# LEARNING USER BEHAVIOUR IN A PERVASIVE SOCIAL NETWORKING SYSTEM

Elizabeth Papadopoulou, Sarah Gallacher, Nick K. Taylor, and M. Howard Williams
Heriot-Watt University
School of Math & Comp Sciences, Riccarton, Edinburgh EH14 4AS, UK
{ E.Papadopoulou, S.Gallacher, N.K.Taylor, M.H.Williams}@hw.ac.uk

**ABSTRACT**
Social networking systems and pervasive computing are two essential paradigms for systems of the future. There has been an increasing amount of research and development done on combining location awareness with social networking. Our current research is aimed at taking this a step further and combining more general pervasive system behaviour with social networking in a fully integrated way. In order to achieve this, one of the key functionalities on which the system is based, is that of context aware personalization. However, one of the major problems with personalization lies in dealing with the changeability of user preferences, and this needs to be taken into account when choosing a strategy to handle learning of user preferences. This paper presents an approach that we have been developing, which uses two different strategies in tandem – one based on a rule-based approach, the other on a neural network with which a user can interact. The paper briefly outlines these and then describes an experiment conducted to evaluate the time required by the neural network to adapt to changes in user preferences. This is used when the two approaches produce different results, to determine which results to use. It also provides input to help determine the frequency of execution of the learning algorithm used in the rule-based approach.

**KEY WORDS**
Pervasive systems, Social networking, Personalization, Neural networks, Rule-based preferences.

## 1. Introduction

The continuing development of the technology for sensors and other devices has led to dramatically reduced costs and an increasing range of different functionalities. As a result the number of devices of different kinds in the environment surrounding the user is growing rapidly. As the plethora of devices increases and the environment surrounding the user becomes more complex, so the need increases to assist the user to enable him/her to take full advantage of this. This is one of the motivations underlying the development of pervasive systems [1] – namely, to support the user in controlling and managing the growing numbers of devices (including sensors, computers and general appliances), networks and services that are available at any time or place. As a consequence, an increasing amount of research has been directed at finding solutions to the problems of pervasive and ubiquitous computing, and more and more prototypes are emerging to test different subsets of ideas in this area. Particular interest has been focused on fixed smart spaces, and examples of systems of this type include the Adaptive House [2], MavHome [3], Synapse [4], Ubisec [5], the Intelligent Home [6], etc.

On the other hand social networking is a paradigm that has developed rapidly with huge success and is now well established. By facilitating social connections between users on a very large scale through simple, easy to use interfaces, it has opened up a range of new opportunities for exploiting the Internet. Systems such as Facebook, Youtube, LinkedIn, Flickr, MySpace, etc., have transformed the way a large number of users use their systems, and their use takes up a significant proportion of the time that the average user spends on the computer.

Although these two paradigms are very different, they complement each other rather neatly, and significant benefits could be gained if the two can be brought together and integrated seamlessly into a single system with the benefits of both – a Pervasive Social Networking system. Already there are a number of applications in which location awareness has been combined with social networking – for example, systems such as FourSquare rely entirely on this. However, the idea of combining full pervasive system behaviour with social networking goes much further than this.

We are one of fifteen partner institutions working on the Societies project, which is a large European research project which aims to build on recent technical developments in pervasive computing and social networking to create a Pervasive Social Networking system. The project, which started in October 2010, has been developing a platform based on the use of mobile phones connected to backend servers using cloud technology. The software developed for the mobile phones themselves is based on Android although the approach used is general and could easily be extended to other mobile phone operating systems.

The project has reached a stage where the implementation will soon be tested in a number of real user trials. The first of these, a short Enterprise trial, is scheduled for mid-April while the major trial, based on the use of the system by a group of university students over a period of a few months, is scheduled to begin in October.

This paper is concerned with the problem of learning user preferences in such a system in order to personalise the behaviour of the system. The next section describes the specific problem in more detail and the solution adopted. Section 3 provides a brief background to this research. Section 4 describes very briefly the relevant processes relating to personalization based on rule-based preferences, and in particular how these preferences are acquired. Section 5 provides a slightly more detailed description of the neural network approach that we are using. Section 6 describes the experiment conducted on acquiring user preferences on the selection of channels for television sets. Section 7 presents the results obtained and section 8 summarises and concludes.

## 2. The Problem and Proposed Solution

One key aspect of a Pervasive Social Networking system for it to be acceptable to the end user is that it needs to be able to adapt its behaviour in response to the needs and preferences of the individual user as well as the circumstances prevailing at any point in time – or, in other words, it must be both personalizable and context aware. Not only does this include adaptation of the content of services and their presentation to the user but also any proactive behaviour which the system undertakes on behalf of the user (based on observations of their previous behaviour). However, this relies on the system having sufficient knowledge on what adaptations or actions to perform and in what context for each individual user. Since one cannot expect the user to provide such information directly, the strategy that is usually followed is to monitor the user's behaviour and apply machine learning techniques to infer preferences from it.

Unfortunately, in the case of a Pervasive Social Networking System this task is far from simple and presents a major challenge to system developers. At the heart of it one has the question as to which type of machine learning technique to use. Some pervasive systems use rule-based approaches to represent user preferences. These have the advantage that the user can view the state of user preferences at any stage and, if necessary, correct these. This can help to gain the confidence of the user. Other systems use a neural network or Bayesian network to capture preferences. In general these cannot be displayed to the user as they are not meaningful.

However, the choice of which technique to use is more complex than this. In the first place many user preferences are context dependent – e.g. preferences relating to the use of a particular service or device may be quite different when the user is at work from when he/she is at home or even out and about. As a result the preferences need to be built up gradually, reflecting the different actions that need to be taken in different contexts. As a result at any moment in time some of these preferences will be incomplete in that the appropriate contexts have not yet arisen and hence the system has not yet been able to ascertain what the user's preference would be in such cases.

Secondly, some user preferences will change with time. This can make it difficult, if not impossible, to extract a complete representation of some preference before it changes. This can lead to a conflict in which the system does not know whether a preference has changed permanently, is subject to a one-off change or simply that a new context situation has arisen that has not previously been taken into account.

In the Societies project the approach that we are experimenting with uses two different machine learning techniques in tandem – one based on a rule-based approach, the other on a neural network with which the user can interact in a similar way to the rule-based system.

Both techniques are used in the learning process and both are used to predict what action the system needs to take to personalize its behaviour at any point in time. As long as the results produced by the two techniques agree, the system proceeds with the action. If the two techniques disagree on what action should be taken, the system resorts to a conflict resolution process which is based on the degree of trust in each at this stage.

Furthermore, we believe that it is essential to keep the user in the loop wherever possible. Thus whenever the system takes any action on the user's behalf, it informs the user and allows the user to override this if it is not what he/she wants to happen. In addition, the user can inspect the user preferences at any stage to see why the system is behaving in a particular way and can alter these if they are not correct.

Both of these techniques have been implemented and are ready to use in the trials. This paper describes the two. It then describes an experiment conducted using the neural network to evaluate the time required by it to adapt to changes in user preferences. This is used in the conflict resolution stage when the two approaches produce different results, to determine which results to use. It also provides useful feedback for determining the frequency of execution of the learning algorithm used in the rule-based approach.

## 3. Background

While there is general agreement on the concepts of pervasive systems, the individual systems which have been developed to test ideas in this area have varied considerably in their focus, adopting different assumptions, exploring different approaches, developing different architectures and creating different prototypes. Some have focused on the idea of fixed smart spaces such

as the “smart home”. These are generally concerned with providing support for elderly and disabled people to help them to maintain their independent living. Examples include [2, 3, 4, 5, 6].

Other fixed smart space systems include the smart office, smart building, etc.; for example, MIT’s Project Oxygen [7] creates intelligent spaces inside offices, buildings, homes and vehicles using sets of embedded devices.

Besides the work done on fixed smart spaces, another major focus in which there has been considerable interest lies in developing systems to support the mobile user. Examples of this type of pervasive system include mobile systems such as Mobilife [8], Spice [9], Daidalos [10], and so on.

The Persist project [11] aimed to produce a general pervasive system, based on the notion of a Personal Smart Space (PSS). This approach combined the developments on fixed smart spaces with those of mobile systems to create a new type of system in which the user is constantly covered by their own pervasive PSS. One consequence of this is that the facilities that a system can provide and the way in which it will behave at any point in time will depend not only on its own resources and characteristics but also on those of any other PSSs which may be nearby.

Basically a PSS consists of a set of devices that belong to a single user together with services that are owned, controlled or administered by the user. The collection is a dynamic one in that individual devices can join or leave whenever they need to do so. They are connected together in a network using peer-to-peer communication in such a way as to behave like a single system (although each device can operate independently if required). Furthermore the set of services associated with the PSS can be shared with other PSSs or protected from being seen by other PSSs depending on the current context. A significant advantage of this is that it does not require any fixed infrastructure (although it is able to take advantage of such infrastructure where it is present).

Other important properties of a PSS include:

(1) A PSS may be either fixed or mobile.

(2) A PSS must be able to identify and interact with another PSS when they are in close proximity.

(3) A PSS must be context aware and personalizable.

In the Persist project the architecture [12] adopted for a PSS is shown in Figure 1.

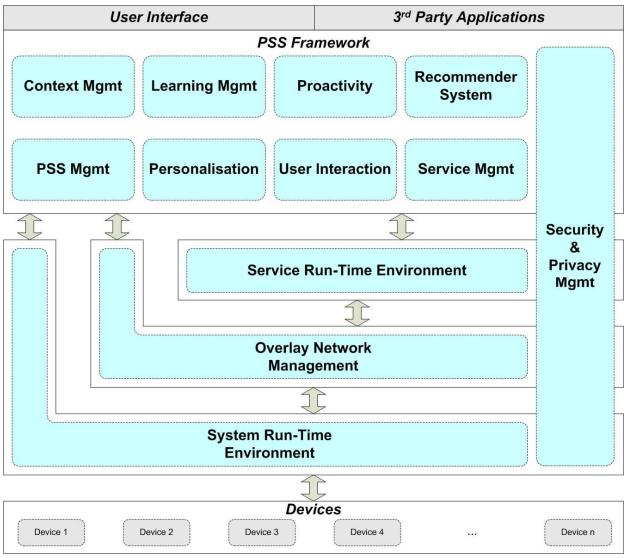


Figure 1. The high level architecture used to develop a prototype of a Personal Smart Space in Persist

The Societies project [13] is building on some of the ideas developed within Persist to produce a Pervasive Social Networking system which combines the ideas of pervasive systems with those of social networking in a seamless integration.

Applications that involve the use of location information within social networking systems generally start with a social networking system and extend this accordingly. However, in order to create a system in which full pervasive system behaviour is combined with social networking functionality, the approach we have followed is to build a pervasive system with its own social networking functionality which can connect to and interact with other existing social networking systems.

The pervasive system functionality is based on a variation of the model of a PSS described above. However, in order to incorporate social networking functionality, this has been extended to introduce the notion of a community with the appropriate functionality needed to manipulate this and integrate it with the properties of a PSS.

In both of these types of system (Persist pervasive platform and Societies PSN system), one of the major challenges lies in the development of approaches that will alleviate the user from some of the detailed interaction and decision making that is needed. To do this in a way that is acceptable to the user, it is essential that his/her needs and preferences are taken into account. This constitutes personalization of the ubiquitous/pervasive systems. By personalization we mean the process of creating, maintaining and applying user preferences in decision making, since it has the effect of tailoring the system’s behaviour to the individual needs and wishes of the user so that it appears or acts differently for different users or for the same user under different circumstances.

## 4. Personalisation Based on Rule Based Preferences

A number of the pervasive system prototypes that have been developed employ a simple rule-based format to represent user preferences. The major advantage of this is that the result can be read and understood by the end-user. This enables the system to provide the user with the ability to understand the actions it performs on his/her behalf and change them manually if necessary. This in turn gives the user ultimate control over the ways in which their environment is adapted.

Since many user preferences are context-dependent, it is natural to use an IF-THEN-ELSE format – in our case, a nested IF-THEN-ELSE format. The condition part of each IF-THEN-ELSE contains conditions based on user context. The result of executing such a rule is referred to as the outcome, and represents an action that the system needs to perform.

In the Societies Pervasive Social Networking system the user will generally start off with an initial default preference set. This could simply be a standard default set or one could provide different default subsets for different types of users, i.e. some form of stereotyping. Whatever the case, this initial set merely provides a starting point which is adapted with time as the individual user's preferences become known. In the process existing preferences may be altered or refined while new preferences may be discovered and added.

The process of refining existing preferences and acquiring new ones is achieved through monitoring user actions and inferring preferences through some form of machine learning.

The type of action that is referred to here is any act performed by the user that changes the behaviour of a service – whether an internal service of the PSN system or an external third party service. Thus the first step is to identify the particular types of action that are needed for user preferences.

The component responsible for User Monitoring is alerted whenever an action of the type referred to is identified. The information about the action is then stored together with the relevant context information in the History database. The crucial challenge here lies in selecting "relevant" context since storing the complete set of context attributes each time an action is encountered would lead to huge storage requirements and a significant increase in processing requirements while most of the context data would not be relevant.

One approach which helps to reduce the problem is to identify groups of actions that have the same or similar sets of relevant context attributes. However, ultimately the challenge of distinguishing what context attributes are relevant for what actions, rests with the system developer to resolve.

The algorithm which we are using in the Societies platform for inferring preferences from the History database is based on C45. Gain ratios are used instead of simple Gain to avoid any problems that might arise from attributes with multiple values. The algorithm has also been adapted to include the calculation of confidence levels that are used in subsequent preference merging and conflict resolution.

However, this led to several problems as the size of the History database grew. As a result a two-phase approach has been adopted in which the database is divided into two partitions corresponding to short-term and long-term memory.

The short-term memory store is used to contain the set of tuples (user action + context) that have been captured since the last execution of the learning algorithm. When the next execution of the learning algorithm is triggered, it is the data in the short-term memory that is used for this purpose. The preferences obtained from this are then merged with the existing preference set to produce an updated set. The data in the short-term memory data set are then transferred to the long-term memory data set which contains the complete set of data for the user (or an appropriate subset thereof).

If a conflict arises when merging the new preferences with the existing preferences, the learning algorithm can be applied to the complete data set to resolve such conflicts.

## 5. Personalisation Based on Neural Network

The second technique that we are using to handle personalization in the Societies Pervasive Social Networking system is that of a neural network (ANN). However, our strategy was constrained by two factors:

(1) As stated in section 2, one of the important assumptions that has underpinned our development was to keep the user in control so that as far as possible he/she should be able to understand and control the way in which the system adapts itself to his/her needs. In the case of rule-based preferences, it is easy to display these to the user but for neural networks this is a more difficult problem.

(2) The other problem outlined in section 2 was that of the changeability of the user and how user preferences may change with time. Thus one needs to have a neural network that takes adequate account of the temporal effect.

In order to address the first problem we decided to use a fairly simple neural network and to develop an algorithm to map this into a set of rules for the user to view and change in the same way as rule-based preferences.

Although the challenge of extracting rules from neural networks has been an area of research in the network community, in general techniques for doing this have not been taken up by the developers of pervasive environments to present user preferences to users. The general aim of rule extraction research has been to improve understanding of neural network behaviour by extracting a rule based explanation of network functionality that could be used to create better

classification systems while interpretation of the actual knowledge held in the network comes as a secondary benefit. On the other hand our aim is more user-centric and focused on performing two-way interpretation of the knowledge held by the network (i.e. from network weights to preference rules and vice versa).

For this reason the type of neural network selected was a binary neural network that takes real world inputs relating to the user's context and the selected preference outcomes, and learns associations between them in an incremental online manner. It is essentially a single layer model, although for simplicity it will be described in terms of two layers with weighted connections between them as shown in Figure 2.

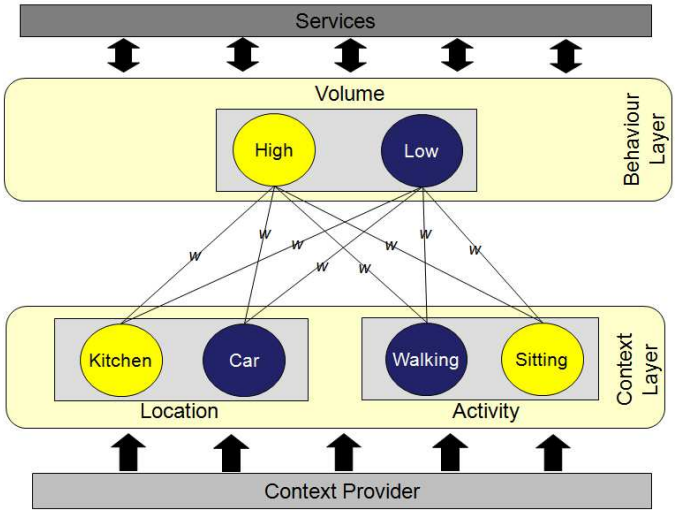


Figure 2. The topology of the neural network employed in the Societies Pervasive Social Networking system

The user's current context provides the input to the context layer and whenever the context changes this layer is updated. Likewise the user's current behaviour (in the form of actions on the part of the user in interacting with a particular service) is mapped onto the behaviour layer. In both cases a binary representation is used in which nodes in this layer are activated or deactivated in accordance with whether the corresponding value is true or false.

To reduce the problem of conflicts, nodes relating to the same behaviour or the same context parameter are grouped together and checks for mutual exclusion are applied to each such group. For example, one may have several nodes describing the location of the user, such as "home", "office", "cinema", etc. but only one can be true at any point in time. The same applies to behaviour nodes. This overcomes the problem of conflicts in context or behaviour values.

Since context nodes represent the input layer of the neural network, their activation depends on updates from the real world. Their input potential is binary and is directly dependent on their activation. Thus when context node $c_i$ is active it has an input potential of 1 and if it is not active, its input potential is 0.

A behaviour node may be activated either by a behaviour update from the real world (corresponding to the user taking an action that must be learnt as a preference) or by internal network knowledge (the network recognizing a situation in which an action is normally taken, and applying the preference). Each behaviour node has an associated output potential value. This is a measure of how true the system believes this node to be in the current context. The output potential is the sum of its inputs multiplied by their associated weights. Thus the output potential of node $b_j$ at time t is defined as:

$$op(b^t{}_j) = \sigma\left(\sum_{i=0}^{n} w^t{}_{ji} c^t{}_i\right)$$

where $c^t_i$ is the input potential of context node $c_i$ at time $t$, $w^t_{ji}$ is the weight value between behaviour node $b_j$ and context node $c_i$ at time $t$ and $\sigma$ is the squashing function that maps the output potential from the possibly very large range of values to a finite range of values between -1 and +1.

Once the output potentials have been computed, the values for all behaviour nodes in the same group are compared and the node with the largest value is selected as the active node unless the system is receiving a contradictory update from the real world (e.g. the neural network may predict that the attribute 'volume' should be set to "low" whereas the user has manually set it to "high"), in which case the conflict must be resolved in real time.

The second problem which needs to be taken into account is the temporal effect. In order to deal with this, the algorithm used consists of two phases, a layer update process and a learning process, which are executed one after the other in a continuous loop with an appropriate frequency.

In the first phase (layer update) any new updates of context or behaviour received since the last cycle, are processed and the appropriate nodes updated. Then using the Hebbian/anti-Hebbian Learning rule, all weights in the network are updated using the activity values of the context and behaviour nodes to which they are connected. Using these new values for the weights, the new output potentials are computed and the active behaviour nodes for each group identified. If there is a conflict between the new set of active behaviour nodes and the values arising from the real world, this is dealt with as before.

In this way the learning process takes account of temporal information relating to the duration of user behaviours and context states. In other words the strength of the connections between nodes in the context layer and those in the behaviour layer depends not only on the fact that the context states and behaviour values occurred together at the same time but also on the length of time that this combination co-occurred.

Finally the challenge of extracting rules from this to present to the user will be described in a future paper. The focus of this paper is on the system's ability to adapt to changes in user behaviour patterns.

## 6. Experiment with Neural network

This section describes an experiment which we conducted using the neural network on its own to determine the time required by the neural network to adapt to changes in user preferences. This is an important factor that is used in the conflict resolution process when the two techniques (rule-based and neural network) produce different results.

To put this experiment in context, consider the following scenario:

"John has created a PSS in his home in which a number of devices can be controlled as part of the PSS. On Wednesday his alarm goes off at 7 am as he has to be at work at 9 am. Once the system detects that John has moved to the bathroom it switches on the coffee-maker as John always starts the day with a mug of coffee. When John emerges from the bathroom, the television set in the kitchen is switched on to the news channel in preparation for him to come through for his coffee and toast. At 8.30 am it contacts his 'Lift Club Community' (a community of friends who share cars to travel to/from work) and checks the time that John is to be collected to go to work. It notifies John that pick-up is scheduled for 8.40 am. Of course, if it had been a weekend or a holiday, the system would have behaved completely differently."

Although this scenario focuses mainly on pervasive behaviour, one could use other scenarios that focus on social networking with some pervasive aspects. However, an example like this is relatively complex with a number of different user preferences/behaviour patterns inter-related. To simplify this for the sake of our experiment, we selected one aspect of the scenario – switching on the television to the user's preferred channel.

If this particular behaviour pattern (switching on the television in the kitchen to the news channel) was completely static and did not change, it would not matter which learning algorithm was used. The system would behave exactly as the user wanted it to. Moreover, once set up, the user would never need to inspect the preferences or change them in any way. However, as pointed out earlier, John's routine may change for a variety of reasons and the system needs to be able to adapt to such changes.

For our experiment we created a mock up using three television sets (or rather three computers controlling large screens masquerading as television sets) located in different locations in the building. Eight possible channels were selected which could be viewed on any of the television sets. Twenty four participants were recruited, the majority of whom were postgraduate students in Computer Science.

Each participant was told that the aim was to experiment with learning user preferences associated with location so that one could distinguish viewing habits of the user at home from those at work or in other locations, and identify what channel the participant preferred to watch in which locations. To this end the "televisions" that had been set up in different locations represented television sets at home, at work or wherever the participant thought appropriate (e.g. friend's flat).

Each participant was given an RFID tag to wear to keep track of their location. They were then taken on a circuit, visiting each screen in turn so that they could select a channel for that screen.

In order to do this each participant was allowed to view as many of the channels as he/she wished at each television before making a final choice. The number of channels viewed at each television varied from a couple to all eight. Once a channel had been selected for a television, this was noted and the participant moved on to the next screen where he/she repeated the process, selecting either the same or a different channel at each screen.

This cycle was then repeated three times, but in this case using only the final choice for each television rather than browsing through the channels. In each case the system selected a channel when it detected the presence of the participant near a screen. The channel that was selected by the system was noted and the participant corrected this choice if it did not show the correct channel. This corresponds to the preference learning phase and established the initial preferences. The circuits in this phase are referred to as initial circuits.

The second part of the experiment was aimed at investigating the effect of a change in a user preference. For this each participant was told that they could change one or more of the channels associated with the television sets, and note any such changes. They then repeated the process of visiting each television set and correcting channels where necessary – completing a further five circuits of the screens. In this phase the circuits are referred to as secondary circuits and each participant could change as many of the selections as they wanted to.

While this experiment represents a relatively simple example, the aim was to extract one simple case from the scenario described at the beginning of this section and study how the neural network coped with changing preferences in a real environment. We could have made this more complex (using a combination of different parts from the scenario) although this would not necessarily have provided any better understanding of its behaviour.

## 7. Results

The results from the first phase, the learning phase, are presented in Table 1.

Table 1
Percentage accuracy of the neural network in learning user preferences over the initial circuits

| After initial circuit n | Percentage accuracy |
|---|---|
| n = 1 | 98.5% |
| n = 2 | 100% |
| n = 3 | 100% |

As can be seen from this, the neural network stabilizes very rapidly to produce correct predictions. This is what one might expect.

However, the challenging part of the experiment lies in the second phase where users changed their preferences. For the analysis of the results obtained, the instances where the participant did not alter their preferred channel for a particular television have been removed from the set of results since the accuracy in each case was 100%. For the remaining instances, where the participant did change their preference, the average accuracy of the neural network at each stage in the five circuits is shown in Table 2.

Table 2
Percentage accuracy of the neural network in adapting to changed user preferences over the secondary circuits.

| After secondary circuit n | Percentage accuracy |
|---|---|
| n = 1 | 7 % |
| n = 2 | 27 % |
| n = 3 | 43 % |
| n = 4 | 86 % |
| n = 5 | 97 % |

The question then was whether this level of response to learning a user's preference was perceived to be acceptable to the participants. As a result at the completion of the experiment each participant was given a questionnaire to determine their reactions to the experience. Twenty two of the participants completed this and some of the results are shown in Tables 3 and 4. Taking 5 as the highest, 1 the lowest and 3 as neutral, it is clear that no one was unhappy with the automatic screen selection and only 6 found the choice of incorrect channels in the process of learning annoying. In general participants do not have a problem with being monitored nor with the system attempting to predict their behaviour. Ultimately, in Table 4 it can be seen that over 86% of the participants would either definitely use this functionality or would possibly do so.

Table 4
The ultimate test question on the questionnaire

| | Yes | No | Maybe |
|---|---|---|---|
| Would you use such functionality in your own home if it were free? | 10 | 3 | 9 |

## 8. Summary and Conclusions

The context of this research is the work currently being done to develop Pervasive Social Networking Systems. The Persist project [12] developed a pervasive system based on the notion of Personal Smart Spaces. This work is currently being extended in another European research project, Societies. We are one of 15 partners in this large European research project, which is aimed at combining the concepts of pervasive systems with those of social networking to produce a pervasive social networking system. The first complete prototype of the system is nearing completion and will be subjected to real user trials scheduled for March and October-December 2013.

One of the key functionalities needed in such a system is that of personalisation. And a crucial aspect of this is the ability to learn user behaviours and preferences. For this two different mechanisms are being used by the system. These two techniques correspond to the two basic strategies generally employed in pervasive systems, namely:

(1) Rule-based approach. For this the data from monitoring the user's behaviour is accumulated until an appropriate point is reached and then analyzed using an appropriate batch processing learning algorithm. In our case the algorithm used is an extension of C45 with confidence levels associated with the preference rules. As a batch processing strategy, this does suffer from the problem that preferences are only updated when an analysis is performed.

(2) Neural network approach. This corresponds to an incremental learning strategy. However, even this does not change the predictions immediately.

Table 3
Results of questionnaire for 22 participants completed after the experiment. 1 is lowest, 5 highest.

| | 5 | 4 | 3 | 2 | 1 |
|---|---|---|---|---|---|
| How pleasing did you find automatic changing of screens to correct channel? | 5 | 13 | 4 | 0 | 0 |
| How annoying did you find selection of incorrect channels? | 2 | 4 | 5 | 9 | 2 |
| How comfortable were you with system monitoring your behaviour during trial? | 11 | 5 | 4 | 2 | 0 |
| How comfortable were you with system predicting your behaviour during trial? | 9 | 8 | 3 | 2 | 0 |

To obtain the best results, both techniques are used in tandem. As long as they both produce the same results, the system can act on their agreed recommendations. However, if they disagree, the system refers them to a conflict resolution process to decide on what action to take.

One of the key problems that any learning system of this type needs to address is that of changes in user behaviour/preferences. To assist the conflict resolution process in determining which results to select in the light of changing preferences, an experiment was conducted with the selection of television channels to determine the time required by the neural network to adapt to such changes. This showed that after four repetitions of the changed preference, the neural network had reached 86% reliability and after five 97% reliability. This can be used in the conflict resolution process to help in the decision.

This also provided useful information for the rule-based user preferences. In this case the accumulated History data is analysed from time to time to extract new behaviour patterns/preferences. The question is how frequently this should be done. From the results of the neural network experiment an acceptable point at which to perform a new analysis in the absence of user guidance would be after the same preference has been found to fail for the fourth time.

## Acknowledgements

This work is supported by the European Union under the FP7 programme (Societies project) which the authors gratefully acknowledge. The authors wish to thank all colleagues in the Societies and Persist projects, without whom this would not have been possible. However, it should be noted that this paper expresses the authors' personal views, which are not necessarily those of the Societies consortium. Apart from funding the Societies project, the European Commission has no responsibility for the content of this paper.

## References

[1] M. Satyanarayanan, Pervasive computing: vision and challenges, *IEEE PCM*, *8*(4), 2001, 10—17.

[2] M.C. Mozer, Lessons from an Adaptive House, in D. Cook & R. Das (Eds.), Smart Environments: Technologies, protocols and applications, 2004, 273-294.

[3] M.G. Youngblood, L.B. Holder & D.J. Cook, Managing Adaptive Versatile Environments, *Proc. 3rd IEEE Int. Conf. on Pervasive Computing and Communications (PerCom '05)*, 2005, 351-360.

[4] H.K.Y. Si, A Stochastic Approach for Creating Context-Aware Services on Context Histories in Smart Home, *Proc. ECHISE2005, Pervasive '05*, 2005, 37-41.

[5] J. Groppe & W. Mueller, Profile Management Technology for Smart Customizations in Private Home Applications, *Proc. 16th Int. Workshop on Database and Expert Systems Applications (DEXA '05)*, 2005, 226-230.

[6] V. Lesser, M. Atighetchi, B. Benyo, B. Horling, A. Raja, R. Vincent, T. Wagner, P. Xuan, & S. Zhang, XQ.: The Intelligent Home Testbed, *Proc. Anatomy Control Software Workshop (Autonomous Agent Workshop)*, 1999, 291-298.

[7] L. Rudolph, Project Oxygen: Pervasive, Human-Centric Computing—An Initial Experience, *Proc. 13th Int. Conf. on Advanced Information Systems Engineering*, Springer, 2001, 1–12.

[8] M. Strutterer, O. Coutand, O. Droegehorn, & K. David, Managing and Delivering Context-Dependent User Preferences in Ubiquitous Computing Environments, *Proc. Int. Symp. on Applications and the Internet Workshops (SAINTW '07)*, 2007.

[9] C. Cordier, F. Carrez, H. Van Kranenburg, C. Licciardi, J. Van der Meer, A. Spedalieri, J. P. Le Rouzic, & J. Zoric, Addressing the Challenges of Beyond 3G Service Delivery: the SPICE Service Platform, *Proc. Workshop on Applications and Services in Wireless Networks (ASWN '06)*, 2006.

[10] M.H. Williams, N.K. Taylor, I. Roussaki, P. Robertson, B. Farshchian, & K. Doolin, Developing a Pervasive System for a Mobile Environment, *Proc eChallenges 2006 – Exploiting the Knowledge Economy*, IOS Press, 2006, 1695 – 1702.

[11] M. Crotty, N. Taylor, H. Williams, K. Frank, I. Roussaki, & M. Roddy, A Pervasive Environment Based on Personal Self-Improving Smart Spaces, *Proc. Ambient Intelligence 2008*, Springer-Verlag, Heidelberg, 2009, 58 - 62.

[12] E. Papadopoulou, S. Gallacher, N.K. Taylor, & M.H. Williams, A Personal Smart Space Approach to Realising Ambient Ecologies, *Personal and Mobile Computing, 8*, 2012, 485-499.

[13] S. Gallacher, E. Papadopoulou, N.K. Taylor, F.R. Blackmun, & M.H. Williams, Intelligent Systems that Combine Pervasive Computing and Social Networking, *Proc Ninth International Conference on Ubiquitous Intelligence and Computing (IEEE UIC 2012)*, IEEE Computer Society, 2012, 151-158.