

Giving the User Explicit Control over Implicit Personalisation

Sarah McBurney¹, Nick Taylor¹, Howard Williams¹ Eliza Papadopoulou¹

Abstract. Personalisation has an important role to play in a pervasive environment, supporting resource management tasks and tailoring the system to behave in ways that suit the user. However, this depends on creating and maintaining an adequate set of information on the user's preferences. Experience has shown that one cannot simply rely on the user to input preferences, and it is essential to employ some form of machine learning to support this; on the other hand the user needs to be able to control this. Two major approaches used for this purpose are rule-based strategies and artificial neural networks (ANNs). Within the Persist pervasive system these two different approaches are being combined to give maximum benefit. However, in order to enable the user to maintain control over the resulting set of user preferences, it is essential that he/she can see the current state of this preference set and change it whenever this is required. This paper describes how this will be achieved.

1 INTRODUCTION

Since the initial ideas of ubiquitous and pervasive computing were put forward [1] over 16 years ago, much effort has been invested in achieving them. As the costs of devices continue to fall dramatically, the number of different networks open to the user rises and the number of available services soars, it is increasingly apparent that the goal of pervasive environments with ubiquitous access to services, networks and devices comes at a price. The burden on the user to manage resources becomes great unless the system provides adequate support. This is done through some form of personalisation. By this we mean that the system retains knowledge about the user's preferences in an appropriate form and applies this in the decision making process so that the system behaves differently for different users according to their needs.

Early personalisation systems [2,3,4] relied on the user to provide and manage their preference set manually. Although such *explicit* personalization may mitigate resource management to some degree, the burden of manually managing an entire preference set means that user involvement is merely shifted to preference management responsibilities. For example, each time a new service, network or device is encountered in a pervasive environment the user's existing preference set may need to be refined or new preferences added.

For this reason, many pervasive and ubiquitous projects [5,6,7] include machine learning techniques coupled with user behaviour monitoring systems to provide *implicit* personalisation

where preferences are created and managed on behalf of the user. These automatic learning techniques fall into two broad classes – those that are based on data structures that are meaningful to the user (generally some form of rule-based representation) and those that are not (generally some form of network representation). The former have the advantage that the user can be engaged in the process of building up the set of preferences, manually changing preference rules that have been wrongly inferred or adding or deleting rules as needed – thereby using both implicit and explicit approaches to complement one another, reducing the need for user intervention but allowing the user full control of their preferences when required. The second class of techniques (including neural nets and Bayesian approaches) can be more efficient for some decisions but the user simply has to accept the result and cannot view or change the data structure created.

However, for some decisions rule based approaches are more useful while for others network based ones are preferable. The Daidalos project [8], which developed a pervasive system aimed at the mobile user, and the Persist project [9], which is developing a pervasive system based on Personal Smart Spaces (PSSs) [10], both employ a combination of automatic learning techniques combined with user involvement. One aim was to achieve the optimal situation in which the learning algorithm can update the preference data structure whenever a change is learnt and the user can inspect the preferences and change them whenever he/she needs to. This is straightforward in the case of rule-based approaches, but it can also be achieved in the case of a neural net using an appropriate form and algorithms to convert the neural net to a set of preference rules and vice versa. The goal of this paper is to present this process.

The next section describes the rule-based preference format used in Daidalos and explains how the Persist project will utilise these rules as well as providing preferences stored as neural networks. Section 3 outlines the problem of translating neural network knowledge into human readable form and sections 4 and 5 propose a possible solution that will be implemented in the Persist project. Section 6 provides a conclusion.

2 PREFERENCE MANAGEMENT

In the rule based approach that was used in Daidalos, and will be used in Persist, user preferences are represented by an IF-THEN-ELSE rule format and are therefore human-readable. Most preferences are context-dependent although context-independent preferences can also be represented. The condition part of an IF-THEN-ELSE consists of a number of primitive conditions, generally tests on context attributes, combined by logical operators (AND, OR). The THEN and ELSE parts consist of outcomes, in the form of actions, or a nested IF-THEN-ELSE.

¹ Dept. of Mathematics and Computer Sciences, Heriot-Watt University, EH14 4AS, UK. Email: {ceesmm1, nkt, mhwh, ceeep1}@macs.hw.ac.uk.

The outcome is followed when the condition is met (e.g. if the outcome is an action it may indicate that a service is started or a video stream is paused). A full BNF description of the preference format is presented in [11].

In order to assist the user in getting started, stereotypes may be used to create an initial set of preferences. By selecting appropriate stereotypes, a set of preferences can be generated, which provide an approximation to what the user wants. Thereafter automatic learning techniques are used to update this set. The user can also view the rules at any time and change them as needed. The learning technique used was an adapted version of Quinlan's tree building algorithm [12], the output of which can be easily translated to the human-readable preference rule format.

The Persist pervasive system will extend the approach used in Daidalos, using both a rule-based approach and a neural net (ANN). This will provide maximum flexibility in that either technique may be used for handling any subset of preferences. For example, the preferences for selecting a service, deciding on which device or network to use, or personalising a third party service, are all different and for each different techniques may be more suitable.

However, in order to maintain the ability for the user to interact with the preferences a more sophisticated ANN learning algorithm (to be fully described in a forth-coming paper) will be used for preference learning.

3 PROBLEM DESCRIPTION

Interpreting the Persist ANN into the human-readable preference format mentioned above is the challenge of rule extraction from neural networks. Although this area has received much research it seems extraction techniques have rarely been used in pervasive environments to present learned preferences to users. This may be due to the different goals of each field. The general aim of rule extraction research is to better understand neural network behaviour by extracting a rule based explanation of network functionality that could be used to create better classification systems. Interpretation of network knowledge comes as an aside. Our aims in the pervasive domain are more user-centric and focused on performing two-way interpretation (i.e. from network weights to rules and back again). Several challenges arise from this user-centric, pervasive problem domain.

One significant problem is ensuring that the extracted preference set is devoid of duplicates and large, over-complex preferences. A typical neural network can contain numerous nodes and even more inter-connections. The translation of a complex network should not result in numerous preferences relating to the same personalisable element, neither should it result in a very large preference with multiple nested IF statements. A typical user would find such a preference set impossible to understand even if it is human-readable.

Another challenge is allowing the user to view their entire preference set, not just the preferred preference outcomes for the context the user is currently in. This requirement indicates that an exhaustive rule search is required. Such searches are expensive both in terms of processing time and memory and their cost increases with the size of the network. It is undesirable for a long delay between the user requesting to view the preference set and it being displayed. Further, the mobile nature

of pervasive devices places a limit on processing power and memory.

4 PRESENTING USER PREFERENCES TO THE USER

The Persist ANN is a binary neural network that takes real-world inputs regarding the user's context and the selected preference outcomes. It is a single layer network but is described in terms of two layers for clarity. The context layer represents the user's context and acts as an input layer. The outcome layer represents the user's selected preference outcomes and acts as both an input and output layer hence retrieval is possible during network learning. Figure 1 shows the high-level Persist ANN topology and input/output fields.

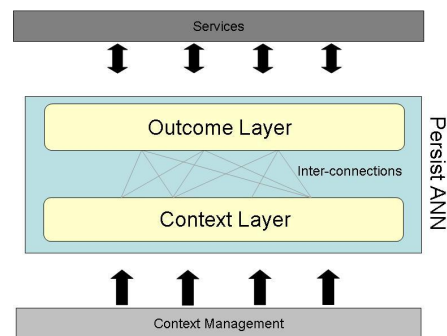


Figure 1. Persist ANN

The Persist ANN has been designed to overcome the challenges listed above. Binary activations and linear connections reduce internal complexity, hence simplifying the network interpretation process. The weight on each network connection is directly related to the strength of the association between some context node and some outcome node. Therefore, by comparing weights, weak connections can be disregarded. The benefits of such pruning have been recognised in rule extraction. It is generally agreed that in order to extract simpler, more concise rules, the network should have less connections. Therefore, by reducing the number of connections through the removal of weak, less influential ones we can more easily extract preferences that consist only of those context values with the strongest influence on the preference outcome. By reducing the number of connections we are also minimising the search space and hence the search time relieving the burden on processor time, processor power and memory required.

Consider the situation where we have the network state shown in Figure 2. (Each node is connected to every other node in the opposite layer. The strength of each connection is determined by the weight value at the synapse on each connection):

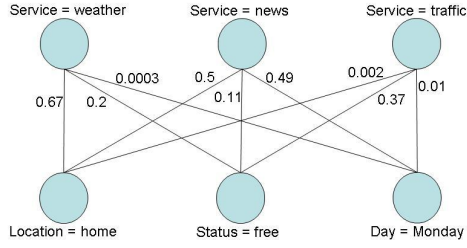


Figure 2. Example Network State

First we prune the network to remove the connections with a synapse value below some threshold Θ (e.g. 0.2). This means we lose connections 0.0003, 0.11, 0.002 and 0.01. From the remaining connections we can now create IF-THEN-ELSE preference rules. One possible approach involves systematically activating the possible patterns of context nodes and capturing the preferred outcomes for each pattern creating a set of IF-THEN preference parts. These individual preference parts can then be merged together using ELSE branches and logic simplifications. In this example the resulting IF-THEN-ELSE preference will be:

```

IF location='home' OR (location='home' AND
status='free')
THEN
service='weather'
ELSE IF status='free'
THEN
service='traffic'
ELSE IF day='Monday'
THEN
service='news'

```

5 CAPTURING MANUAL USER UPDATES

Giving the user complete, final control of personalisation means allowing the user to manually manipulate their preference set. This could be creating new preferences, deleting preferences or changing the outcome or conditions of existing preferences. Whatever the case, the user updates must be captured in the human-readable format and translated into the internal preference format understood by the system (i.e. neural network weights).

As mentioned above, one of the rule extraction research goals is to use a rule base (usually rules extracted from the neural network) to create new neural models. In a sense it is translating rules back to neural network form. Some inspiration can be taken from this area to overcome the challenge of translating the user's preference set back into neural network form. However, this is limited as in our problem domain we do not wish to create a new network model. The Persist ANN will continue to exist internally after human-readable preferences have been extracted and presented to the user. If the user performs any updates on their preference set while in human-readable format we wish to translate only those updates back into the existing network. We do not wish to create a new ANN.

To create a new preference the user must specify an IF-THEN-ELSE Daidalos preference format rule through the preference GUI. Each IF-THEN branch will contain a context

condition (set of context tuples e.g. 'location = home') and an outcome (e.g. 'service = weather'). To translate a new preference into the internal network structure we complete several steps. Firstly we must identify context nodes that represent the stated context condition values. Secondly we need to identify the outcome node that represents the related preference outcome. Finally we increase the weight value on the connections between these context and outcome nodes making the connections excitatory. This ensures that the outcome node will fire when the user stated context condition is true. When the user wishes to delete a preference we prune the network to remove the output nodes that relate to the deleted preference. We cannot remove the nodes relating to the context values in the preference condition as these nodes also provide input to other output nodes.

When the user updates an existing preference they can either change the context condition or change the outcome. If the context condition has changed we must identify the context nodes that represent updated attributes in the context condition. By manipulating the weight value on the connections between such nodes and the node representing the preference outcome we can ensure that context nodes (and hence the related context values) reflect the user specified influence over the activation of the outcome node. If the preference outcome has changed we must identify the outcome node that represents the updated preference outcome. As above, we alter the weight value on the connections between this outcome node and the related context nodes to ensure that they are excitatory or inhibitory as specified by the user.

Consider the example above. Imagine the user changes the IF-THEN-ELSE preference (generated from the network) to:

```

IF location='home' OR (location='home' AND
status='free') OR day='Monday'
THEN
service='weather'
ELSE IF status='free'
THEN
service='traffic'

```

To translate this back into the network we must ensure that the strength of the connection between 'day=Monday' and 'service=news' does not over-ride the strength of the connection between 'day=Monday' and 'service=weather'. To do this we must inhibit the former connection and excite the latter. Figure 3 shows the updates represented in the network.

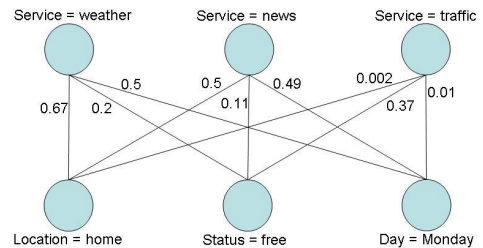


Figure 3. Updated Network State

Notice that the strength of the connection between 'day=Monday' and 'service=weather' has been increased just

enough to allow it to over-ride the association between 'day=Monday' and 'service=news'.

6 CONCLUSION

There is no doubt that personalisation mechanisms can greatly enhance user experience in a pervasive environment. They unburden the user of resource management tasks, adapting the environment to meet the current needs of the user. However, to provide accurate personalisation the system must maintain an accurate set of user preferences on which it can base decisions. Without such information it is impossible to know what actions will help rather than hinder the user.

There are two main approaches to the management of preferences. Explicit personalisation places the onus on the user to manually maintain their preference set through some GUI. Although this places the user in complete control, the burden of maintaining such a large rule base is great. This undermines the benefits of personalisation and can often lead to a sparse preference set and hence inaccurate personalisation. At the other extreme, implicit personalisation uses monitoring and learning techniques to maintain a preference set on behalf of the user. The benefit of such an approach is the minimal burden on the user however care must be taken to provide some method of user control. Without such functionality the user cannot alter system behaviour to reflect new situations or behaviours in a rapid way.

Therefore, the most successful systems take a hybrid approach providing implicit personalisation where possible but also providing a GUI through which the user can manually manipulate their preferences and take final control. The learning approaches employed by such systems often fall under two types; Rule-based learning algorithms (which store preferences as rules) and network algorithms (which store preferences in some network structure). Rule-based learning algorithms have the advantage that their output is easily translated into human-readable form allowing their knowledge to be understood. However for many problems they lack the success and flexibility of network approaches. The internal complexity of networks means they are very efficient learning tools for some problems but also means that often humans must accept their output without understanding why the network reached this conclusion.

The Persist project aims to gain the best of both worlds by utilising the Daidalos IF-THEN-ELSE rule-based learning techniques as well as introducing a more complex neural network learning approach that can be translated into IF-THEN-ELSE rule format for human understanding. This paper illustrated how the Persist personalisation system will accomplish this task. The Persist personalisation system will be fully demonstrated in 2010.

7 ACKNOWLEDGEMENT

This work was supported by the European Union under the FP7 programme (Persist project) which the authors gratefully acknowledge. The authors also wish to thank all colleagues in the Persist project developing the pervasive system. However, it should be noted that this paper expresses the authors' personal views, which are not necessarily those of the Persist consortium. Apart from funding the Persist project, the European Commission has no responsibility for the content of this paper.

REFERENCES

- [1] WEISER, M., The Computer for the 21st Century, Scientific America, 256(3), pp. 94-104, 1991.
- [2] YOSHIHAMA, S., Chou, P., Wong, D., Managing Behaviour of Intelligent Environments, Proceedings of the First IEEE International Conference on Pervasive Computing and Communications (PerCom '03), pp. 330-337, 2003.
- [3] LESSER, V., Atighetchi, M., Benyo, B., Horling, B., Raja, A., Vincent, R., Wagner, T., Xuan, P., Zhang, S. XQ., The Intelligent Home Testbed, Proceedings of the Anatomy Control Software Workshop (Autonomous Agent Workshop), pp. 291-298, 1999.
- [4] SOUSA, J.P., Poladian, V., Garlan, D., Schmerl, B., Shaw, M., Task-based Adaptation for Ubiquitous Computing, IEEE Transactions on Systems, Man and Cybernetics, Part C: Applications and Reviews, Special Issue on Engineering Autonomic Systems, 36(3), pp. 328-340, 2006.
- [5] ZIEBART, B.D., Roth, D., Campbell, R.H., Dey, A.K., Learning Automation Policies for Pervasive Computing Environments, Proceedings of the 2nd International Conference on Autonomic Computing (ICAC '05), pp. 193-203, 2005.
- [6] SI, H., Kawahara, Y., Morikawa, H., Aoyama, T., A Stochastic Approach for Creating Context-Aware Services based on Context Histories in Smart Home, ECHISE2005, Pervasive 2005, pp. 37-41, 2005.
- [7] CORDIER, C., Carrez, F., Van Kanenburgh, H., Licciardi, C., Van der Meer, J., Spedalieri, A., Le Rouzic, J.P., Zoric, J., Addressing the Challenges of Beyond 3G Service Delivery: the SPICE Service Platform, Workshop on Applications and Services in Wireless Networks (ASWN '06), 2006.
- [8] MCBURNEY, S., Papadopoulou, E., Taylor, N., Williams, H., Adapting Pervasive Environments through Machine Learning and Dynamic Personalisation, Proceedings of the International Conference on Intelligent Pervasive Computing (IPC '08), pp. 395-402, 2008.
- [9] CROTTY, M., Taylor, N., Williams, H., Frank, K., Roussaki, I., Roddy, M., A Pervasive Environment Based on Personal Self-Improving Smart Spaces, Workshop on Architectures and Platforms for Aml at the European Conference on Ambient Intelligence 2008 (Aml 08), 2008.
- [10] TAYLOR, N., Personal space and Personal Smart Spaces, 1st PerAda Workshop on Pervasive Adaptation at the 2nd IEEE International Conference on Self-Adaptive and Self-Organizing Systems (SASO 2008), 2008.
- [11] MCBURNEY, S., Papadopoulou, E., Taylor, N., Williams, H., Managing User Preferences for Personalisation in a Pervasive Service Environment, Proceedings of the 3rd Advanced International Conference on Telecommunications (AICT '07), pp. 32-37, 2007.
- [12] QUINLAN, J.R., C4.5: Programs for Machine Learning, Morgan Kaufman, 1993.