users in each community. Thus, the natural way to construct meaningful communities is by trying to identify sets of preferences that are representative for the participating users. Ideally we would like to be able to construct a prototypical model for each community, which is representative of the participating user models and significantly different from the models of other communities.

The construction of prototypical models for the communities is a problem in itself. We specify a metric to decide which are the representative news categories for each community of news readers. This metric measures the increase in the frequency of a category within a community, as compared to the default frequency of the category in the whole data set. The frequency of a category corresponds to the proportion of users that are interested in this category. In [3] the increase in frequency was used as an indication of the increase in the predictability of a feature within the community. Given a category *c*, with default frequency f_c, if the frequency of this category within a community *i* is f_i, the metric is defined as a simple difference of the squares of the two frequencies:

$$FI_c = f_i^2 - f_c^2 \qquad (\mathbf{6})$$

FI stands here for Frequency Increase. Clearly, when FI_c is negative there is a decrease in frequency and the corresponding category is not representative for the community. The definition of a representative news category for a community, is simply that $FI_c > \alpha$, where α is the required extent of frequency increase. If $\alpha>0$ then the requirement is that the frequency of a category within a community has increased, in comparison to the frequency in the initial dataset. The question that arises is how large the increase should be in order for the category to be considered as a characteristic one for a community.

In order to see the impact of the parameter on the characterisation of the communities, we propose to vary α and measure the following two properties of the generated community descriptions:

- *Coverage:* the proportion of news categories covered by the descriptions. Some of the categories will not be covered, because their frequency will not have increased sufficiently.
- *Overlap:* the amount of overlap between the constructed descriptions. This is measured simply as the ratio between the total number of categories in the description and the number of distinct categories that are covered.

In the following section, we evaluate the results of the two learning algorithms in a case study for the construction of meaningful communities. More specifically, we measure the coverage and overlap properties of the generated community descriptions, varying α values.

4 Case Study

4.1 Experimental setting

We applied the two learning algorithms on the task of constructing user communities for a news-filtering system. This system collects information from various sources in the Internet and forwards it to its users, according to their user models. The news articles are organised by the information provider into 24 news categories, e.g. sports, computers, etc., which are further divided into 239 subcategories, e.g. sports/football, sports/volleyball, etc. During his/her registration, each user specifies a subset of these categories, which correspond to his/her personal interests. This personal list of news categories constitutes the user model, which determines what news he/she receives. The user model can be modified by the user at any point in time, reflecting changes of interest.

The dataset for the experiment contained 1078 user models, with an average of 5.4 news categories and 17.4 subcategories specified in each model. These user models formed a set of training examples for the learning algorithms. Each example was represented as a binary feature vector, which specified which news categories the user was interested in. In other words, each bit in the vector corresponded to a category and was "on" if the user was interested for this category and "off" otherwise. Given these training examples, the two algorithms constructed groups of users with common interests.

The two-layered organisation of news categories in the system suggested two different test cases: one using the 24 general categories and one using their subcategories. Unfortunately, the ITERATE algorithm could not handle the size of the dataset, when considering the 239 subcategories.[1] For this reason, we only examined the first of the two test cases, the results of which are presented in the following section.

4.2 Results

In the first experiment, the COBWEB algorithm was applied on the small data set, i.e., the one with the 24 news categories. The generated concept hierarchy consisted of 699 nodes, of which the first three layers are presented in Fig. 2. An important property of the tree in Fig. 2 is the balanced split of objects in different branches. The nodes in the same level of the tree cover subsets of the data set that are of comparable size. Therefore the underlying concepts are of similar strength.

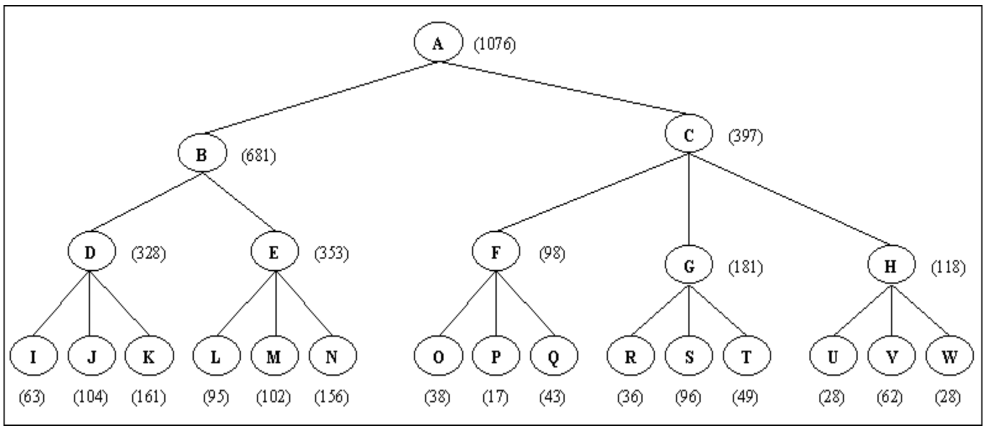

Fig. 2. The top three layers of the concept hierarchy produced by COBWEB. Node A is the root of the hierarchy, covering the whole dataset. The numbers in brackets correspond to the number of objects covered at each node.

We chose to examine only the top three levels of the tree generated by COBWEB, because the number of clusters generated by ITERATE is comparable to the second and third levels of the COBWEB tree. ITERATE generates 11 classes in the second step, i.e., as the initial partition of the object space. The final partition, which is constructed by ITERATE contains only two clusters and is therefore of limited use.

ITERATE's partition of the data set into the 11 clusters is as follows:
C1 (34), C2 (88), C3 (167), C4 (5), C5 (129), C6 (247), C7 (122), C8 (62), C9 (67), C10 (59), C11 (98),

[1] We are currently examining whether this is due to the design of the algorithm or its implementation. If the former case is true, ITERATE is not scalable to training sets with a large number of descriptive features.

where the number in brackets corresponds to the number of objects in each cluster. The majority of clusters are of comparable size, with the exception of a very small one, C4, and four large clusters: C3, C5, C6 and C7.

As mentioned in section 3, we use the FI_c measure in (6) in order to construct meaningful communities from the generated clusters.

Ideally, we would like to acquire descriptions that are maximally distinct, i.e., minimise the overlap, and increase coverage. Fig. 3 and Fig. 4 show how the two measures vary for different sizes of α, for COBWEB and ITERATE. In the case of COBWEB, we examined two different partitions of the objects, corresponding to the second and the third level of the concept hierarchy, i.e., nodes {**D, E, F, G, H**} and {**I, J, K, L, M, N, O, P, Q, R, S, T, U, V, W**}.

Fig. 3 shows that the coverage of the two algorithms varies in a similar manner as the value of α increases. As expected, the larger the value of α the fewer the categories that are covered by the clusters. In the case of COBWEB, the coverage on the second level of the hierarchy is consistently lower than that on the third level. The reason for this is that the concepts on the third level are more specialised than those on the second. The frequency of the categories in the more specific clusters, is higher than in the more general ones.

Fig. 4 depicts the amount of overlap between the community descriptions for the two algorithms. It becomes clear from those results that the larger the number of the communities, the larger the overlap between their descriptions. Thus, there is a trade-off between coverage and overlap, as the number of communities increases. In the case of the 15 different clusters, the extent of the overlap is significantly higher than in the other two cases. The 5-cluster case in COBWEB gives the best results in terms of the distinctness in the descriptions.

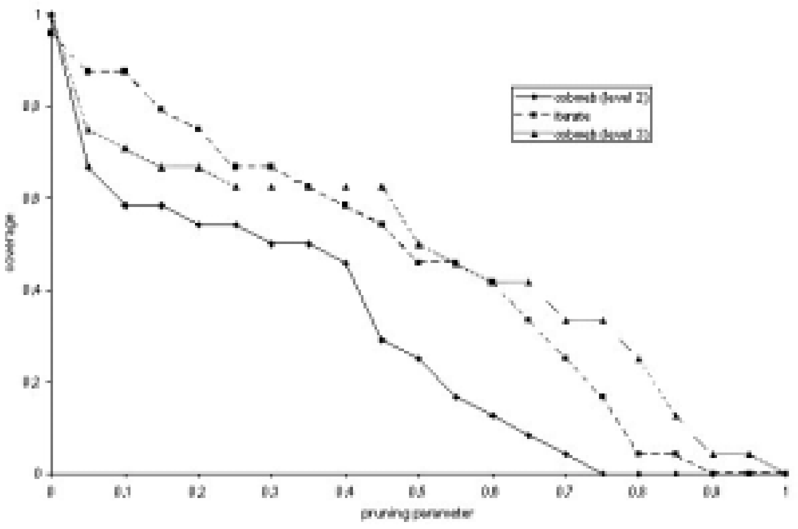

Fig. 3. Results on the coverage of news categories for the two algorithms. On the x axis, the value of the pruning parameter α is changed from 0 to 1 at an increment of 0.05.

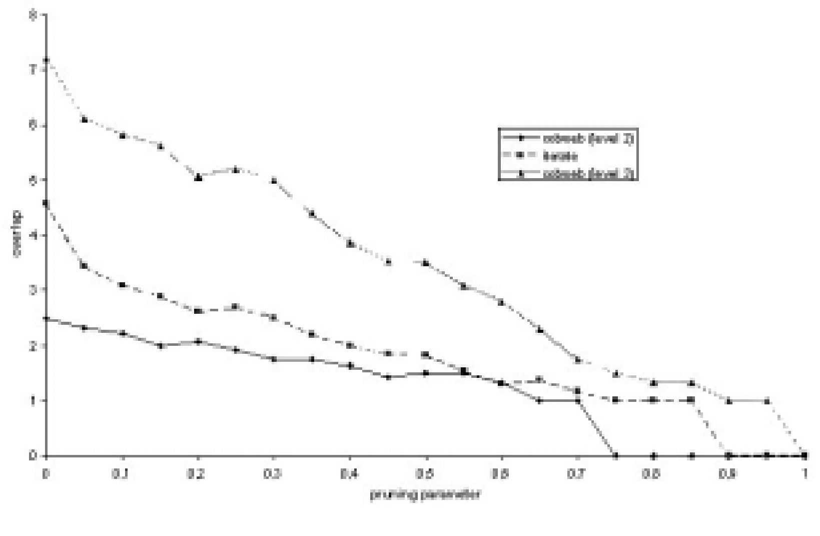

Fig. 4. Results on the overlap between community descriptions as α varies.

In the final step of our work we looked at the actual community descriptions in the manageable case of the 5 clusters generated by COBWEB. For α=0.5, a set of very concise and meaningful concept descriptions were acquired. Table 1 lists the news categories for the 5 clusters, together with their FI_c values.

Communities **E**, **G** and **H** are well-separated, corresponding to a group of people interested in the Internet, another who are interested in Economics and a third interested in computers. Group **F**, consists of people interested mainly in economics and finance, but also in computers. Some interest in computers is to be expected from the users of a system on the Internet. Finally, cluster **D** serves as a "miscellaneous" category and does not seem to be homogeneous. The main similarity between the people in **D**, is that they are very specific in their choices, resulting in user models that are empty, i.e., most categories are not of interest to the user.

Table 1. Descriptions for the 5 categories generated on the second level of the concept hierarchy

D	**E**	**F**	**G**	**H**
	Internet (0.55)	Economic Indicators (0.73)	Economic Indicators (0.58)	Computers (0.53)
		Economics & Finance (0.68)	Economics & Finance (0.61)	
		Computers (0.6)		
		Transport (0.53)		
		Financial Indicators (0.5)		

An obvious problem with the descriptions in Table 1 is that a large number of news categories are not covered. In general, these are the categories that are chosen by either too few or too many users. In the former case the algorithms choose to ignore them during learning and in the latter case, they correspond to such general interests, that they are difficult to attribute them to particular communities. Filtering out these two types of category is a positive feature of the FI metric. Coverage can increase, by moving selectively at lower levels of the COBWEB hierarchy. For instance, the children of node **H** give meaningful and concise communities, corresponding to categories that are related to computing, e.g. electronics, networks, telecommunications. However, this is not the case for all of the five communities in Table 1. For instance, the children of node **E** do not provide further meaningful subgroups in the community. The ability to select nodes in different levels of the concept hierarchy is an important advantage of the COBWEB algorithm. Arguably, this selection is done automatically by the ITERATE algorithm, using the category utility metric. However, in our case ITERATE did not provide meaningful and concise communities.

5 Related work

This paper presents a methodology for the exploitation of user modelling technology in information providing services. There are two main approaches: community and stereotype modelling. The choice depends mainly on the type of information that is available. Community modelling does not require system-independent information. This is important, because such information is often not available, e.g. in WWW applications it is impossible to have personal information about all people who access a site. The lack of system-independent information, in our case study, led us to follow the community modelling approach.

However, there are cases where system-independent variables are also included in a community-based system. This is the case in [15] where both types of variables are treated in the same manner by the system. In our case, the communities include only system-dependent variables, since the registered users do not provide any personal information either explicitly or implicitly. We have to note that this is the case with most information providing services.

Although community modelling has the advantage that it does not require system-independent information, it should be stressed that communities cannot replace stereotypes. No suggestions can be made about a new user, unless he/she explicitly states some of his/her interests. The work presented in [1] pays particular attention to that fact and distinguishes between two types of community-based system. In the first type the user specifies some of his interests, whereas in the second the system builds the user model gradually, since there is no initial information about the user. The news filtering system used in our experiment belongs to the first type, since the user during his/her registration has to specify the interesting news categories. The latter type of system cannot classify a new user into any of the communities and can therefore not provide any user-specific behaviour.

Concerning the construction of user communities there are two main approaches. In the first one the user models themselves are used to reason about the interests of a new user. For instance, we could search for an old user B who shares most of his interests with the interests of the new user A and then use the user model of B to suggest extensions to the model of A. This type of reasoning is called *case-based* and does not involve any learning, in the sense of drawing general rules from the data. The algorithm that is mostly used for this type of reasoning is called k-nearest-neighbour. The information systems that are based on this approach are called *recommender or collaborative filtering systems* [18].

The second approach for constructing user communities is to perform some kind of clustering, using either a standard statistical method, such as a hierarchical clustering algorithm or a neural network. Although clustering seems a computationally expensive task compared to the case-based approach, this is not a real problem since communities change far less often than individuals. On the other hand, clustering is a necessary task according to the objectives of the work presented in this paper. As we mentioned in the introduction, the resulting communities can be used to improve the services provided by the information system. However, this can be done effectively, only when the generated communities are meaningful. That's why we examined the use of a metric (i.e. the Frequency Indicator) to decide which categories are representative for the community. Although the clustering approach is used for the construction of communities [15], this is the first time, as far as we know, that the clustering algorithm results are evaluated in order to generate meaningful community descriptions.

6 Conclusions

In this paper we have presented a methodology for constructing user communities. These communities can be used to improve the exploitation of an information system by its users. The construction of the communities was achieved by elaborating unsupervised learning techniques. In particular, we used two conceptual clustering algorithms: COBWEB and ITERATE. Our main concern is whether meaningful communities can be constructed. This is a major problem in statistical clustering methods, the results of which are usually hard to interpret. A sound and objective method for solving this problem is highly desirable, in order for the clustering results to be of practical use for the service provider. This issue forms the motivation of our research work. We examined the use of a metric in order to decide which categories are representative for a community. We evaluated the results of the two learning algorithms, based on that metric, in a case study.

Our results are very encouraging, showing that useful information can be extracted about the use of an information system, through the characterisations of the generated community descriptions. We consider this work as a first step towards a method that can easily be integrated in a variety of information systems, in order to provide the required insight into the use of the system.

A further direction of interest is the use of machine learning to construct user stereotypes, when personal information is available. The conceptual relationship between stereotypes and communities suggests that some of the work presented in this paper will also be of use there. However, as mentioned above, learning user stereotypes is primarily a supervised learning task, due to the assumed dependence between user groups and their personal characteristics.

Concluding, we believe that the task of constructing user communities for information systems is of great practical use and it is also a technically challenging matter. Despite the encouraging results presented in this paper, we would like to evaluate our methodology on other systems as well. So, we have already started to validate the proposed methodology, in the recently established information providing system of the NCSR "Demokritos" Library.

Acknowledgements

We would like to thank Prof. Biswas for supplying the ITERATE code and the GMD machine learning group for their public version of COBWEB. We are also grateful to the Internet service provider, who prefers to remain anonymous, for the large dataset we used in our experiments.

References

1. Balabanovic, M. and Shoham, Y.: Content-Based, Collaborative Recommendation. Communications of the ACM 4 (1997) n. 3 66-72
2. Benaki, E., Karkaletsis, V. and Spyropoulos, C. D.: Integrating User Modelling Into Information Extraction: The UMIE Prototype. Proceedings of the User Modelling conference UM'97 (1997) 55-57
3. Biswas, G., Weinberg, J. B. and Fisher, D.: ITERATE: A Conceptual Clustering Algorithm for Data Mining. IEEE Transactions on Systems, Man and Cybernetics 28 (1998) 100-111

4. Bloedorn, E., Mani, I. and MacMillan, T. R.: Machine Learning of User Profiles: Representational Issues. Proceedings of the National Conference on Artificial Intelligence (AAAI) (1996) 433-438
5. Brajnik, G. and Tasso, C.: A Shell for Developing Non-monotonic User Modeling Systems. International Journal of Human-Computer Studies 40 (1994) 31-62
6. Brajnik, G., Guida G. and Tasso, C.: User Modelling in Intelligent Information Retrieval. Information Processing and Management 23 (1987) 305-320
7. Brusilovsky, P., Schwarz, E.: User as Student: Towards an Adaptive Interface for Advanced Web Applications. Proceedings of the User Modelling conference UM'97 (1997) 177-188
8. Chin, D.N.: KNOME: modelling what the user knows. In: Kobsa, Wahster (eds): User models in dialog systems. Springer-Verlag, Berlin (1989) 74-107
9. Chiu, P.: Using C4.5 as an Induction Engine for Agent Modelling: An experiment of Optimisation. Proceedings of the User Modelling conference UM'97 (1997) Workshop on Machine Learning for User Modelling
10. Fisher, D. H.: Knowledge Acquisition via Incremental Conceptual Clustering. Machine Learning 2 (1987) 139-172
11. Gluck, M. A. and Corter, J. E.: Information, Uncertainty and the Utility of Categories. Proceedings of the Seventh Annual Conference of the Cognitive Science Society. Lawrence Erlbaum Associates (1985) 283-287
12. Kay, J.: The um Toolkit for Cooperative User Modelling. User Modeling and User Adapted Interaction, 4 (1995) 149-196
13. Maes, P.: Agents that Reduce Work and Information Overload. Communications of the ACM 37 (1994) n. 7 31-40
14. Michalski, R. S., Mozetic, I., Hong, J. and Lavrac, N.: The Multi-Purpose Incremental Learning System AQ15 and its Testing Application to Three Medical Domains. Proceedings of the National Conference on Artificial Intelligence (AAAI) (1986) 1041-1045
15. Orwant, J.: Heterogeneous Learning in the Doppelgänger User Modeling System. User Modeling and User-Adapted Interaction 4 (1995) 107-130
16. Quinlan, J. R.: C4.5: Programs for Machine Learning. Kaufmann (1993)
17. Raskutti, B. and Beitz, A.: Acquiring User Preferences for Information Filtering in Interactive Multi-Media Services. Proceedings of the Pacific Rim International Conference on Artificial Intelligence (1996) 47-58
18. Resnick, P. and Varian, H.R.: Recommender Systems. Communications of the ACM 4 (1997) n. 3 56-58
19. Rich, E.: Users are Individuals: Individualizing User Models. International Journal of Man-Machine Studies 18 (1983) 199-214