A MULTI AGENT APPROACH TO INTEREST PROFILING OF USERS

P.H.H. Rongen[1], J. Schröder[1], F.P.M. Dignum[2], J. Moorman[2]

[1]IBM Nederland - Center for Advanced Studies
{erik, jasper}@nl.ibm.com
[2]Institute of Information and Computing Sciences
Utrecht University
{dignum, jmoorman}@cs.uu.nl

Abstract. Intelligent applications deliver personalized experiences and services to the user. This is done by creating and using a profile of the user: user profiling. Several approaches and algorithms are developed for user profiling. This paper describes a multi agent approach that allows multiple algorithms to be combined dynamically to generate a knowledge and interest profile of a user. IBM's ABLE environment was used for the implementation of the multi agent system. To test the system, the user interest profile is build on browse behavior and this profile is applied in a TV program recommender system. The results of implemented system show that multi agent systems provide an excellent platform for an extendible user profiling system that can use multiple classifiers.

1 Introduction

The amount of channels that provide us with information increases rapidly. These channels get more and more available to us at any time and any place. In the last ten years the clear distinction between places and activities has faded away. As a result daily life is becoming more and more complex, despite the increasing capabilities of intelligence in applications and services [SC04]. Intelligent applications provide personalized services and experiences, based on user profiles. A user profile is a collection of attributes that belong to an individual. This includes address books in mobile phones, browser history information, and messages, emails and documents authored by the user. This data can be processed to obtain additional user profile information. Examples are the extraction of a user's social network from address books, the deduction of user interests from the browser history and the extraction of expertise from messages and documents written by the user.

User profiles can be created explicit or implicit by the subject or can be created by others. When signing up to a web site like hotmail, a user has to fill out a form containing amongst others name and address. This is defined as the explicit creation of a user profile. Implicit creation of a user profile happens automatically during other activities. Examples of implicit user profiling are Amazon that keeps an interest user profile based on bought books and your computer keeping track of visited web pages.

Doctors create medical records for their patients which is a sample of explicit profile creation by others.

Explicit creation of a user profile by a user has a number of disadvantages amongst which that users get annoyed at them and thus do not fill in correct and/or complete information. In the other hand, implicitly build profiles can be very incomplete.

Besides the distinction of explicitly and implicitly build profiles we can also distinguish profiles on the basis of whether they are structured or unstructured. Many implicitly build profiles are unstructured. I.e. they are not predefined in any way and will either be defined ad hoc during the profiling process, or remain undefined while the profile data is just stored in raw form. Unstructured profilers are often used by systems that use the profile directly in local applications. A good example of such a system is WebMate[CH97]. WebMate is a personal assistant that produces a personalized news paper by observing multiple news sources and selecting the news articles that match with the current user profile.

A good example of a structured profiler is the Quickstep system [MI04] that acts as a recommender system for a group of researchers within a computer science laboratory. The Quickstep system monitors the browsing behavior of a specific researcher by recording the visited web pages, classifying them onto an existing ontology of topics.

A major drawback of unstructured user profiles is that it is difficult to reuse them in different systems or to compare them. In addition structured profiles can more easily be visualized, reviewed and modified by a user.

The different user profiling systems use different approaches and techniques for the internal representation of the user profile. Samples of these techniques are the vector space model, n-grams, semantic networks, associative networks, and classifiers including neural networks, decision trees, inducted rules and Bayesian networks [MO02]

In situations where multiple techniques and profilers compete with each other to be the best and perform differently in different environments, the best solution is to combine multiple techniques and profilers into one system. This is possible through the use of a multi agent system, which also provides the possibility to distribute the application over multiple systems.

In the remainder of this paper, a flexible multi agent system approach is presented that is able to create and manage the profile of a user dynamically. The profiling system is split up into several autonomous entities that all take care of a specific part of the profile and combine their results in an overall user profile. The agents collaborate in producing an appropriate profile and compete with each other in delivering the best results.

This multi-agent system is tested with the personal television guide created by the Telematica Institute [TE04]. The multi-agent system builds the user profile on browse behavior and the profile is used by the television recommender to create viewing advises.

2 Multi Agent System architecture

The profiling system presented here is able to generate a profile from the data that is obtained by observing user behavior. This data can for instance be the number of web pages the user has visited the past hour, the e-mail the user received today or the transcripts from a number of chats. Both unstructured and structured approaches to the classification of profiling data are considered. The general design of the presented profiling system is based on a structured approach. It consists of a structured hierarchy of profiling agents, each of which is responsible for a sub topic of the total profile. At the top level of this hierarchy, a single agent called the root, can be found. The root agent observes the behavior of a certain user and sends this information through to agents that are connected to it. When no agents are connected yet or the current set of agents is unable to profile the current data, the root will try to connect to other agents by requesting them from an agent pool. Whenever the root obtains data, it will request new agents from the agent pool to profile this new data.

The newly connected agents themselves may request other agents to create more specific profiling data. Figure 1 shows the profiling hierarchy after a number of new agents has been connected to the previously connected ones.

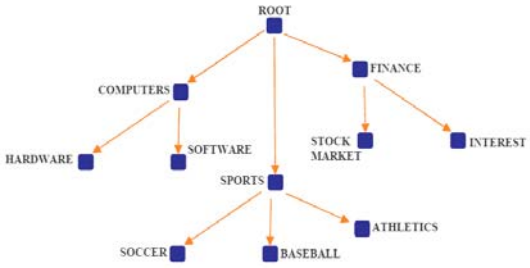

Fig. 1. An iteratively extended profiling hierarchy

An important reason to use a hierarchical agent structure is that generally, a classifier performs better within a limited domain. Classifying algorithms often locate and use the most distinguishing set of elements from the data to be able to determine the right class. In limited domains it is much easier to locate these distinguishing elements. A soccer agent for instance, can assume the incoming information is classified as sports, since it is placed beneath the sports agents within the hierarchy. Such a soccer agent therefore only has to find the distinguishing factors of soccer compared to other sports and doesn't have to worry about distinguishing the data from other topics that don't concern sports.

Another reason for using a hierarchical agent structure is performance. If all agents are connected directly to the root, the system will become increasingly slow, when the number of agents grows, since the root will have to pass the new information to all other agents, and they all have to process it.

The current approach doesn't enforce the agents to use a specific structure for building the hierarchy. The hierarchy is initiated with a root and whatever will be added to this hierarchy is up to the agents. In the current design approach, the agent pool is responsible for defining the structure of the hierarchy. Whenever an agent of

the profiling hierarchy requests a new node agent from the agent pool, a specialized agent called the pool keeper will determine which agents will be advised to the requester and therewith the structure of the hierarchy

2.1 The profiling agents

The profiling agents can decide autonomously whether the incoming information can be profiled as belonging to their designated topic, or not. To increase the flexibility of the classification, different algorithms can be applied to different domains and different algorithms can be combined to produce a weighted average. This is achieved by connecting a profiling agent to multiple classifiers. These classifiers are specialized in classifying documents on a certain topic with a certain algorithm. A number of classifiers are connected to the profiling agent. The profiling agent forwards the information it retrieves from super-nodes to the classifiers and combines their results to determine the outcome of the profiling process.

Whenever the profiling process is successful, indicating that the incoming information could be classified according to the designated topic of the profiling agent, the agent will forward the data to its connected profiling agents or, if the current agents are unable to process it, will contact the pool keeper to obtain new profiling agents for the new data.

Figure 2 shows the resulting model for the multi agent profiling system. The model indicates the different types of agents and their relationships. The profile root can be connected to multiple pool keepers and a pool keeper can be connected to multiple profiling roots, for instance a web profiling root and a mail profiling root. The profile root is connected to at least one pool keeper, since the profile root needs to obtain new agents from at least one agent pool. Besides the pool keeper each profile root is also connected to the profile portal of the centralized profile. This connection is needed for obtaining the appropriate profile keepers that assemble the profile data from the several profile hierarchies.

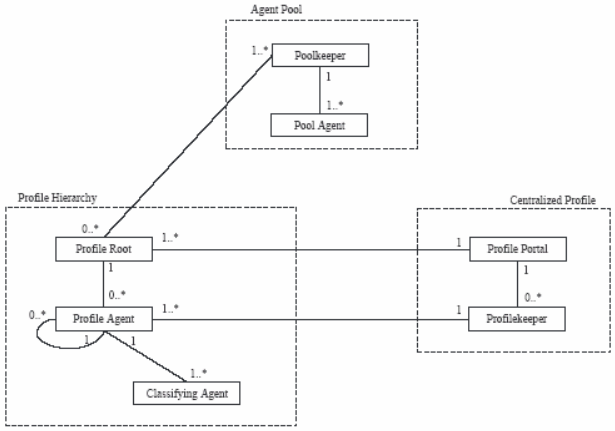

Fig. 2. The multi agent profiling system model

Furthermore the profiling agents are connected to a set of classifiers, or at least one, that perform the classification tasks on the specified topic in the specified domain, each using their own algorithm. The agent pool consists of the pool keeper and a number of agents of any type.

The centralized profile is the portal to the outside world. It contains a profile portal, an agent that is connected to all profile roots in a one-to-many relationship. A web profile root may for instance track the internet behavior of a user while a mail profile root observes the activity within a mail application.

This agent can be contacted by any outside agents or applications for obtaining an up-to-date profile of a user.

2.2 Collaboration Model

The agent collaboration model in Figure 3 specifies how agents collaborate and communicate. Basically the data streams within the collaboration model can be separated into three parts: a profiling part, a hierarchy management part and a profile extracting part. In the profiling part the user is observed by the several profiling roots.

When a root is presented with new data for profiling, this data is forwarded to the connected profiling agents. A profiling agent in turn collaborates with several classifiers to produce a profiling result for the incoming data. It forwards the data to the connected classifiers and combines the returned classifying results. When the profiling agent concludes that the current data can't be classified according to its assigned topic, the data flow will stop at this point. However, if the agent decides the current information can be classified according to the topic, the profiling agent will forward the information to its connected children and the profiling process will iteratively be repeated until the end of the hierarchy is reached or the profiling agents were not able to classify the incoming information anymore. The profiling agents send feedback to their parent nodes, indicating whether they were able to classify the incoming data. The sender of the data can use this feedback information to keep track of the information flow and is able to decide when agents should be added to or removed from the hierarchy.

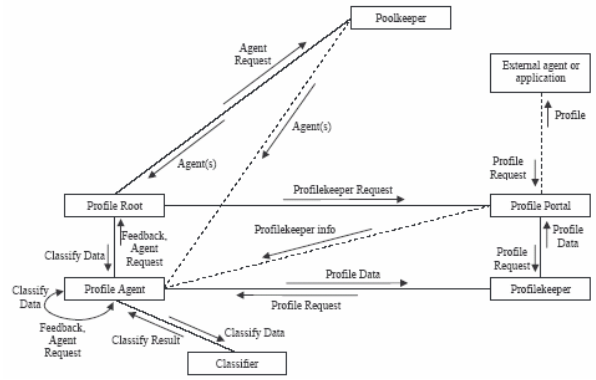

Fig. 3. The agent collaboration model

The data streams that concern hierarchy management are used for extending the hierarchy with new agents. Agents within the hierarchy can extend their set of connected agents by requesting a new agent from the agent pool. The pool keeper takes the request into consideration, checks the pool for the availability of the appropriate agents and returns a set of selected agents.

The pool keeper receives agent requests only from the profile roots. This approach makes the system more flexible since the roots of the profiling hierarchies are the only agents that are responsible for maintaining a connection to the agent pools. Whenever a pool is added or removed, only the profile roots have to be informed.

3 Classifiers

In the current system, a classifier receives a text document as input and will either conclude that the document can be classified as belonging to the specified topic or not. Neural networks have been used in a number of text classifying projects. In [CA01] a neural network approach is used to classify financial news articles, [RU02] describes a hierarchical text classification approach using neural networks and in [SE03] the application of neural networks for classifying web pages is investigated. We follow this approach in this project as well.

The network does not get fed the actual text, but rather the distinguishing features of the text such as length, language, number of words and number of sentences that could be useful for classifying.

The text classifier in our system uses trigrams, sets of three letters like AAA and IBM, as input to the network. In [HO99], a project for determining the author of a poem with neural networks, it was shown that trigrams in comparison with other n-gram (n-letter words) distributions produced the optimal results in classifying text. For the 26 characters in the alphabet, 26^3 = 17576 trigrams are possible, but in practice only about 9000 actually occur. Therefore, there is no direct need for preselection of trigrams. Maybe a smaller amount of trigrams would do for the current problem domain but, during the trainings process, the network itself should be able to figure out which trigrams are the most important and will increase the weights of these trigrams. In the neural text classifier, the input of the network is composed by collecting all the occurring trigrams within the documents of the training set. This set of trigrams is then trimmed down by the least and most occurring trigrams to exclude trigrams from stop-words and trigrams that only occur in very few documents of the training set. Every input of the network is then linked to one of the trigrams of the remaining set. The output layer of the network consists of a single neuron that should be activated when a document about the classifiers topic is fed to the network. Reinforced training is then applied and the network is tested with a test set to avoid overfitting of the training set.

3.1 The Neural Web Classifier

The web classifier is part of the hierarchy of agents that is able to monitor the web behavior of a user by observing the pages the user has visited. A single classifier

within this agent network determines whether a certain html page can be classified to the topic it has been trained to classify. The web classifier receives a web document from a profiling agent that in its turn received the data directly from the root agent or from a peer profiling agent. The web classifier should be able to classify an incoming web document and adjust his current profile if needed. For this purpose the neural web classifier is trained with positive and negative sample data about the assigned topic of the agent. The positive sample data contains representative web documents on the specific topic while the negative sample data contains any other web documents as long as they fall within the domain the classifier is placed. For example, if an agent that is specialized in managing a profile about soccer within the field of robotics, it should contain negative sample data about other robotics topics. An agent that is not trained within the right domain might still be a good classifier but will have a high misclassification than agents that are trained with the appropriate domain specific data. The ratio between the positive and negative samples is a reflection of reality. Thus, if the current domain is very large and a topic covers only a small part of this domain, this then a relatively small positive and a large negative data sample set of web documents is provided to the agent.

3.2 Other Classifying Algorithms

As shown in fig 2, one profiling agent can use multiple classifiers, using different algorithms. In the current implementation a simple title classifier is created that extracts the title from a web page and classifies the page according to that title by mapping it to a predefined set of key phrases. This approach is very simple but is a good example of the classifiers that can be used for special sources of information.

3.3 Accuracy of the classifiers

Currently two types of classifiers have been implemented to assist the profiling agents in obtaining a profile on a specific topic. The results of these classifiers basically determine the performance of the profiling process, since the other agents of the system depend on the outcome of the classifiers.

The results of the simple title classifiers show the expected results. When topics contain a number of specific and unique key phrases the classifier can produce reasonable results. The title classifier however only has been added to the system to show the ease of adding new classifiers and to show the collaboration with other classifiers. In a professional environment, the title classifier should therefore either be replaced by a more sophisticated one or be extended with some machine learning features to be really useful to the system. The current profiling system leans heavily on the results of the neural classifying agents. The neural classifiers show good results across multiple domains. During the training process, the neural classifiers show up 80 to 100% accuracy on the test sets, indicating that the training process is able to create good (and sometimes perfect) classifiers. It is hard to obtain exact precision and recall rates however, since there is no such thing as a test database for web classifying systems similar to the Reuters-21578 database that is often used to test text clas-

sifying algorithms. In general the results of the neural classifiers show a tendency to a higher accuracy within limited domains.

Experiments were performed to discover the ideal network topology of the neural classifiers. Multi layer networks structurally outperform the single layer network.

Although the neural classifiers perform well on the test sets, the classifiers may perform differently in practice if the training and test sets are a bad reflection of reality. The actual performance of the classifiers thus depends heavily on collecting representative data on the specified topics. In the current implementation this data selection process is done manually and can be very time consuming since the performance of the neural classifier improves when more training data is made available. Experiments show that at least 40 to 60 positive samples of web pages per topic are needed to train a good classifier.

Besides the positive samples, a classifier needs a set of negative sample data that represents the other topics within the domain a specific agent is classifying.

In the current implementation the neural network are trained automatically by providing sets of negative and positive sample data. In practice the negative sample database is built by using the positive sample data of other topics and adding some sample web pages of the remaining topics within the domain. In this approach still a lot of data has to be assembled manually though.

4 The TV Program recommender

To show the easy use of the currently implemented profiling system, the profiler is connected to an actual recommender system, a TV program recommender that is currently being developed by the Dutch Telematica Institute [TE04] in association with public broadcasting (omroep.nl). The TV program recommender takes a constructed profile in the form of an XML document, as in figure 4 as input and is able to recommend TV programs to the user based on this profile. The genre field is determined by the profiling system, and is used to determine the user's interest for TV programmes.

```
<profile>
  <pages> <page>
   <id>thePageId</id>
     <url>http://someurl.com
     </url>
     <title>Voetbal
         International
     </title>
    <genre>Sport</genre>
    <date>2004-2-23</date>
    <time>06:59:00</time>
   </page> </pages>
 </profile>
```

Fig. 4. Example XML input document of the TV Program recommender.

If for instance the profile indicates that a user is interested in sport and news, TV programs that cover these topics will be recommended to the user.

More complex profiles make it possible to advise the user in a more specific and sophisticated way. The TV recommender shows the straightforward use of a constructed profile by other applications. A special profile tracking agent is generated that requests and receives the profile from the profile portal once in a while and informs the recommendation application when modifications to the profile have occurred. Furthermore, a specific taxonomy for the profiling system defines what categories can be handled by the recommender. The 17 categories the TV recommender can handle are specified in the taxonomy as classes directly placed under the root and classifiers have been trained especially for this purpose. Due to the flexible agent structure the changes in the taxonomy and the new classifiers in the agent pool automatically lead to the forming of a new kind of profile hierarchies.

5 Conclusions

The implemented profiling system is able to create a profile of a user that is based on its internet browse behavior. The platform agents work well together in building the hierarchy and consulting each other about incoming web documents.

A number of advantages result from the agent approach to the profiling system. One of the most important advantages of the agent approach is the ease of adding new profilers and classifiers to the system dynamically without having to change any code within the other parts of the system. Currently two classifying algorithms are used. New algorithms and agents can be attached easily by adding them to the agent pool. This approach to the combination of multiple algorithms has shown to be easy and flexible. By delegating the responsibilities to the appropriate agent the architecture and implementations are simplified significantly.

Furthermore, the general agent approach makes it easy to retract the profile from the profiling system. Currently an XML version of the created profile is extracted for the use in a TV recommender system but the profiling system can be connected to any other application that is able to benefit from a structured profile of a user. As discussed above, we have connected our profiling system to a TV program recommender system, without these systems having been designed to collaborate

Another emerging advantage of the multi agent approach to profiling is the ability to distribute parts of the profiling systems when needed. This ensures scalability when a large number of agents are added to the profiling hierarchies.

Besides the mentioned advantages, the agent approach inevitably also results in some shortcomings, especially for development and maintenance. Since the agent system is basically a system of independent running threads, debugging the agent community has shown to be a difficult and time consuming task.

Another effect of agent oriented programming is the inability to access and interpret the state of the overall agent system in an easy way.

Agent oriented programming thus has shown to take some extra time in the development phase but the approach also offers major advantages in extending the system and connecting the agent system to new applications. Especially in the field of profiling where many different classifiers can be implemented to assist in the profiling process, the advantages of the agent approach have shown to be a suitable solution.

While the system provides promising results for user profiling, its major drawback for using it as a cross-domain profiling approach is the lack of openness of the architecture. In a world where profiles provided by different owners are to be combined, the use of open standards such as Web Services and Semantic Web Services is crucial. Other classifying algorithms are also being considered. Natural Text analysis seems a promising technology to provide additional information about texts above what trigrams may provide. Finally, the combination of other sources of information such as email with web pages is also being considered to provide additional profiling information. In the case of using email as a source, the issues related to privacy become very important, especially in the case where a company enforces a classifying solution upon its employees.

Acknowledgements

We wish to thank Peter Fennema for integration with the TV Recommender. This work is part of the MultimediaN project (http://www.multimedian.nl). MultimediaN is sponsored by the Dutch government under contract BSIK 03031.

6 References

[CA01] Calvo R.A. (2001) Classifying financial news with neural networks, 6th Australasian Document Symposium

[CH97] Chen, L. and Sycara K. (1997) WebMate: A Personal Agent for Browsing and Searching

[HO99] Hoorn, J.F. et al (1999) Neural Network Identification of Poets Using Letter Sequences, Literary and Linguistic Computing, 14(3), 311-338.

[IE04] IE Canvas, http://www.nothome.com/IECanvas/

[MI04] Middleton, S. E., Shadbolt, N. R. and De Roure, D. C. (2004) Ontological User Profiling in Recommender Systems, ACM Transactions on Information Systems (TOIS), 22(1), 54-88

[MO02 Montaner M, Lopez B, and de la Rosa J. (2003) A taxonomy of recommender agents on the internet. Artificial Intelligence Review, 19:285 - 330

[PA02] Padgham, L and Winiko, M. (2002) Prometheus: A methodology for developing intelligent agents, Third International Workshop on Agent-Oriented Software Engineering

[PR04] Protégé, Ontology Editor and Knowledge Acquisition System, http://protege.stanford.edu/

[RU02] Ruiz, M.E. and Srinivasan, P (2002) Hierarchical Text Categorization Using Neural Networks Information Retrieval, 5, 87–118

[SC04] Schuurmans J, Zijlstra E (2004). Towards a continuous personalization experience. ACM International Conference on Dutch directions in HCI

[SE03] Selamat, A. et al (2003) Web page classification method using neural networks, Transactions of The Institute of Electrical Engineers of Japan, 123(5), 1020-1026

[TE04] Telematica Instituut, http://www.telin.nl/