Terms and Conditions of Use of Digitised Theses from Trinity College Library Dublin

Copyright statement

All material supplied by Trinity College Library is protected by copyright (under the Copyright and Related Rights Act, 2000 as amended) and other relevant Intellectual Property Rights. By accessing and using a Digitised Thesis from Trinity College Library you acknowledge that all Intellectual Property Rights in any Works supplied are the sole and exclusive property of the copyright and/or other IPR holder. Specific copyright holders may not be explicitly identified. Use of materials from other sources within a thesis should not be construed as a claim over them.

A non-exclusive, non-transferable licence is hereby granted to those using or reproducing, in whole or in part, the material for valid purposes, providing the copyright owners are acknowledged using the normal conventions. Where specific permission to use material is required, this is identified and such permission must be sought from the copyright holder or agency cited.

Liability statement

By using a Digitised Thesis, I accept that Trinity College Dublin bears no legal responsibility for the accuracy, legality or comprehensiveness of materials contained within the thesis, and that Trinity College Dublin accepts no liability for indirect, consequential, or incidental, damages or losses arising from use of the thesis for whatever reason. Information located in a thesis may be subject to specific use constraints, details of which may not be explicitly described. It is the responsibility of potential and actual users to be aware of such constraints and to abide by them. By making use of material from a digitised thesis, you accept these copyright and disclaimer provisions. Where it is brought to the attention of Trinity College Library that there may be a breach of copyright or other restraint, it is the policy to withdraw or take down access to a thesis while the issue is being resolved.

Access Agreement

By using a Digitised Thesis from Trinity College Library you are bound by the following Terms & Conditions. Please read them carefully.

I have read and I understand the following statement: All material supplied via a Digitised Thesis from Trinity College Library is protected by copyright and other intellectual property rights, and duplication or sale of all or part of any of a thesis is not permitted, except that material may be duplicated by you for your research use or for educational purposes in electronic or print form providing the copyright owners are acknowledged using the normal conventions. You must obtain permission for any other use. Electronic or print copies may not be offered, whether for sale or otherwise to anyone. This copy has been supplied on the understanding that it is copyright material and that no quotation from the thesis may be published without proper acknowledgement.

Improving Pervasive Application Behavior with Other Users' Information

Michael Spence

A thesis submitted to the University of Dublin, Trinity College

in fulfillment of the requirements for the degree of

Doctor of Philosophy (Computer Science)

March 2012

Declaration

I declare that this thesis has not been submitted as an exercise for a degree at this or any other university and is entirely my own work.

I agree to deposit this thesis in the University's open access institutional repository or allow the library to do so on my behalf, subject to Irish Copyright Legislation and Trinity College Library conditions of use and acknowledgement.

Summary

The behavior of a pervasive application is much improved with access to accurate, relevant information describing the situation of the application's user. Unfortunately, sensor failure, sensor drift, and device mobility mean that the primary sources of information are not guaranteed to always be available or applicable. A promising alternative source is the information describing other users. The use of information from other users has the potential to allow a pervasive application to utilize the knowledge, experience, reasoning methods, and sensors of other users' devices in the environment. Despite its potential benefits, there are two reasons why a convincing case has, to date, not been presented for the utilization of other users' information to improve accuracy in pervasive applications.

The first reason is that there have been no compelling studies with sufficiently valid evaluations exploring the utility of other users' information in pervasive environments. This is mainly a consequence of the lack of multiuser datasets on which to perform comparative evaluations which is due to the prohibitive costs of multiuser, multi-device deployments. Randomly generating the information describing the users in an environment is not suitable for testing the utility of other users' information as randomly generated data will not produce general, realistic relationships between different types of information, e.g., temperature is highly dependent on location. Additionally, random data generation does not realistically model the uncertainty inherent in the information or how information describing a user changes over time.

The second reason a convincing case has not been presented is that current methods for accurately acquiring other users' information are not suited for general use in pervasive environments. The current methods acquire information according to which values best satisfy a set of parameters such as precision, trustworthiness, and timeliness. Most critically, selection methods assume that the information describes the

application's user and is relevant to its current situation. This is not always the case, meaning the accuracy of the information is potentially negatively affected. Outside the case of considering other users' information, research in artificial intelligence has introduced several learning methods such as naive Bayes classifiers and case-based reasoning (CBR) approaches that can be used as an aid to identifying situational relevance. While used in many pervasive applications, such learning methods are trained mostly from the previous experience of the user. Additionally, the accuracy of selection and learning methods using other users' information suffers in the presence of sparse previous experience and the sparse environments common to pervasive applications. The research question addressed in this thesis is whether a convincing case can be presented that the inclusion of other users' information improves the overall accuracy of information describing the situation of an application's user.

This thesis presents a study that explores the suitability of using learning methods applied to other users' information to acquire accurate information. Comparative experiments on existing selection methods and the learning methods of CBR, naive Bayes classifiers, support vector machines (SVM), and a boosting ensemble method applied to the information of other users and the previous experience of the application's user are performed. In order to significantly counteract the limitations of randomly generated simulations, a dataset conversion procedure has been developed that uses existing single-user benchmark datasets taken from actual pervasive environments to generate a more realistic multiuser dataset.

The evaluations show that other users' information is often useful for increasing the overall accuracy in pervasive applications. Learning methods outperform existing selection methods when acquiring information from other users. Results show that using other users' information is not always more accurate than using previous experience but it is more accurate in some cases. In addition, combining information from both previous experience and other users aids learning methods when one of these sources is sparse. For the CBR and SVM approaches, augmenting the previous experience with other users' information proved to be the approach with the most overall accuracy in full and sparse environments for all the datasets. For the naive Bayes classifier and boosting algorithm, the most accurate source was dataset-specific.

Acknowledgements

First of all, I would like to thank my supervisor, Dr. Siobhán Clarke, for her encouragement, advice, and patience. Due to her influence, I am now not only a researcher but I'm also a much better writer, presenter, and lecturer than I would have been without her.

I would also like to thank all of my colleagues in the Distributed Systems Group. It has been my pleasure to spend time with some of the smartest and most generous people I've ever met. Particularly, I'd like to thank the members of the Hermes team: Éamonn Linehan, Cormac Driver, Shiu Lun Tsang, and Andrew Jackson. The feedback and support I've received from them over the years has been invaluable. I cannot imagine a better group of colleagues to have had on my PhD journey and consider them all friends.

Outside of DSG, I would also like to extend my gratitude to my colleagues in the statistics wing of the building: Éamonn Mullins, Simon Wilson, and especially Brett Houlding. This study would not have been possible without their generosity and guidance with my thornier analysis problems.

I would also like to recognize the indispensable feedback from my examiners. This thesis has been very much improved as a result of their insightful comments and suggestions.

Finally, I would like to extend my love and gratitude to all of my friends and family. They have all had to listen to my worrying and griping, and they have done so with love and without complaint. Special thanks go to Paul Marrinan for acting as the Dublin branch of my family and for giving me a non-computer scientist's view on this thesis. Most of all, I would like to thank my grandparents for their unwavering support, without whom I would never have been able to study in Dublin or have even dreamt that I could, and my mother who has selflessly given me more of her life and love than

I could ever begin to thank her for.

Michael Spence
University of Dublin, Trinity College
February 2012

Publications Related to this Ph.D.

Mike Spence and Siobhán Clarke. Improving Pervasive Application Behavior Using Other Users' Information. In *Proceedings of the 18th International Conference on Case-Based Reasoning, ICCBR 2010*, Alessandria, Italy, July 2010. Springer-Verlag.

Cormac Driver, Éamonn Linehan, **Mike Spence**, Shiu Lun Tsang, Laura Chan, and Siobhán Clarke. Facilitating Dynamic Schedules for Healthcare Professionals. In *Proceedings of the 1st International Conference on Pervasive Computing Technologies for Healthcare*, Innsbruck, Austria, 2006. IEEE.

Mike Spence and Siobhán Clarke. Preserving Context with Context Capsules. In *Modeling and Retrieval of Context, Third International Workshop, MRC 2006, AAAI 2006*, Boston, MA, July 2006.

Mike Spence, Cormac Driver, and Siobhán Clarke. Sharing Context History in Mobile, Context-Aware Trails-Based Applications. In *1st International Workshop on Exploiting Context Histories in Smart Environments, ECHISE 2005, Pervasive 2005,* Munich, Germany, 2005.

Cormac Driver, Éamonn Linehan, Siobhán Clarke, Andrew Jackson, Shiu Lun Tsang, and **Mike Spence**. A Framework for Mobile, Context-Aware Trails-based Applications: Experiences with an Application-led Approach. In *Workshop 1 - "What Makes for Good Application-led Research in Ubiquitous Computing?", PERVASIVE 2005*, Munich, Germany, 2005.

Mike Spence, Cormac Driver, and Siobhán Clarke. Collaborative Context in Mobile, Context-Aware Trails-Based Applications. In *3rd UK-UbiNet Workshop*, University of Bath, UK, 2005.

Contents

List of Tables

List of Figures

Chapter 1

Introduction

The behavior of a pervasive application is much improved by more accurately modeling the application user's situation. A promising source of accurate information is that describing the situation of other users in the environment as the devices of other users may have sensors or knowledge not available to the application's user. Unfortunately, a convincing case for utilizing other users' information has not been presented by existing research for two reasons. First, the evaluations described in the related works have limited validity, with the most important threat to validity being that the existing pervasive datasets are not suitable for examining the utility of other users' information. Second, the assumptions of current methods for acquiring context information from other sources are not suitable for pervasive applications. Most importantly, they assume that the information about other users describes the situation of the application's user without consideration of the actual relevance of that information. This thesis presents a study that examines the utility of other users' information using learning methods that aid in identifying the relevance of that information. In order to significantly counteract the threats to validity of existing evaluations, the study also utilizes a conversion procedure that uses existing single-user datasets recorded from actual pervasive environments to generate multiuser datasets. This chapter provides the background and motivation of this work, introduces the study, lists the work's contributions, and provides an outline for the rest of the thesis.

1.1 Background

This section presents the background information necessary to understand the motivation for using learning methods to help acquire information in pervasive applications and discusses the validity of evaluations and simulations in multiuser environments. First, pervasive applications and the multiuser environments in which they operate are presented through discussion and examples. Additionally, the importance of acquiring accurate information for these applications is discussed along with how a user's situation is modeled. Second, learning methods that are used to automatically reason about a domain are introduced. This includes a discussion of why applications should be knowledge autonomous and how this is achieved via machine learning. Third, evaluation validity is introduced with an emphasis on the importance of realistic multiuser datasets to validity. Despite this importance, the prominence of simulations in the evaluations of applications in multiuser environments is noted and their effects on validity are discussed.

1.1.1 Pervasive Applications

Pervasive applications run on devices that travel with the user or that are embedded in the environment [142]. The environments in which pervasive applications operate are constantly changing due to device mobility and evolving situations [10]. Appropriate application behavior depends greatly on the application user's current situation [119]. The types of behavior that pervasive applications provide are presentation of information and services to a user, automatic execution of a service for a user, and tagging of metadata describing the situation to information for the support of retrieval at a subsequent point in time [34]. Dynamic, pervasive applications must be aware of the environment and situation in which they operate in order to identify appropriate behavior. This capability is termed context-awareness, where *context* is defined as any information that can be used to characterize the situation of an entity [34]. Context information is acquired through sensors such as GPS, thermometers, accelerometers, and light sensors, or is derived from sensor data and knowledge using reasoning methods to attain higher-level context information such as user activity and situation. There can be many types of context attributes, e.g., location, temperature, speed, activity,

and preferences.

The physical environment in which a pervasive application operates is called a *pervasive environment.* These environments contain a number of other users with their own devices each with their own unique capabilities, e.g., sensors, experience, and reasoning methods. Pervasive environments can contain a great number of users, e.g., a sporting event, or can be very sparse with just a few or no other users, e.g., hiking alone on an isolated trail. There can be several sources of context information, both on the local device and in the environment. Local sensors, context information from other users, remote services, and previous experience of the user are all examples of context information sources. The previous experience of the application's user (also called "context history" [19]) reflects the recorded representations of situations experienced by the user and the set of recorded situations have a varying level of sparseness depending on the recording policies, the diversity of the user's previous experience, and the amount of time that the application has been recording its experience.

There are several ways to model a user's situation with context information, e.g., object oriented models, ontology based models, and key-value models [131]. A situation can be modeled most simply as a vector of attribute-value pairs (i.e., a key-value model) that each represent a different type of context information and the associated value of each type [119]. This representation is chosen in this thesis for its simplicity of explanation, the fact that it is the representation of the pervasive datasets for the evaluations that are surveyed in Section 3.2.2, and because all knowledge required for this representation is contained within the datasets, e.g., no extra knowledge for named or inheritance relationships between variables is required. This thesis defines the vector of context information as a *situation instance* (it is also known as a *context description* [152] and a *feature vector* [68]), and it represents a snapshot of the user's current situation.

The more accurate and relevant the context information is, the closer the application's situation can be modeled and, in turn, the more appropriate the application's behavior can be [34]. A scenario involving a commuting user (as in the work of Flanagan et al. [41]) shows an example of context information use in a pervasive environment. The user is currently riding a bus and is carrying a device which stores and updates its context information via sensors or remote services. The user's current situation is

modeled as a situation instance, i.e., a vector of attribute-value pairs containing both static and dynamic context information such as: the user's age is 32 and the current ambient temperature is 10 degrees Celsius. There are also a number of other users in the environment, e.g., other bus passengers, drivers of adjacent automobiles, and employees in nearby office buildings, all represented by their own situation instances on their devices that describe their unique situations. One particular application on the user's device adjusts the volume of an aural notification (the behavior) when an incoming message is received according to the amount of ambient noise and current location (the context information) [88]. The consequences of inaccurately modeling the ambient noise for the user are missing a message if the notification is too quiet for the user to hear or potential embarrassment if the notification is at an inappropriately loud volume for the situation.

1.1.2 Learning Methods

Reasoning methods perform a mapping from inputs to outputs. For example, if an accelerometer on the user shows a great deal of activity and the user's location is a park, these methods can infer that the user is jogging. Outputs are a category for classification, e.g., the activity is jogging, or a numeric value for regression, e.g., the accelerometer reading is -.06815 [7].

Reasoning methods in pervasive applications require knowledge of the environment in which they operate, e.g., that the inputs above actually entail that the user's activity is most probably jogging. *Knowledge* is defined as any information that is useful for a particular task [3]. It is difficult to obtain for many domains and the trouble doing so is termed the knowledge acquisition bottleneck [138]. Even if knowledge is acquired, it may become obsolete in the presence of changing environments due to concept drift. *Concept drift* is when a change in the hidden context changes the target concepts [144]. For example, the concept of "bad weather" will be different in the frozen tundra of Siberia than in the Sahara desert. If an application or method is able to automatically acquire knowledge with which to more appropriately adapt its behavior without requiring the user to manually supply that knowledge, the method or application is called *knowledge autonomous* [94]. Knowledge autonomy is desirable as it minimizes the elicitation interaction with the user causing less distraction and

requiring less knowledge about the environment from the user.

Machine learning methods allow an application to be more knowledge autonomous by automatically extracting the reasoning knowledge from available data. *Learning methods* allow the performance of a computer program to automatically improve with experience [87]. These methods have both *learning algorithms* and *models* that those algorithms assume and train [7]. For example, a decision tree learning method uses learning algorithms such as entropy-based attribute splitting and pruning to train their model, i.e., a decision tree (see Section 3.3.2).

Machine learning methods are used to automatically detect patterns in a problem domain using a set of data instances from that domain. This set is called a *training set.* The patterns are then employed to understand aspects of the domain or to make predictions [7]. These methods can be divided into supervised and unsupervised learning. *Supervised learning* derives the mapping from an input to an output given a set of examples with an input and its corresponding output, i.e. the training set. *Unsupervised learning* is learning that has no corresponding output data, and its purpose is to find regularities in the input rather than a mapping from inputs to outputs.

The discipline of artificial intelligence (AI) has produced several supervised and unsupervised learning methods. Clustering and dimensionality reduction are examples of unsupervised learning methods. Decision trees, case-based reasoning, and Bayesian methods are all examples of supervised learning methods. These methods are divided into lazy and eager learners. *Lazy (or instance-based) learners* defer the decision of how to generalize the data until the problem is posed and can therefore use the problem to determine how to generalize the data [87]. *Eager learners* generalize the data before the problem is posed and require less computation time to solve the problem.

If multiple training sets exist, automatic selection of the set that is most appropriate for the current situation requires extra knowledge. Choosing the most appropriate training set(s) is termed *instance set selection* (it is also called *case* [71] or *problem* [73] *dispatching* in the learning methods of case-based reasoning). Instance set selection requires the knowledge of which sets are the most appropriate to train on for the current problems the learning method is trying to solve. This knowledge requirement negatively affects the knowledge autonomy of an application.

If a training set does not exist, one must be collected or elicited. Some applications

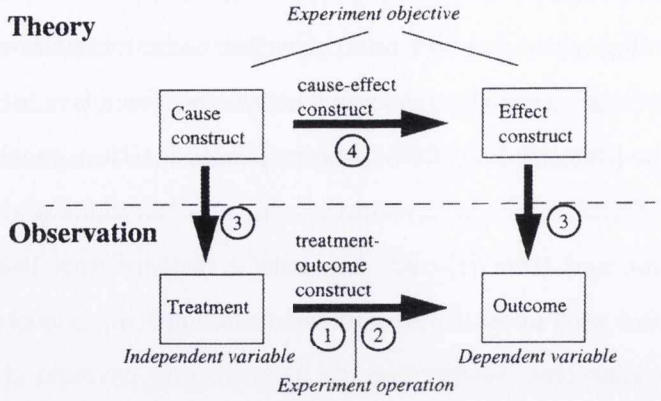

Figure 1.1: Experiment principles from [145].

cannot function appropriately without an initial set of training instances, e.g., the application presented by McGinty and Smyth [83]. Creating instances on which to train the learning method is problematic as it is a manual process [64, 152]. The more user interaction required for learning, the less knowledge autonomous the method becomes.

1.1.3 Evaluation Validity and Simulating Multiuser Environments

A *valid evaluation* is one whose results and analysis have the best available approximation to the truth about the proposition being evaluated [31]. The word "approximation" is used with the definition of validity as one can never know what is actually true. The four types of validity are [31, 145]:

1. *Conclusion validity*: Also called statistical conclusion validity, it concerns the covariation of the dependent and independent variables. The different values an independent variable can take on are called *alternatives [61]* (also called *treatments* [145]). Statistical tests are used to determine if the differences between alternatives are most likely a cause of the alternatives themselves and not the cause of some other variation. This validity of the conclusions about the differences between alternatives is maximized by the application of appropriate statistical methods to identify the statistical significance of the differences (see Section 4.3).

2. *Internal validity*: Internal validity refers to the approximate validity that the relationship between two variables is causal, i.e., that change in the independent variable is the cause of change in the dependent variable.

3. *Construct validity*: Construct validity is maximized when the variables in the experiment correspond to the theoretical concepts under examination. That is, the independent variable reflects the construct of cause and the dependent variable reflects the construct of effect (see Section 4.2).

4. *External validity*: External validity reflects how well the outcome of the evaluation generalizes outside the scope of the study. This is maximized when the experiment subjects, setting, and timing are indicative of the population and setting under consideration. Questions of external validity are only interesting when the results are internally valid [51].

The relationship among the types of validity is shown in Figure 1.1. Construct validity (3) involves the mapping from theory to observation and external validity (4) determines how applicable the results of the experiment are to the general population under study. Internal (2) and conclusion (1) validity determine the causal nature and statistical significance, respectively, of the experiment results.

The thesis sections dealing with conclusion and construct validity are listed in the descriptions above. Internal validity is not discussed in much detail as it is quite good in experiment environments where there is a high degree of control over the environment conditions as is the case with many evaluations in computer science. External validity is a major concern to the evaluations and is a prominent thread that runs throughout the remainder of this thesis.

To aid external validity, the alternatives of the evaluation should be run on a dataset that realistically reflects the situation under examination. The field of AI has several of these benchmark datasets on which competing learning methods can be run and their performance analyzed. For example, the UCI Machine Learning Repository [8] maintains almost 200 datasets across several domains on which different approaches can be evaluated.

For some newer domains, e.g., pervasive environments, collections of realistic benchmark datasets are not yet available. Pervasive applications operate in multiuser en-

vironments which are ones in which a number of users and their respective devices operate in the same environment at the same time. A *multiuser dataset* is a dataset that is recorded in a multiuser environment.

As the costs of producing multiuser datasets are high, a secondary option is to simulate the data on which to evaluate. *Simulations* use mathematical models to imitate the system under consideration [69]. A simulation can be used to more easily control the experiment conditions and to study a system for a large number of runs and over a long period of time. It can be the only option in evaluating large-scale, multi-device pervasive applications if deploying the application for testing is too costly [113, 56].

However, there are external validity problems with randomly generating the information describing the users in a multiuser environment. Firstly, simulations will not produce general, realistic relationships between different types of information. For example, temperature, humidity, and GPS signal strength are highly dependent on location. Additionally, heterogeneous user preferences are complex to model realistically. For example, users may prefer different types of transportation depending on temperature and precipitation, and not all users will have the same preferences given the same environment conditions.

Secondly, a simulation that randomly generates information does not realistically model how an attribute describing the situation changes over time. This is particularly important for evaluating how an approach compares to one using the previous experience of a user. If the subsequent situations are not realistic given the situations experienced previously, the external validity of the tests becomes questionable. For example, over a set period of time it is not possible for a user to travel distances longer than that which the mode of transport can support or for the ambient temperature of a room to suddenly change by an impractically large magnitude.

Thirdly, context information has a great deal of uncertainty that must be modeled [52, 11]. Sensors are inherently imprecise so value output from sensors exhibit uncertainty. In addition, sensors can degrade over time (i.e., sensor drift) and manual feedback from users is not always reliable even if they are willing and able to provide it as the users can make mistakes. Realistically modeling this uncertainty is essential to external validity as reasoning is done on these uncertain observations and not on

the true values in multiuser environments.

1.2 Motivation

Context information that more accurately describes the situation of the application's user improves pervasive application behavior [34]. Acquiring accurate context information, while essential to operation, can be difficult in pervasive environments as pre-assigned sources of context information cannot always be relied upon, e.g., if the audio sensor fails or is otherwise obstructed [11]. Sensors often degrade or fail entirely. Additionally, changing situations cause sources of context information that are useful in some situations to be ineffective in others, e.g., attempting to use a GPS sensor while indoors or reasoning about audio notifications at a loud sporting event.

Deriving context attributes from other available attributes can help accuracy in context acquisition, but even this reasoning can become obsolete due to concept drift (see Section 1.1.2). For example, proximity-based methods assume that devices with which they can communicate share the same location and situation. This sort of reasoning would not be appropriate if that device connects to the Internet, e.g., the device it is communicating with could literally be on the other side of the globe. The hidden attribute causing concept drift in this example is the distance between the two devices.

To meet the challenges of changing pervasive environments, a pervasive application must use general methods of context acquisition that adapt to the current environment and situation. It does so by gathering knowledge about the environment in which the context acquisition takes place. This knowledge can be elicited manually from the application's user, but this limits the knowledge autonomy of the application. The amount of time and expertise required of the user for customizing the learning methods can be significant, making it difficult or impossible to accomplish [53]. Additionally, even if the user is willing and capable of manually providing appropriate feedback for the current situation, this is not always possible in pervasive applications, e.g., distraction may be dangerous while the user is mobile.

Another method of learning context acquisition knowledge is to obtain it from

sources in the environment. One of the sources is the previous experience (or history) of the application's user. However, in unfamiliar situations previous experience is not as useful because the user's device does not have any experience in the situation from which to extract the necessary knowledge. In addition, in the case of concept drift the previous experience will be obsolete when attempting to obtain knowledge about the new concept.

In the case of changing and unfamiliar situations, a promising source of both context information and knowledge about context acquisition is that describing other users on devices in the application user's surroundings. Collectively, other users in the environment can be a rich source of context information for an application as the devices of other users have access to different knowledge, experience, reasoning methods, and sensors [83, 68]. The context information of other users' devices is particularly relevant when considering that many activities are done with other users who are in similar situations, e.g., doing an activity with friends and family, collaborating with coworkers, or travelling along the same route as strangers.

Despite these potential benefits, there are two main reasons that a convincing case has, to date, not been presented for the utilization of other users' information to improve accuracy in pervasive applications. The first is that the assumptions of existing methods are not suitable for general context acquisition from other users in pervasive environments, and the second is that there are no valid studies to show the utility of other users' information.

1.2.1 Existing Methods Are Not Suited for Pervasive Environments

The first reason a convincing case has not been presented for the use of other users' information is that current methods for accurately acquiring such information are not suited for general use in pervasive environments. These methods are called context selection methods. *Context selection methods* choose context values according to which ones best satisfy a given set of parameters such as precision, trustworthiness, and timeliness [26, 123, 111]. The individual degrees of satisfaction for each parameter are then combined using a utility function to reflect the overall satisfaction of each context

value and the context value with the highest utility is then selected for use.

Beyond timeliness, i.e., a more recent value is considered more favorable than a less recent one, the selection methods do not consider that the information might not describe the current situation of the application's user. The selection methods' assumption that all other users are in the same situation as the application's user is false and leads to the selection of highly inaccurate context information. For example, assuming that the application user's destination is the same as another driver's only because they are in communication range will result in undesirable behavior for a navigation system. Existing selection methods are missing an ability to determine how relevant a piece of context information is to the current situation. This thesis defines this property as *situational relevance*, and it can be estimated by determining the relationships among the attributes. In the above example, communication range has poor correlation to the destination of two drivers whereas being in the same automobile is highly correlated with having the same destination.

Outside the case of considering other users' information in pervasive environments, research in artificial intelligence has introduced several learning methods that can be used to estimate situational relevance. Given both the nature of the problem, i.e., acquiring an output value that is a specific attribute type, and the sources that are available, i.e., situation instances where the desired output attribute type has a value, supervised learning methods such as naive Bayes classifiers [37] and case-based reasoning (CBR) approaches [2, 33] can be used as an aid to identifying situational relevance. Depending on the specific learning method, situational relevance is provided by either reusing the output value of the most similar situation instance(s) in the training set (e.g., the lazy learning techniques of CBR) or generalizing the training instances into a specific set of rules or probabilities to predict the most appropriate value for the given situation.

While used in many pervasive applications, such methods learn mostly from the previous experience of the user, e.g., [137, 62, 92]. Learning from previous experience is very effective in many static domains, but is less useful for the changing and possibly unfamiliar situations encountered by a pervasive application. That is, if the model is learned from one situation, it may not generalize well to unfamiliar situations or familiar situations that have experienced concept drift. An additional problem with

learning from previous experience is encountered when the desired attribute type is one with which the user is completely unfamiliar.

The accuracy of methods using both previous experience and other users' information also suffers in the face of instance set sparseness. *Instance set sparseness* (also called "limited competence" [71]) is when the training set of a learning method has a small number of instances. This occurs in the sparse environments common to pervasive applications and at the initial stages of application deployment when the previous experience of the user is not extensive. In both of these circumstances, there is limited problem-space coverage as there are not enough instances available from which to select or to properly train learning methods.

While the context values that make up the situation are sensed automatically, the output value must be assigned by another means (otherwise the reasoning or sensing of the desired output value would already be complete, precluding the need for a learning method). The user is a possible source from which to elicit knowledge in the face of instance set sparseness, but even if the user is able and willing to give this feedback the knowledge autonomy of the application is decreased.

1.2.2 No Valid Studies Exist

The second reason that a convincing case has yet to be made is that there are no valid studies to show the utility of other users' information. As discussed in Section 1.1.3, a valid study compares experiment alternatives in a manner that maximizes all types of validity for the evaluation results. The chief reason that there are no valid studies is that there are no multiuser datasets on which to perform comparative evaluations that can address the utility of other users' information in pervasive environments. Multiuser datasets are important as an environment with multiple users is necessary in order to be able to evaluate the utility of other users' information. The lack of these datasets is mainly due to the costs of multiuser, multi-device deployments and as a result the current best datasets come from simulations with data randomly generated from probability distributions, e.g., [113, 56, 83, 11].

However, these simulations are subject to the external validity threats presented in Section 1.1.3. That is, simulations of multiuser environments have questionable validity when approximating relationships between context attributes, representing the manner

in which the attributes change over time, and modeling the uncertainty inherent in the measurements of sensors. Without realistic relationships between values of different types, it is difficult to use one type of attribute to realistically predict another. In addition to the threat to validity that simulations entail, the evaluations of works that are related to utilizing other users' information do not maximize the validity of their evaluations. Most limit their conclusion validity by omitting any reporting of statistical analysis for their results, many limit their generalizability by focusing their evaluation on only one situation, and some of the experiments have considerable bias (see Chapter 2).

1.2.3 Research Questions

To deal with the main problem preventing valid studies (Section 1.2.2), the question is asked if there is a procedure to generate multiuser datasets in order to allow evaluations of the utility of other users' information to be performed with good external validity (RQ-1). To address the problems with the suitability of existing methods (Section 1.2.1), it is asked in this thesis whether using learning methods to acquire other users' information will improve the overall accuracy of context acquisition when compared to existing context selection methods (RQ-2). In addition, it is asked which source of context information, i.e., previous experience, other users, or both sets together, can most improve the overall accuracy of context acquisition in both full (RQ-3 to RQ-5) and sparse (RQ-6) environments. Given the problems described in the motivation, the following research questions must be addressed to determine whether a convincing case can be presented for the utility of other users' information in a pervasive environment:

RQ-1 Multiuser dataset procedure: Is there a procedure for generating multiuser datasets for evaluations involving other users' information that realistically models the relationships between different types of context information describing a user's situation, the uncertainty inherent in the information, and the manner in which that information changes over time?

RQ-2 Situational relevance: Can machine learning methods using situational relevance to acquire context information from other users perform with more overall accuracy than existing context selection methods that do not employ situational

relevance?

RQ-3 Previous experience versus other users: Does context acquisition using machine learning methods trained only from the information of other users perform with more overall accuracy than the existing approach of training them from previous experience?

RQ-4 Combining instance sets: Is there an improvement in overall accuracy when utilizing the previous experience of the user in addition to the context information of other users?

RQ-5 Instance set selection: What is the approach for instance set selection in pervasive environments that exhibits the most overall accuracy and how knowledge autonomous is this approach?

RQ-6 Instance set sparseness: What is the approach with the best overall accuracy for context acquisition involving sparse instance sets, i.e., when the previous experience of the user is small or nonexistent or when the environment contains few or zero other users?

1.3 A Study of Utilizing Other Users' Information in Pervasive Environments

This thesis presents a study that performs comparative evaluations of acquisition methods on generated multiuser datasets. The evaluations show the differing effects on accuracy of: i) acquiring other users' context information using learning methods and existing selection methods using different *situation instance source sets*, i.e., other users' information, previous experience, and both of these sets simultaneously and ii) using the different methods and instance sources in sparse environments. The generated multiuser datasets differ from randomly generated simulations by combining existing single-user benchmark datasets containing context information recorded from actual pervasive environments to generate multiuser datasets. This significantly counteracts the external validity limitations of randomly generated simulations by including

realistic relationships between types of information, by allowing the information describing a user to evolve over time in a realistic manner, and by realistically modeling the uncertainty of the context information. Each of the experiment alternatives are tested in a manner that maximize the validity of the evaluations.

The following sections introduce the application of learning methods to other users' information (Section 1.3.1), discuss how the multiuser datasets are generated and introduce some threats to the validity of that process (Section 1.3.2), and outline the overall validity of the evaluations of the study (Section 1.3.3).

1.3.1 Learning Methods and Other Users' Information

As mentioned in Section 1.2, existing selection methods lack a general means to estimate any sophisticated situational relevance, i.e., anything beyond timeliness. To aid situational relevance when acquiring information from the history of a user, learning methods in pervasive applications have been used on the user's previous experience. However, this approach is not appropriate in unfamiliar situations or in the presence of concept drift without elicitation from the user, which hinders knowledge autonomy.

This thesis proposes using learning methods for context acquisition and to train them on the current context information of other users. These learning methods satisfy situational relevance, as they consider the relationship of all the input attributes that make up the situation when estimating the output attribute. Unlike learning from the application user's previous experience, the methods learn from other users who potentially have knowledge about unfamiliar situations as they have values from sensors or experience in the environment that the application user's device does not. Diversity in the training set allows a learning method to have competence in a broader range of problems [71]. In addition, having the current information from other users rather than potentially stale previous experience allows for a window of examples representing the current situation which updates the concepts in the case of concept drift [144]. Existing context selection methods can be treated as learning methods, albeit ones that perform poorly in the problem domain.[1]

Knowledge autonomy is also improved as there is no need to manually assign val-

[1]Unless context selection methods are referenced specifically, any discussion of learning methods also includes the existing selection methods.

ues for the desired output attributes. This is because the other users from which the applications acquire context information already have values for the output attributes via sensors or reasoning, otherwise these users would not be queried. Knowledge autonomy is essential in pervasive environments where a user is often mobile or otherwise unable or unwilling to provide constant manual feedback.

To answer the research questions concerning instance set selection, three situation instance source sets are considered: that of the application user's previous experience, that of other users, and the union of both. Exactly one of these sets is used as the training set for each learning method to form the experiment alternatives of the evaluations. As the learning methods only require a set of training instances to learn, they can be used without modification to evaluate the comparative performance of the different sources of situation instances. The fixed nature of the methods improves internal validity as any variation is then due to the instance source variation.

Finally, the problem of instance set sparseness is dealt with in two ways. First, only learning methods that can be trained with a small training set size are selected for the evaluations, e.g., the naive Bayes classifier lessens the instance set size requirements of a regular Bayes classifier by its independence assumption [87]. Second, instead of just training from the sparse set, the union of both the previous experience and other users sets are used. It is hypothesized that such an augmentation of the sparse set will improve overall accuracy.

1.3.2 Generating Multiuser Datasets

For the study presented in this thesis, the multiuser datasets on which the evaluations are performed must exhibit a high level of external validity. There are no existing multiuser datasets with multiple context types on which the evaluations in this thesis may be run validly (see Section 3.2.2). As mentioned in Section 1.1.3, randomly generated simulations are not ideal for creating multiuser datasets that exhibit good external validity. There are, however, several single-user datasets, e.g., [89, 41, 60], that track how the context information of an actual user in a pervasive environment changes over time as that user performs a scenario. They provide the readings of a group of sensors representing a user's current situation instance. There are several of these situation instances each at fixed time intervals over the recording period, and the

entire set of situation instances for the duration is called a *run*. Each dataset has several runs from a single user or from multiple users recorded at different times. While these datasets do possess the required properties to aid external validity that simulations lack (see Section 1.1.3), they are not usable as is because they do not describe users acting in the environment simultaneously. That is, they are not multiuser datasets.

To obtain a multiuser, simultaneous dataset that enables a valid evaluation, this thesis presents a procedure for generating a multiuser dataset by combining the individual runs of a single-user dataset. The procedure assumes that each individual run is a different user and that they perform simultaneously in a single environment. This is accomplished by giving each user a randomized start time and from there using the same time intervals for recording each situation instance. For example, the situation of a user at the 20th time iteration is represented by the 20th situation instance after the assigned randomized start time in each corresponding run.

This procedure for generating a multiuser dataset combats the limitations of simulations listed in Section 1.1.3. This is ensured as the individual runs are taken from users recorded in actual pervasive environments. However, an actual multiuser pervasive environment is not exactly the same as one converted from single-user datasets. The three main threats to validity are briefly summarized as follows:

Shifting the Times of the Runs. Treating the runs that were recorded at different times as having been recorded at the same time.

Users Performing the Same Scenario. By placing repeated runs of the same scenario alongside the test scenario, this potentially ensures that there will be a similar user available from which to retrieve a similar value.

Treating Runs from the Same User As Runs from Different Users. Treating runs from a single user as runs from multiple users might create a bias in favor of other users' information, especially if the user performs tasks that are user-specific or if the attributes recorded are static or measure something personal to specific users.

A more detailed exploration of the threats and an analysis of why these have a less severe impact on validity for the evaluations in this thesis appear in Section 3.2.4.1 and Section 5.7.

1.3.3 Valid Study Evaluations

There are two evaluations of the study that address the remaining research questions. The first is a comparative evaluation that determines the most accurate methods and instance sources to use in a full environment (addressing RQ-2 to RQ-5), while the second evaluation measures their performance with sparse instance sets (addressing RQ-6). Sparse instance sets are simulated by using a subset of the number of other users or previous experience instances. The evaluation of instance set sparseness also shows the effects on overall accuracy of introducing training instances of just a small number of other users along with the previous experience of the application's user. The evaluation validity is maximized for all four types of validity.

The external validity of the evaluation results is greatly enhanced by the realistic properties of the generated multiuser dataset described in Section 1.3.2. It is further strengthened by the experiment design of the evaluations. Firstly, the experiments produce a large set of diverse values from which to compute the overall accuracy for each evaluation alternative. Along with testing each learning method using several different situation instances for each application user, the evaluation repeats this test for each user in the generated multiuser dataset using leave-one-out cross-validation [87]. These evaluations are also repeated for each attribute type and for many situation instances across each run for a range of time. Measuring the performance from a heterogeneous set of situations and attributes improves the external validity of the results as their applicability is not limited to just one situation or attribute type [145]. Secondly, instead of being influenced by the biases of a single dataset, the study in this thesis uses multiple benchmark datasets recorded in different problem domains to maximize the external validity of the evaluation results [85] (see Section 3.2.3).

Construct validity for the evaluation is good as the selected measures of independent and dependent variables reflect the theoretical concepts under study. That is, the alternatives reflect the constructs being compared in the research questions and the measure chosen in Section 4.2 reflects the changes in the theoretical construct "overall accuracy".

Internal validity of the evaluations is high as the experiments exhibit a high degree of control over the environment conditions and the process for applying them to each user is exactly the same. This implies causality between the alternatives and the results

as the alternatives are the only variations in the evaluations.

Finally, the analysis uses the statistical models and techniques for repeated measures evaluations increasing the conclusion validity. Both parametric tests (e.g., ANOVA [143]) and non-parametric tests (e.g., the Friedman test [30]) are used for analysis. Among other benefits, appropriate statistical analysis allows statements about the likelihood that the average overall accuracy of the differing alternatives are in fact different, i.e., the likelihood that the average performance of an alternative is better than another and that this difference is not just a product of the biases of the particular configuration of the input dataset (see Section 4.3).

1.4 Contributions

The main contributions of this thesis can be summarized as follows:

C-1 A procedure that aids evaluation validity by generating multiuser datasets: Combining the datasets of individual runs of single-user datasets to generate a multiuser dataset leads to better external validity than simulations. This combination allows for realistic relationships between types of context information, for the realistic modeling of context information uncertainty, and for the information describing a user to evolve over time in a realistic manner. (Addressing RQ-1)

C-2 Knowledge of the utility of learning methods with situational relevance: The evaluation compares the overall accuracy of the learning methods with situational relevance versus a context selection method for acquiring context information from other users. This answers whether learning methods with situational relevance exhibit better overall accuracy when utilizing other users' information. (Addressing RQ-2)

C-3 Knowledge of the utility of other users' information by itself: The evaluation compares the overall accuracy of learning methods trained from previous experience versus the same methods trained from the current situations of other users. This comparison aids in answering whether other users' information by itself is more useful than the existing approaches using only previous experience. (Addressing RQ-3)

C-4 Knowledge of the utility of augmenting the set of other users' information: The evaluation compares the overall accuracy of the methods using the union of the situation instance sets versus those using only the set of other users' information. This answers the question of whether there is any improvement in overall accuracy of using the knowledge contained in both instance sources simultaneously. (Addressing RQ-4)

C-5 Determination of the best approach for instance set selection: The result of this evaluation determines what sort of instance set selection knowledge is needed for each learning method, if any. That is, if one instance set is always more accurate than the other, there is no need for extra knowledge that would allow the automatic selection of the set with the most overall accuracy. (Addressing RQ-5)

C-6 Determination of the best approach for sparse instance sets: The evaluation compares differences in overall accuracy of learning methods on the different instance sources for sparse instance sets. These comparisons address which approaches are best in both sparse environments and when the extent of the user's previous experience is limited. (Addressing RQ-6)

1.5 Thesis Outline

The remainder of this thesis is organized as follows. An overview of the related work is presented in Chapter 2. The study inputs, i.e., the learning methods and the generated multiuser datasets on which they are trained, are presented in Chapter 3. The design of the experiments in the study including the dataset partitioning, the measure of overall accuracy, and the statistical methods used for analysis are discussed in Chapter 4. The evaluations of the study and an analysis of their results and validity is presented in Chapter 5. Finally, the conclusions are discussed and future work is suggested in Chapter 6.

Chapter 2

Related Work

The acquisition of accurate context information is a necessary requirement for appropriate behavior in pervasive applications. Using the context information of other users in the environment can be a great aid in acquiring accurate context. This chapter presents the state of the art on approaches that use the current context of other users to aid the acquisition of accurate context and also provides a broader survey of approaches to missing value prediction in general.

This chapter uses five criteria to measure both how well each approach to context acquisition deals with pervasive environments and how valid the results of their evaluations are. Each of the five *research question criteria* are associated with at least one research question (see Table 2.1). The degree to which the related works support each criterion determines the degree to which the research questions are addressed in the state of the art. In addition, the research question criteria are also used as a guide to select the study inputs (Chapter 3) and the evaluation and analysis methods (Chapter 4). The following describes the five research question criteria used to analyze each work and how they relate to each research question:

Instance set selection Defined in Section 1.1.2, instance set selection chooses the set of instances from which to acquire context information. This is particularly important in unfamiliar situations and in domains that experience concept drift where the selection of instance sets containing older instances will not cover recently introduced concepts. Instance set selection is addressed for both full (RQ-3 to RQ-5) and sparse (RQ-6) environments. It is explicitly addressed by RQ-5.

Research Question	Research Question Description	Associated Research Question Criteria
RQ-1	Multiuser dataset procedure	Validity (external)
RQ-2	Situational relevance	Situational relevance
RQ-3	Previous experience versus other users	Instance set selection
RQ-4	Combining instance sets	Instance set selection
RQ-5	Instance set selection	Instance set selection; Knowledge autonomy
RQ-6	Instance set sparseness	Instance set sparseness; Instance set selection; Knowledge autonomy

Table 2.1: The relationships between the research questions and the research question criteria.

Situational relevance Introduced in Section 1.2.1, situational relevance describes how relevant a context value is for the current situation of a user. This is especially important when the context information describes previous situations or other users. To maintain situational relevance, the learning methods should be tailored for the current user and situation. For example, in CBR approaches the relative importance of each type of context information should be up-to-date using appropriate weight learning methods. Learning methods that aid situational relevance are used throughout RQ-2 to RQ-6, and the utility of using learning methods that aid situational relevance versus existing methods that do not is explicitly addressed in RQ-2.

Instance set sparseness Presented in Section 1.2.1, instance set sparseness occurs when there are an insufficient number of instances from which to select or to use for training. For pervasive applications, instance set sparseness occurs when a user encounters a sparse environment. It also occurs when the user lacks extensive previous experience from which to draw, such as when the application is first deployed. Instance set sparseness is addressed in RQ-6.

Knowledge autonomy As established in Section 1.1.2, knowledge autonomy relates to the capability of an application to adapt to the current situation and function automatically without the need for intervention from the user. This is especially important in dynamic and evolving domains where new reasoning configurations must be learned. For this chapter, knowledge autonomy is specifically applied to the selection and automatic training of learning models (aiding RQ-2 to RQ-6), instance set selection (RQ-5), and the instance elicitation required to combat instance set sparseness (RQ-6).

Evaluation validity As described in Section 1.1.3, an evaluation is most valid when the internal, external, construct, and conclusion validity of an experiment are maximized. Validity is a criterion that is necessary to convincingly answer any of the research questions. For this chapter, the focus is on the generalizability and statistical rigor of the conclusions based on the results and the realistic nature of the datasets used in the evaluations. Specifically, this focus concerns external and conclusion validity.

The chapter is divided into two parts that reflect the different approaches to missing value prediction. The order of the sections in the first part of the chapter (Section 2.1 to Section 2.4) reflect the increasing degree to which the works of the respective sections handle overall situational relevance when performing context acquisition from remote sources. As much as possible, these initial works deal with or can be applied to pervasive computing applications that are people-centric and use the current information of other users to aid behavior. *People-centric* sensing is that which is performed by devices that are owned by the individuals rather than residing in a common administrative domain [68]. Section 2.1 presents a sampling of context selection methods that are common for acquiring context from remote devices in pervasive environments. Beyond consideration of timeliness, no relevance is considered in these works. Next, Section 2.2 introduces proximity-based acquisition methods that appropriate context information from users whose devices are closest to the application's device. Proximity-based acquisition increases the relevance of Section 2.1 by adding spatial situational relevance. The subsequent two sections attempt to deal with the situational relevance limitations of the first two sections via case-based reasoning (CBR) techniques. Section 2.3 details works that use CBR techniques in pervasive environments and Section 2.4 describes CBR applications in multi-agent systems that do not specifically operate in pervasive environments.

The second part of the chapter (Section 2.5 and Section 2.6) expands the scope to include related works on missing value prediction outside of pervasive computing and also relaxes the first parts' restrictions of applications that are people-centric, learn from current information, and use information of other users. For the these last two sections, the scope is broadened to include predicting user preferences with recommender systems (Section 2.5) and works performing missing value prediction in general (Section 2.6).

Finally, Section 2.7 presents a summary of the works in this chapter, focusing on the closest competitors to the approach laid out in this thesis. The section also lays out the limitations of the state of the art based on the five research question criteria and discusses how this thesis addresses these limitations.

2.1 Context Selection Methods

Context selection methods acquire values from a group of context sources according to a set of parameters. The aim of these parameters is to aid in acquiring the least ambiguous and uncertain context information available while also taking into account the application user's preferences for the retrieval itself. Almost all the selection methods cite Buchholz et al. [16] as a starting point for the parameters they consider. Buchholz et al. introduce the concept of Quality of Context (QoC) as the quality of the information that is used as context information, e.g., precision and trustworthiness of a context value. The authors introduce five Quality of Context parameters deemed most useful:

Precision Precision specifies how precise the value is, usually represented with bounds, e.g., the temperature is correct within a one degree interval.

Probability of correctness This is an estimate of the confidence in the correctness of the context information.

Trustworthiness Trustworthiness is the likelihood that the provider of the context information is supplying information with the properties that it is advertising. In contrast to probability of correctness, this estimation reflects the degree to which the provider of the information has been reliable in the past.

Resolution Resolution denotes the granularity of the information, i.e., the point at which a measurement cannot discern any further change. For example, room-based resolution for a temperature would not be able to detect changes in sections of the room, only the total room.

Up-to-dateness This is also referred to as timeliness, and it describes the age of the context information. In general, the assumption is that recent context values are more relevant to the application's situation.

There are several other QoC parameters and some are unique to certain types of context information [16]. The five parameters of Buchholz et al. are the most common across context selection methods, and any additional parameters will be discussed in the section of the work in which they are introduced. Quality of Service (QoS) parameters

define how well the services that supply the context perform logistically, e.g., the time to respond to a request and bandwidth to communicate. These parameters are particularly important when running on limited-resource devices or when frequent, timely updates are key to application performance. The set of parameters deemed relevant are application- and problem-specific. The degree to which each context value and context provider satisfy the totality of the parameters determines which value is chosen by context selection methods.

Three works that are representative examples of context selection methods are presented in this section. There are other works that explore context selection methods, e.g., Shi et al. [123] and Pawar and Khedr [102], but they do not add any more to the exploration of the utility of other users' information than the following three works already provide.

2.1.1 Huebscher and McCann

Inspired by the Context Toolkit [35], the adaptive middleware framework of Huebscher and McCann [57, 55] from Imperial College London focuses on aiding the development of pervasive applications in active spaces. Active spaces are physical spaces that have been enabled with an infrastructure for pervasive applications. The authors' work concentrates on supporting elderly people using smart-homes. The main goals of their framework are to provide these infrastructures for health-related applications with adaptive middleware and to provide that adaptation with good performance.

As an initial step towards these goals, their main focus is on the problems of selecting among different context sources with the same context type and adapting the application to a better source if the current one disappears or becomes inferior. The work directly adopts the QoC attributes of Buchholz [16] to determine the most appropriate context source. The identification of which source to choose is achieved by selecting the source that has the highest value for a utility function with the QoC attributes as input and a number indicating the degree to which these attributes satisfy the parameters in total as output.

Analysis

The adaptive middleware framework maximizes a number of QoC parameters when selecting a context source. However, in its quest for context information it does not consider whether it appropriately describes the application's user, so the application selects from context information describing any available device. Because the framework does not consider situational relevance beyond up-to-dateness, i.e., timeliness, the probability of acquiring irrelevant context information is great. While this liberal approach to selection means that there are more sources from which to acquire context information, the context values are likely to be inaccurate overall.

There is very little information on the implementation of the actual QoC parameters as this is considered application-specific. They suggest using a relevance-based decision tree learning model to learn the utility function and the QoC parameters' relative importance. However, this requires that the user provide feedback or the application itself has the knowledge to do so, limiting the framework's knowledge autonomy. There is no mention of considering the context of previous experience, so instance set selection is not applicable and instance set sparseness remains a problem in sparse environments. They present no evaluation of their framework.

2.1.2 IST-CONTEXT

The IST-CONTEXT project [27, 26] developed a context acquisition architecture whose goal is the quality-aware discovery of context information sources. In this work, a context information source is a service that provides one or more context attributes. This quality-aware discovery is performed by dynamically selecting the source of context information according to the tradeoffs with cost, user preferences, and Quality of Context. The architecture contains a federation of *Context Brokers* that acquire the context information from *Context Providers* and then provide the acquired context information to client context-aware applications called *Context-Aware Services*. Within each *Context Broker* there is a *Context Matching Engine* that is responsible for the quality-aware acquisition of context information.

The *Context Matching Engine* has three different modules that aid in the acquisition of context information (Figure 2.1). The *Context Validation Process* monitors the

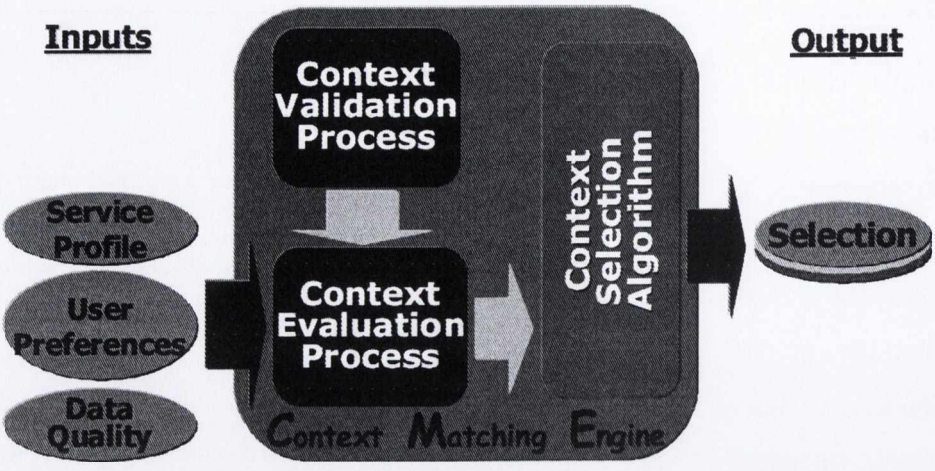

Figure 2.1: Context Matching Engine modules from [27].

Context Providers for their availability and for logistical properties such as response time. The *Context Evaluation Process* computes the expected service and user satisfaction for the available context information. Finally, the *Context Selection Algorithm* is responsible for determining the best source from which to acquire the context information. The algorithm considers parameters such as Quality of Context and logistical information. It maximizes a utility function to select the context source that best satisfies these parameters overall.

Analysis

IST-CONTEXT is very similar to Huebscher and McCann [57] (Section 2.1.1) with the added selection restrictions of cost and time-response limits. To maximize the total benefit and minimize the total costs, the approach also acquires multiple context types at once instead of acquiring them independently. Their approach to timeliness is to favor a more recent context value over one with an older timestamp, all other things being equal. The assumption is that a fresher value will be more accurate. However, the approach only considers context information that describes the desired user, so relevance is not considered beyond timeliness.

Only the current context of the available *Context Providers* is considered, so no instance set selection is performed. Additionally, instance set sparseness is not addressed and the problem is exacerbated by limiting the choices that are available by only considering context information that explicitly describes the application's user. By not

considering context information describing other entities, the application runs a high risk of not acquiring the most accurate value or not being able to acquire one at all. As the approach does not address situational relevance, their knowledge requirements are small. However, knowledge autonomy is limited as the relative importance of each parameter is required beforehand and is specific to each user. The authors suggest it could be learned, although this requires user feedback.

The authors evaluate their approach against various selection methods that favor one parameter or do not consider their relative importance, i.e., all parameters have equal importance. The results show that their approach has the highest utility of approaches that do not violate their cost or latency constraints. However, the external validity of these results is limited as the data used for this test is simulated and they generalize their conclusions based on a single test.

2.1.3 Preuveneers and Berbers

While not explicitly using a linear utility function, Preuveneers and Berbers [111] combine and maximize parameters in a very similar manner to the other works in this section. Their target environment is a collection of smart objects such as shoes, cars, coffee cups, and refrigerators. These objects are connected together through Mobile-Area Networks (MANETs). The goals of the authors' work are to combat the uncertain and ambiguous nature of context information, to acquire the relevant information for their object's applications, and to identify context providers that provide that relevant information.

To accomplish these goals the authors created an information model that allows the identification of relevant context information. Their model identifies the fitness of information with a quality vector consisting of information-specific context quality parameters: accuracy, precision, spatial coverage, timeliness, and semantic interpretability. Precision and timeliness were discussed in the introduction to this section. Accuracy is a number representing the context provider's confidence in the accuracy of the value. Spatial coverage is a knowledge-heavy parameter that determines what locations have relevant context to the current locations. Semantic interpretability uses an ontology to determine if the context representation used by the provider to describe the context information can be understood by the user's application. They also include

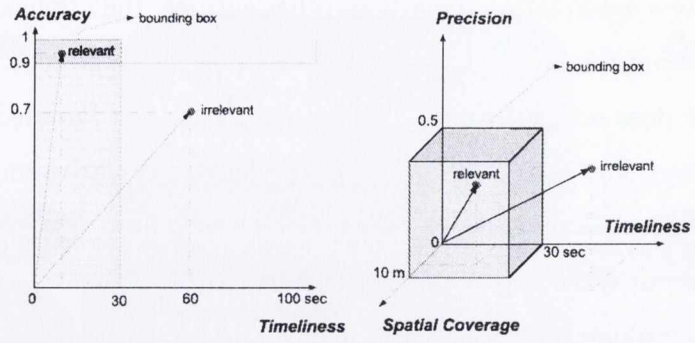

Figure 2.2: Bounding boxes for various parameters from [111].

observed non-information specific parameters of context providers representing trust, availability of information, and missing information. The available context information is first filtered through the quality vector's pre-defined boundary box, i.e., thresholds for each parameter. An example bounding box and associated relevant and irrelevant quality vectors are shown in Figure 2.2. The first graph shows a bounding box for the accuracy and timeliness parameters, and the second graph shows the bounding box for the precision, timeliness, and spatial coverage parameters. The irrelevant information falls outside of the bounding box. The approach then uses the Quality of Context and non-information specific parameters to acquire the value that best satisfies the parameters, and this value is considered the most relevant.

Analysis

With the addition of the spatial coverage parameter to the timeliness parameter, the authors' approach to situational relevance is more sophisticated than the previously discussed approaches. However, this improvement is specific to spatial context and comes at the cost of acquiring spatial knowledge that the other approaches do not require. The knowledge autonomy is also hindered by the need for defining the parameters of the bounding box and an ontology for calculating the semantic interpretability parameter. Instance set selection is not taken into account as only the information of other context providers is considered. Additionally, instance set sparseness is not dealt with and is exasperated by filtering values whose parameters are not within the specified bounding boxes.

The authors' evaluation of their approach tests the relevance and required band-

width for a multi-hop network. This is done for a specific, arbitrarily simulated environment. Additionally, the results of the experiment only apply to the particular simulation, making it difficult to make generalizations about the approach. These limitations negatively affect the external validity of the evaluations. Finally, a lack of reported statistical analysis hinders the conclusion validity of their results.

2.1.4 Summary

Context selection methods do not adequately support any of the research question criteria presented in the introduction. Most significantly, they all assume that the application's user is described by the information of the context provider. Assuming that the context information of all context providers is relevant to the application's user is not a correct assumption in general pervasive domains and will cause the accuracy of applications using such approaches to suffer. While they all consider timeliness in some form, they do not handle the situational relevance criterion required to confidently accept another entity's context information as describing the application's user.

2.2 Proximity-Based Methods in Pervasive Environments

To determine the relevance of other users' context information, the works in this section appropriate context information only from sources that are spatially near their user's device. They assume that any devices within communication range are in close proximity to the device by using short-range communication means, e.g., Bluetooth or Wi-Fi. These proximity-based acquisition methods attempt to supplement the lack of situational relevance beyond the timeliness of context selection methods (Section 2.1) by including spatial relevance as a requirement for the appropriation of other users' context information.

2.2.1 Mobile Sensing Group

The work of the Mobile Sensing Group at Dartmouth College is concerned with people-centric sensing using mobile phones [68, 20]. Their work focuses on how the knowledge

of a user's social network is manipulated to improve people-centric sensing [68] and how the sensing enhances the social network, e.g., sensed aspects of a user's situation such as location or activity are displayed in a user's social networking application like Facebook or MySpace [86].

The work focuses on heterogeneous environments where some users' devices have more accurate sensing capabilities than others. In their scenario, more capable devices share context in order to improve the accuracy of the context information of less capable devices. When possible, less capable devices supplement their collection of current context information with the more accurate context values of the attribute types they are missing by acquiring them from nearby devices. This is achieved through a process called *opportunistic feature vector merging* [68]. A feature is a piece of context information and a feature vector is a collection of context representing the user's situation at a specific time, i.e., a situation instance (see Section 1.1.1). The merging of feature vectors from the local user and from a remote device is referred to as opportunistic because it is an interaction based on the devices that are currently in proximity to each other. These merged feature vectors are used to more completely describe the user's situation for use in learning a classification model (as a training instance) and for providing a richer description of the problem. The latter is shown in Figure 2.3.

Features and models of a remote user are assumed to be more relevant the closer the two users are connected in their social-network, e.g., features from friends are assumed to be more relevant than those of "friends of friends". They refer to sharing with social network bias as *social-network-driven sharing.* Social-network-driven sharing is carried out by reusing a remote device's training data (i.e., feature vectors) and by reusing remote devices' learned classification models (as shown in Figure 2.3). The training data and models are chosen depending on an estimated confidence in the ability of each to solve the current task and the closeness of their respective devices in their social-network.

Analysis

The approach of the Mobile Sensing Group to leveraging other users' information based on social networks is novel but useful only in specific situations and for certain types

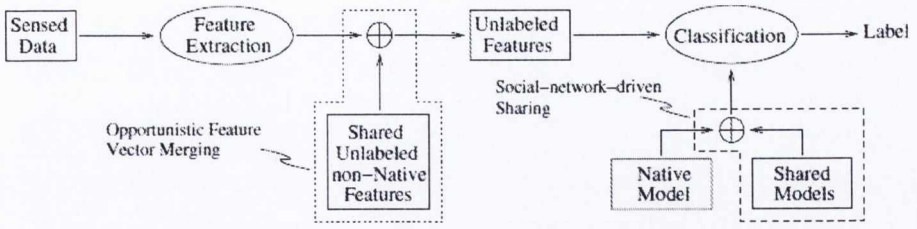

Figure 2.3: Opportunistic Feature Vector Merging and Social-network driven sharing from [68].

of context information. Instance set selection based on social circle requires knowledge of the social network and of how each social network applies to a classification task. For example, a family is a strongly connected social network, but the relevance of their features to a member while that member is at work is limited. More seriously, if there are no members of a social circle available, sharing does not take place. The devices of "strangers" are ignored, even if their situations are potentially similar. For example, strangers riding a bus together or attendees at a conference that are unfamiliar with each other will not be able to utilize each others' potentially relevant information.

Most of these limitations are due to the fact that situational relevance is only partially addressed by proximity-based relevance. Particularly, the opportunistic feature vector merging suffers from a lack of generality, most of which can be attributed to a lack of situational relevance beyond spatial proximity. In their approach, if a feature is available from a device in proximity it is acquired and assumed that the feature describes the user's situation. The authors recognize this is a problem and propose a solution of limiting the features by identifying which ones are "sharable", i.e., which features are relevant to a classification task based on proximity. This decreases the generality of the approach by limiting the acquisition to certain feature types and also requires the knowledge of which features are sharable in a certain task environment. In addition, training instances are shared without regard to timeliness which can affect accuracy in the face of concept drift, i.e., when recent instances more accurately reflect the current situation.

As noted in the previous paragraph, knowledge autonomy is limited due to a lack of general situational relevance. However, the authors emphasize that labeling and other feedback is problematic to obtain manually and note that reusing other users' labeled

instances is a good way to avoid these problems. Instance set sparseness is also not explicitly addressed.

The authors evaluate their approach by distributing devices to members of their department and determining how often a significant location is correctly recognized. A significant location is one that is meaningful for a user, e.g., a lab or home. They use different social groups such as "students" and "Facebook" to evaluate the degree to which membership in these social networks can aid in recognizing significant locations. Opportunistic feature vector merging is tested by merging different forms of location-sensing, i.e., GPS, Wi-Fi, and Bluetooth. Their results showed that classifier accuracy was improved with merging and with sharing models based on social network.

The external validity of their evaluation is initially promising as their dataset comes from actual devices in a pervasive environment. For training purposes, the users had to manually indicate when they were in a significant place and what label that location should receive. Beyond that, the devices tracked the users' normal daily routine. However, the external validity for applications without location attributes is not as convincing. The favorable results for opportunistic feature vector merging are attributable to the type of context information being shared, i.e., location-based information. The relatively limited range of the Bluetooth communication channel used to identify users with which to share meant that users had to be in the same place in order to share their location information. It is less clear how successful generalizing to the merging of features not explicitly representing location based only on device proximity would be. Additionally, the single classification task for a single dataset limits the external validity of the experiments. Finally, the lack of reported statistical analysis on the significance of the differences in the average classification accuracy for each approach limits the evaluation's conclusion validity.

2.2.2 Collaborative Context Recognition

The Nokia Research Center in Tampere, Finland presents a process called collaborative context recognition [58, 81, 88] as a means to improve the accuracy of context information through negotiation with other users. It is not strictly a method of context acquisition, but the approach is pertinent to this thesis as it uses information from surrounding users to improve the accuracy of the derived value. Specifically, the mo-

bile device of a user collaborates with other users' mobile devices in a wireless ad hoc proximity network to reach a consensus of what the current situation is. The approach is presented as analogous to human behavior in social situations by adjusting behavior to be the same as other people in the environment, e.g., setting mobile phones to silent when others do so while at the cinema or at a dinner party.

In the method for collaborative context recognition, the result is a weighted average between the value of the context information on the local device and the average value of all context information of other users. The weights are based on how constant the values for context information are over the last fixed time interval and by how similar the current context values of all users are.

Analysis

Collaborative Context Recognition is not a method for context acquisition and requires that all devices already have a value for the attribute type being recognized. Like the proximity-based appropriation of context information in the approach proposed by the Mobile Sensing Group [68] (Section 2.2.1), the main limitations of this approach are due to the lack of situational relevance beyond proximity-based relevance. They identify this limitation [88] and suggest limiting the types of context to those that are similar for users in close proximity. However, this severely limits the applicability of the method for general pervasive environments. The weight for the average of all current context information of a type in the environment is affected by the "reliability", i.e., how far off an aggregate of all context values for that attribute type are. However, there is no discussion of relative importance of an attribute which limits the identification of situational relevance.

There is no instance set selection as the resulting value is always an average of all the values of each user in the environment. This can also lead to problems with sparse instance sets in environments with few or no other users. Additionally, there are very few knowledge requirements due to the limiting assumptions of the approach. The few parameters that are required are noted as being derived from unexplained heuristics, limiting the knowledge autonomy of the approach.

In earlier evaluations [81], the authors use a collection of users with devices and attempt to predict whether the current context is "walking outside" or "walking inside".

The external validity for the dataset is good for the first test as the data is generated from real context data run on customized equipment. It is not as good for the other test as the data is generated from a simulation by introducing artificial noise with seemingly arbitrary parameters. The external validity for generalizing the results is limited as the evaluation is for a single, simple scenario, i.e., walking inside or outside, and the result is only valid for context attributes that show correlation with close proximity.

Another evaluation [58] has limited external validity as it performs evaluations of context recognition where users are performing the exact same activity. The classifier used is simplified from [81] and is a minimum-distance classifier [37]. The data for the evaluation is collected from actual users. While it is not a simultaneous, multiuser environment, they combine portions of datasets from different users that are performing the same activity and test different voting mechanisms for collaborative context recognition. The use of real data from single users to produce a multiuser scenario increases the external validity of the results relative to simulating the data. However, the dataset consists of users performing the exact same activity. Pervasive environments often have different users performing varied activities, so this restriction harms the external validity of the results.

2.2.3 Summary

Proximity-based acquisition methods appropriate the context of other devices that are near them, where proximity is assumed by the ability to communicate. This is useful for predicting the relevance of environmental context information in some situations. However, as both works note, this is not a viable general approach to context acquisition. Although the Mobile Sensing Group (Section 2.2.1) also includes social-network-based relevance, it does not identify users that are in a similar current situation. The approaches of the next two sections attempt to deal with the totality of situational relevance.

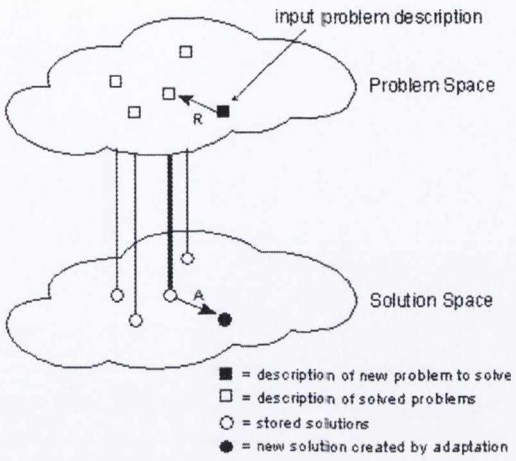

Figure 2.4: The relationship between problem and solution spaces in CBR from [33].

2.3 CBR in Pervasive Environments

The general situational relevance that is missing from the works described in the first two sections of this chapter can be addressed by applying learning methods to the acquisition of context. The assumption is that the collection of context information available is enough to define the situation in which the user is operating. When two users are in similar situations, i.e., when their available context attributes are similar, context can be shared between them. This generalizes the proximity-based situational relevance described in Section 2.2 to the entire available context of the situation. For example, context is appropriated not only when two users are spatially close but also requires some overall similarity with context such as temperature, movement, aural sensors, and user role.

In the works of this section, Case-Based Reasoning (CBR) is used to acquire the desired context value based on the values of the other context types available. Other learning methods can be used, but CBR is the only one in the literature that is used for acquiring current context from other users. CBR is an approach that emphasizes prior experience when solving future problems by adapting solutions from similar past problems [33]. This process is shown in Figure 2.4. The unfilled squares are previous problems and the unfilled circles represent their associated solutions. The problem description and solution make up what is referred to as a case. The case of the previous problem that is most similar to the input problem is identified and that case's solution

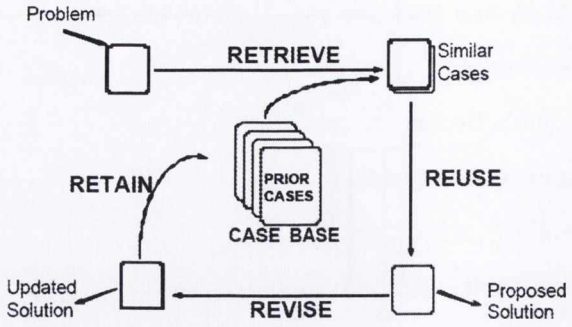

Figure 2.5: The CBR cycle from [33].

is used as a starting point for adaptation. Often, no adaptation is performed and the solution is reused as is.

The CBR cycle is shown in Figure 2.5 and described as follows [2, 33]:

Retrieve This step retrieves the most similar case(s) from the case base.

Reuse This step reuses the information and knowledge in the retrieved case(s) to solve the problem, possibly adapting the previous solution(s).

Revise The revise step is a feedback step that assesses how well the derived solution satisfied the requirements of the problem.

Retain The useful parts of the problem solving experience are retained for future use. This often means that the problem and solution are combined into a new case and added to the case base.

CBR for pervasive applications generally represents a case as a collection of context values representing a snapshot of the user's situation. For context acquisition, the representation of a problem is equivalent to that of a case without the solution's value included, i.e., all the context types have values except for the type of context being acquired. Additionally, previous problems do not have to be instances from some long past situation. They can be used immediately once the case has been defined. This allows pervasive applications to consider current situations as well as ones defined some time earlier. Finally, knowledge defining the relative importance of the attributes for a specific task is also needed for similarity-based retrieval. For example, to determine the ambient temperature of a room, similar location is generally more predictive than similar user role.

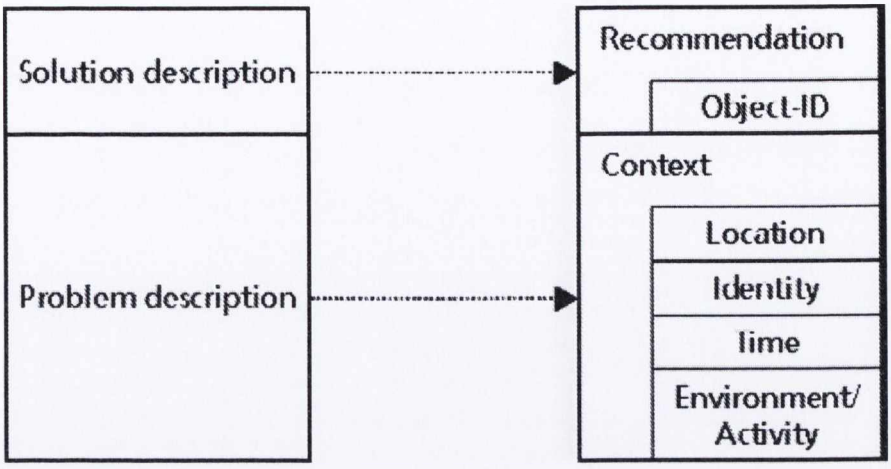

Figure 2.6: Case Representation of a Context Description from [152].

The following works are all applications that use CBR for context acquisition in pervasive environments and involve other users besides the local user. There are many more works that use CBR in pervasive environments, e.g., [78, 62, 117], but these do not consider other users' current information.

2.3.1 The LISTEN Project

The LISTEN Project [153, 152] from the Fraunhofer Institute for Applied Information Technology in Sankt Augustin, Germany presents a pervasive application for the August Macke Laboratory at the Kunstmuseum Bonn. The work was one of the first to use CBR for a pervasive application and to model a case as a collection of context information (Figure 2.6). To enrich a museum patron's experience, the application plays aural messages and makes recommendations to patrons based on their context, e.g., current location, head orientation, and aesthetic preferences. The recommendations are derived using a case-based reasoning approach from an underlying CBR framework that uses cases for exhibits, sound entities, and visitors' situations. One of the main goals of the framework is to simulate the user's way of thinking in order to personalize the recommendations. The *problem description* (Figure 2.6) consists of a four-dimensional vector of location, identity, time, and environment/activity context attributes and the *solution description* is a recommendation for the situation described by the four context values in the problem description. Each individual vector attribute is connected to at least one sensor that automatically updates its associated value.

Like most CBR approaches, a k-nearest neighbor algorithm is used to select the case that is most similar to the user's current situation, which is sensed as the user walks through the museum. The underlying infrastructure then reuses the recommendation of the selected case's solution description as the recommendation for the user's current situation if the similarity of the problem to the case is above a certain threshold. A central infrastructure decides which subset of the case set is useful and performs the retrieve step of the CBR process.

Analysis

The LISTEN project uses the information of other users to determine if two domain objects, i.e., physical objects like paintings or other users, are similar enough to interact. Beyond this, their use of other users' information is limited. The authors mention that other users' information may be used if the personal experience of the application user is inadequate, but do not describe the mechanism behind the instance set selection or how this could deal with instance set sparseness. While the application is deployed in an actual pervasive environment, the evaluation does not compare reasoning methods and only tests the application's "acceptability" to museum visitors [153].

The authors recognize that knowledge autonomy is a major drawback to their approach. The cases for the exhibit objects must be manually elicited and will become obsolete when the exhibit is modified or replaced. They also note the lack of explicit user feedback will be a problem when attempting to learn the importance of each attribute. Methods for learning the relative importance of each attribute generally require supervised learning and this often takes the form of user feedback from experts [128]. Identification of situational relevance suffers without the ability to update the relative importance of an attribute for the current situation.

2.3.2 AmICREEK

AmICREEK [64, 63] is a pervasive application architecture that started with the EU funded AmbieSense project. AmICREEK uses knowledge-intensive CBR to identify the current situation and to recommend appropriate tasks for the situation. It is "knowledge-intensive" as it uses the *CREEK* system [1]. The *CREEK* system does not employ a simple nearest-neighbor algorithm, but instead determines the appropri-

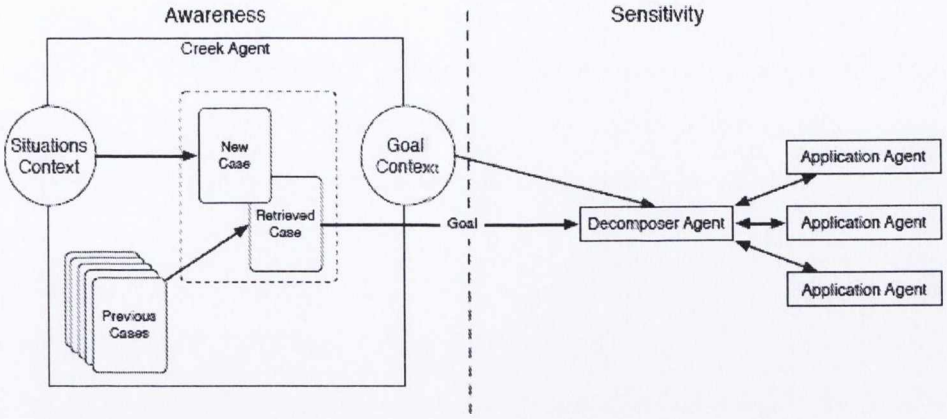

Figure 2.7: The Awareness and Sensitivity layers of the AmICREEK functional system architecture adapted from [64].

ateness of a case by how well the two cases match using a rich knowledge model that connects information such as the context model, domain-specific reasoning knowledge, and existing cases. The cases share the same multi-relational semantic network as the domain knowledge. Cases are prioritized by the number of matched relations and their respective relevance [1]. The application is focused on the healthcare domain and presents an evaluation using real data collected from an ethnographic study at St. Olav Hospital in Trondheim, Norway.

Figure 2.7 shows the *Awareness* and *Sensitivity layers* from the three-layered functional system architecture of AmICREEK. The first layer is the *Perception layer* and it observes the environment through sensors to record the current *Situations Context*. The *Awareness layer* contains the *CREEK* agent which uses knowledge-intensive CBR to derive the appropriate situation label for the new *Situations Context* using the collection of previous cases. The derived situation label is then used to determine the overall goal and tasks appropriate to the situation. These are then broken down to appropriate subtasks specific to the current situation by the decomposer agent in the *Sensitivity layer*.

Analysis

As AmICREEK is knowledge-intensive, it has many advantages over the other knowledge-light approaches discussed in this chapter for many domains. The rich, structured knowledge allows retrieval and adaptation that is domain-specific and that a knowledge-

light model cannot easily provide. Additionally, approaches that use rich domain knowledge can overcome instance set sparseness. They require fewer cases to perform appropriately as these approaches do not rely solely on the knowledge contained in the cases. The domain knowledge can also be used to perform problem solving in the event that no case is returned [1].

The advantages of knowledge-intensive approaches are less compelling in dynamic pervasive environments [70]. Chief among the reasons for this is the lack of knowledge autonomy. There is a significant initial cost for collecting and structuring the knowledge for a specific domain. Additionally, even if a system is initialized properly, if the knowledge or reasoning becomes obsolete through device movement or concept drift, the accuracy of the acquired context information will degrade. The relative importance of context types can also shift, negatively affecting situational relevance.

Like LISTEN (Section 2.3.1), AmICREEK uses the information of other users but does not directly acquire it. Consequently, there is no instance set selection as there is only the local case base from which to acquire information. The AmICREEK architecture could be included as part of an agent in a collaborative CBR system like that described in the CCBR framework in Section 2.3.3. The knowledge-intensive nature of AmICREEK would then aid the selection of similar users, albeit with the same caveat about the long-term viability of static knowledge solutions in dynamic environments discussed in the previous paragraph.

The evaluation of the AmICREEK approach tests the awareness layer by predicting the current situation type of an unlabeled *Situations Context*. A physician from the cardiology ward who encountered the highest number of different situations was observed over several days and the situations encountered were recorded. The situations were then manually labeled by a domain expert. The performance of the automatic situation prediction was then tested by using a subset of the cases as the case base and a further subset as the test set (omitting the value they wished to estimate from the test instances). The data used by the evaluation is externally valid as all data came from a real-life situation and the instances were labeled by an expert. However, the evaluations are not appropriate for the research questions of this thesis as they do not compare against any other methods. The conclusion validity of the evaluations is limited as the overall classification accuracy of each situation type is given without any

statistical analysis. Additionally, only one physician was observed, a specific subset of the days (days 13 and 14) were used, and an evaluation with a single specific partition of those days was reported. Consequently, the results are biased to the particular physician, day, and dataset partition leaving the external validity of the results less convincing.

2.3.3 Collaborative CBR (CCBR)

McGinty and Smyth [83, 82, 84] from University College Dublin introduce the Collaborative CBR (CCBR) framework for allowing agents to share their problem solving experience. They present an application of the CCBR framework that is a CBR-based approach to personalized route planning, i.e., presenting the route between two locations that best satisfies the user's preference model.

The CCBR framework belongs in both this section on CBR in pervasive environments and the next section on multi-agent CBR (see Section 2.4). Unlike this work's application to personalized routes, the works in Section 2.4 do not explicitly deal with pervasive applications but still contain properties that are relevant to pervasive environments. The CCBR framework is presented first here and then referenced where applicable in the section on multi-agent CBR.

This work is concerned with producing routes in areas outside of a user's previous experience. Solutions are achieved by using previous experience of other users who have experience in the unfamiliar geographic areas and who also have similar preference models to the application's user. Preference model similarity is determined by the degree of the similarity of routes chosen in geographic areas where their previous experience intersects.

Figure 2.8 shows the top-level algorithm of the CCBR framework. The application's user passes the problem (p) to its agent (*Target Agent* A_i). If the *Target Agent's* CBR component can solve the problem using only previous experience, a solution is returned (s). If not, the problem and the *Target Agent's* case base is given to its *Collaborator Component* and transferred to the *Collaborator Component* of a *Collaborating Agent.* The *Collaborating Agent* represents another user in the environment. The *Collaborating Agent* determines if it can solve the problem and if so, what the quality of the solution is. For the personalized route planning application, the quality of the solution

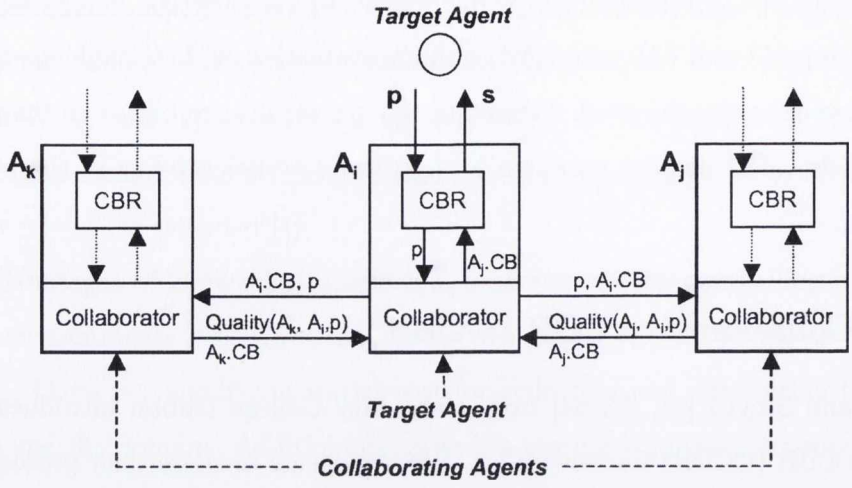

Figure 2.8: The top-level CCBR agent algorithm from [83].

is dependent on the *Collaborating Agent's* familiarity in the geographic area of the problem and the similarity of the *Target* and *Collaborating Agents* on shared routes. Finally, the *Target Agent* selects the highest quality *Collaborating Agent* and uses its cases to construct a solution to return to the application's user.

Analysis

While the personalized route planning application deals with an important problem, its use as an example of the domain and task independent CCBR framework is not ideal. The application's input and output is specific to one type of context, i.e., routes, so issues related to multi-context pervasive environments are not addressed. For example, feedback for learning each context type's relative importance is not applicable with only one type of context. Additionally, the similarity metrics for determining problem coverage and similarity of users are knowledge-intensive and domain-specific. For these reasons, the knowledge autonomy and situational relevance of the presented framework is limited.

The authors deal with instance set selection by first determining if the user's previous experience can help acquire the route context. If the previous experience coverage is too sparse or otherwise not applicable, it then queries the other users. The instance set selection is domain-specific as it identifies similar users by similar routes. In addition, the routes returned are previous routes with no consideration of the time they

were taken. Without consideration of timeliness, a user's preferences that evolve over time or physical changes to certain routes, e.g., newly opened or closed streets, will not be appropriately prioritized.

There is also a requirement for the previous experience instance set to be non-sparse, both for the application's user and for the other users from which the application acquires its information. If either set is sparse, then the appropriate preference model similarity between two users cannot be determined. The authors acknowledge this limitation and suggest manual elicitation of routes or an initial startup time to learn routes where the personalization functionality of the application is not available. The sparse instance set problem is exacerbated by the use of a single context type. The problem can be alleviated somewhat by the inclusion of other readily available context information, e.g., a driver's age or type of vehicle, that has some correlation to a user's preference model (see demographic-based recommenders in Section 2.5.1.3). The multi-context approach would allow the application to be somewhat personalized while the user builds the store of familiar routes.

The lack of a dataset from real users required the authors to simulate route problems and solutions. The case base is created by defining a preference for each segment on a map and generating a route that optimally satisfies these preferences for a given set of problems. The experiment was run using leave-one-out cross-validation for each user in the user set. These experiments used a separate test set and were run for a varying number of case base sizes. The results of the evaluation noted improvements in time efficiency and solution quality of their approach against a standard A* planner. The simulation of the datasets did not have to deal with relationships between context types as route was the only attribute type.

The lack of multiuser datasets required the authors to make several simplifying assumptions in order to evaluate their work. First, there is no correlation between a user's previous routes other than being restricted to a specific area. In addition, the authors imply that the users' preference for a segment are matched somehow, artificially producing the similarity between user preference models necessary to run the evaluations. Finally, a realistic case base for a user would not contain a single, static route between two locations. The simulation assumes the user knows about and is able to calculate the optimal route, ignoring the times the user might have gotten

lost or taken an otherwise suboptimal route. Routes are also unaffected by context information such as user's schedule, time of day, or traffic congestion. These multiple, varied routes would be recorded in the case base in an actual deployment of their application as it currently stands, complicating its capability to determine users with similar preference models and adding routes that are not appropriate for all situations. The simplifying assumptions allowed the authors to answer their research questions, but the results have limited generalizability when considering more attribute types and other pervasive applications.

2.3.4 Summary

The works in this section attempt to solve the situational relevance deficiencies of the first two sections of this chapter by applying CBR to the previous experience of the application's user and to information about other users. By considering the similarity of all attributes that make up the user's situation, they have a richer notion of situational relevance.

However, with this increased functionality comes the necessity of greater knowledge about the relative importance of each attribute, and knowledge autonomy is not very high. It also requires a number of cases from which to select. They attempt to alleviate this problem by selecting from previous experience, but that is not always appropriate in dynamic domains like pervasive environments where unfamiliar situations and concept drift are encountered.

2.4 Multi-Agent CBR

Multi-agent CBR systems face many of the same problems as multiuser pervasive environments. In multi-agent CBR, each agent has a case base where the agent's experience is stored. The CCBR Framework [83] in Section 2.3.3 is an example of a multi-agent system where each agent represents a user. Like the CCBR Framework, the two multi-agent CBR works in this section use CBR to share experience. Unlike the CCBR Framework, these two works' focus is not on pervasive applications.

Multi-agent systems are complex systems that involve many agents and assist the coordination of independent agents' behavior [130]. The initial work using CBR in

multi-agent systems was carried out by Prasad et al. [110] and focused on aggregating a total solution from agents that solved different subsets of the problem. This chapter focuses rather on individual agents that have the competence to solve the problem completely, not just subparts of the problem. To this end, Plaza et al. [108] define a set of requirements for the agents in multi-agent CBR:

Homogeneous agents The representation of the problems are the same for all agents.

Peer agents Each agent is able to solve the entire problem, not just specialized sections of it.

Learning agents The agents learn from their individual experiences and also leverage the usually divergent experience of other agents for solving a task.

The following two works, as well as the CCBR Framework of Section 2.3.3, fit all of these requirements. The summary in Section 2.4.3 will discuss how all three relate to each other and how well their methods are suited for pervasive environments.

2.4.1 Multi-Case-Base Reasoning (MCBR)

Leake and Sooriamurthi, from Indiana University, produced Multi-Case-Base Reasoning (MCBR) [71, 72, 73, 74, 75]. MCBR lays out a framework that allows the utilization of external case bases to supplement the local case base. Figure 2.9 shows the MCBR framework. The problem is presented to the *Case Dispatcher* which selects the case base that is predicted to have the best solution and dispatches the problem to it. These case bases can be the local case base or any number of external case bases. The returned solutions are then adapted for any differences between the external and local case bases by the *Cross Case Base Adapter*. *Cross case adaptations* modify the solutions for the local case base as the external case base might have been developed in different ways and for different tasks or task environments [73]. The most appropriate solutions are selected and then merged, if necessary, by the *Solution Selector* and the *Solution Merger*, respectively. The remainder of the framework performs the last two steps of the CBR process, i.e., reuse and revise, after which the final solution is returned.

Particularly relevant to this chapter is their approach to problem dispatching, i.e., instance set selection. They present two domain-independent, autonomous methods

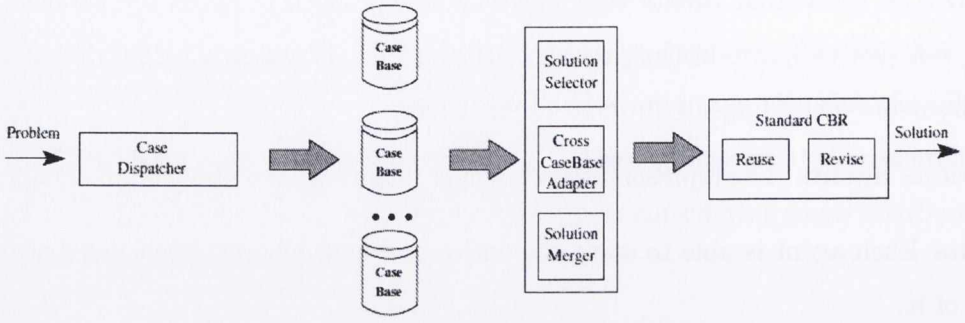

Figure 2.9: MCBR framework from [74].

for selecting the case base: threshold-based dispatching and case-based dispatching [72]. *Threshold-based dispatching* only dispatches the problem to an external case base if the similarity of the problem to the case deemed best from the local case base is below a certain threshold. If it is below the threshold, the best case is retrieved from the external case base. If the external case's similarity is higher than the best case from the local case base, the external case base is selected to solve the problem. Otherwise, the local case base is used. This approach prioritizes the local case base unless it is judged to be ill-equipped to handle the problem, i.e., if the similarity between the problem and the best local case is below the threshold. In their experiments, this prioritization performs more accurately than merging all the case bases and selecting the case with the highest overall similarity [74, 75]. The authors hypothesize that the increase in performance over a merged case base is due to differences between the external and local case bases and that the local case base is personalized to handle the specific problems of the local agent.

Case-based dispatching attempts to determine the best case base for solving a new problem by determining which case base (local or external) best solved similar problems. Initially, a group of test problems is solved and the case base that solves a particular problem the best is recorded. When a new problem is introduced, its similarity to the test problems is measured and the target case base is selected through a similarity-weighted voting scheme where each test problem votes for the case base that solved it the best. The case base with the greatest percentage of the vote is then

selected for retrieval.

Analysis

When applied to relatively static task environments, the MCBR framework fits the research question criteria very well. Their problem dispatching focus enables the best instance set to be selected for their problem-domain. They provide a domain-independent, autonomous method to calibrate the threshold that functions well with a sparse number of training instances. The ability to calibrate the Case Dispatcher with a small number of instances combined with the automatic instance set selection means the approach functions quite well when dealing with instance set sparseness. In fact, most of the experimental results [72, 71] of MCBR focus on the benefits of the approach when building the case base initially, i.e., when dealing with the sparse instance set of the local case base. As the agent becomes more experienced in performing its task, it has less need for the assistance of the external case base. As the number of cases in the local case base increases, the benefits of using external case bases become less pronounced and eventually using external case bases hurts the performance.

The increased competence of the case base as the number of cases increases, however, epitomizes why MCBR and other existing approaches to multi-agent CBR are not suited to dynamic systems like pervasive applications. MCBR's focus is on learning a particular task in a particular static task environment such as travel bookings [71] or real estate [74, 72]. They may borrow cases from external case bases representing other task environments, but their task and environment remains the same. Eventually, after enough relevant cases are collected, the local agent will be a self-sufficient problem solver. In pervasive applications, the assumptions of fixed task and task environment do not hold. For example, the prioritization of the local case base over external case bases is a good general rule in a fixed task environment but that rule is not applicable in the unfamiliar environments common to pervasive applications where external sources may have more experience in these environments. Furthermore, favoring a local agent's previous experience for the acquisition of current shared-environment information over using that same information currently being read in by external sensors would hinder accuracy. Finally, there is no mention of learning the relative importance of each attribute or having to update it as the latter is unnecessary in a static task

environment. This hinders both knowledge autonomy and situational relevance. While MCBR handles many of the research question criteria and many of the issues addressed in multi-agent CBR and pervasive applications are the same, they are solving them for fundamentally different domains.

Finally, there are several comparative experiments run using MCBR. The authors compare methods of problem dispatching for a spectrum of local case base thresholds [74, 72, 71] and cross-case base adaptation strategies [72]. They evaluate the calibration of these two methods on full and sparse sets [72]. They all run on existing datasets where the local case base runs in a different task environment than the remote one(s), e.g., by state for real estate case bases [74, 72] and by hotel star rating for travel bookings [72]. The latter is an artificial division but is entirely feasible based on the idea that travel agents may specialize in either luxury or economy holiday packages. The major threat to applicability to pervasive environments is that it does not model changing environments well, e.g., inflation for holidays or current changes in the real estate market. To aid in generalizability, the evaluations use repeated runs with different subsets and cross-validation. However, there are two ways in which external validity could be stronger. Firstly, only one dataset is used for each test. Testing on multiple datasets representing different tasks increases the generalizability of the results. Secondly, only one of the subsets is used as the local case base through all the tests. A more thorough evaluation would have the different subsets rotating as the local case base in evaluation repetitions. Finally, while the average accuracy is used, there is no reporting on the statistical variation or significance of the differences among the results for each method they are comparing. Without statistical analysis, the conclusion validity is poor as there is no notion of how much the particular random sampling affected the magnitude of the differences in accuracy.

2.4.2 Cooperative CBR (CoopCBR)

The work of Plaza and Ontanon began at the Artificial Intelligence Research Institute at the Spanish Council for Scientific Research in Spain. Plaza et al.'s initial paper [108] laid out Cooperative CBR (CoopCBR) whose main goal was to develop techniques to integrate CBR into multi-agent systems. This allows an agent to benefit from the experience of other external agents. Their more recent work [96, 98] focuses on policies

for CoopCBR where external agents vote on a fixed number of solution categories for a given problem based on the contents of their own case base. Consequently, their framework is restricted to classification tasks but at the same time promotes privacy by not requiring agents to expose their cases.

Agents aid each other through an interaction protocol [109] as follows. *Agent 1* sends *Agent 2* a problem. *Agent 2* returns an error message if it cannot solve the problem or a *Solution Endorsement Record* (*SER*) otherwise. An *SER* contains a value representing the number of *Agent 2's* cases that solve the problem for that solution class. The value is weighted by the sum of cases returned for all solution classes. This prevents a single agent from having a greater influence than others based solely on the size of its case base. *Agent 1* collects all the *SERs* from all the agents in the multi-agent system and selects as a solution the solution class that has the most votes. Later work introduces an argumentation framework that allows more sophisticated reasoning for each agent to also consider cases that contradict a solution class [96].

Analysis

The approach of CoopCBR is similar to MCBR (Section 2.4.1) with three differences. First, MCBR focuses on case bases with different task environments and CoopCBR assumes that each case base is part of the same task environment but covering different individual problems. MCBR prioritizes the local case base as it is specialized for the local agent's particular task environment. Using the same prioritization in CoopCBR where all agents operate in the identical task environment hinders performance. That is, the result of not considering all case bases is to shrink the problem space coverage. Second, CoopCBR is restricted to classification tasks as the approach needs a fixed number of solution classes. However, particularly with the latter additions to argumentation [96], the solution resulting from the cooperation is much more robust as all the external agents can voice their support or disapproval of a solution class. Third, as there is no access to the cases of the external agents in CoopCBR, the local agent must rely not only on the problem coverage of the external agents' case bases but the other agents must be similar in how they solve a problem. For example, the external agents that assign different relative importance to problem attributes than the local agent will not return cases that are similar in the same manner that the local agent

is expecting. Overall, this homogeneity requirement for both task environment and reasoning limits the domains in which CoopCBR can function.

With CoopCBR's assumption of a uniform task environment, instance set sparseness and instance set selection do not need to be considered, i.e., all case bases are considered because more instance sets mean more problem space coverage for the task. Situational relevance is hindered if the agents are not homogeneous (as is common in pervasive environments), as the similarity of agents' case bases and the manner in which they solve the problem would need to be considered. Some of the collaboration protocols require parameters like thresholds [72] and the ability to know when an agent can solve a problem [83]. It is unclear how these calibration parameters could be acquired automatically, so knowledge autonomy is limited.

CoopCBR runs comparative evaluations using multiple 10-fold cross-validation to test the different collaboration protocols on the partitions of a machine learning dataset. Unlike MCBR's evaluations, the case bases are random partitions of a dataset rather than a division based on different task environments, i.e., one that gives external case bases less relevance to the problems. This means that the accuracy advantages of MCBR's threshold-based dispatching over merging do not exist in their experiments. The policy that allows all of the agents to contribute performs the best for CoopCBR [109]. The external validity is limited in that the cases of the dataset used are assigned randomly to each agent. Further experiments attempted to correct for this by creating biased case bases that contain more or less of some solution class [97]. However, each agent was still in the same task environment. Unlike MCBR, the first set of experiments in CoopCBR [109] use statistical methods to determine the significance of the differences using the Wilcoxon signed-rank test. They also present the standard deviation for their results. More recent experiments [96] do not use statistical methods, limiting conclusion validity. However, they are more generalizable than the earlier experiments in that they use two datasets instead of just one. The fact that both multiple datasets and statistical methods were not used at the same time hinders both experiments' external and conclusion validity.

2.4.3 Multi-Agent CBR Summary

The works in this section, in addition to the CCBR Framework [83] in Section 2.3.3, are all multi-agent CBR systems that aid the local agent's competence by leveraging the experience of other agents in the environment. One of the key differences between them is the task environment uniformity. MCBR [71] (Section 2.4.1) assumes that the task environment of each agent is different while CoopCBR [96] (Section 2.4.2) assumes that it is the same. The agents in pervasive environments, however, fall in between the two extremes. As an example, in the CCBR Framework the agents have different preferences and different experiences but they rely on the similarities to determine the best agent to solve the problem. Like CoopCBR, a similar task environment is assumed. However, it is not identical as each user has different route preferences. Like MCBR, the local agent's case base is favored for solving the problem.

Another difference is where the retrieval stage of the CBR cycle [33] takes place. At one end of the spectrum, the retrieval in CoopCBR depends on the individual external agents to solve the problem using their own similarity metrics and provide a vote for the solution. At the other end of the spectrum, CCBR receives all the relevant cases from the external agent and retrieves from that subset based on the local agent's similarity. MCBR falls in the middle of the spectrum as the initial retrieval is done externally but the cases themselves are passed back to the local agent for final retrieval and adaptation. The retrieval of the final case by the local agent is ideal for pervasive applications, as the local agent can determine which case best meets its particular requirements for a solution rather than depending on other heterogeneous agents to know these requirements.

As with the CBR agents in Section 2.3, multi-agent CBR does not perform well in dynamic environments without eliciting cases from the new situation. This elicitation requirement means that the applications are less knowledge autonomous. The applicability of the approaches for general pervasive devices is also limited, as they all require that the external agents have a case base. CoopCBR requires it for voting, the CCBR Framework in order to determine similar users, and MCBR to select and calibrate their approach.

As a final note, the validity of the evaluations of the two works in this section is much higher in both external and conclusion validity than the previous three sections

describing applications in pervasive environments. This has much to do with the availability of datasets for machine learning and the lack of such for multiuser pervasive environments. This division can be seen in the external validity between the evaluations of the simulations in the CCBR Framework and the two works in this section. Even though they are all multi-agent CBR systems, the existing datasets for the target environments of the two works in this section allow the evaluations of these non-pervasive approaches greater validity.

The preceding sections were focused as much as possible on the acquisition of current context information from remote sources in people-centric pervasive applications. The next two sections broaden this scope to include applications that relax the restrictions of remote acquisition of current information, people-centric works, and pervasive computing. The next section focuses on predicting a specific type of context information (preferences) through a review of recommender systems. The section after that focuses on missing value prediction in general.

2.5 Preference Learning and Recommender Systems

Tuzhilin [135] defines personalization as "the ability to tailor products and services to individuals based on knowledge of their preferences and behavior." The execution of desirable, personalized behavior for some context-aware applications depends on the correct modeling of the preferences of a user. For example, a user's preference for type of food should greatly influence the recommendations of a restaurant finder application in addition to the other context information such as the user's proximity to the restaurant and current weather [95, 16]. Like user location or ambient temperature, preferences are considered a type of context information [16].

Preferences can be explicitly elicited from the user or implicitly learned [93]. Implicitly learning preferences can be done through observing a user's behavior such as browser click-throughs and purchases [133] or inducing preference models from empirical data [45]. As for other types of context information, the capability of implicitly learning user preferences increases the knowledge autonomy of applications as it does not require the user to give explicit feedback when the user is unable or unwilling to do so. Preference learning is a specification of the general context prediction previously

stressed in this thesis because preferences can be quite specific to the individual and are more difficult to sense automatically through sensors.

This section further broadens the literature review to include non-sensor based context information by exploring preference learning through recommender systems [48] (Section 2.5.1). Like the other sections of this chapter, the discussion is focused on how the area and its specific works deal with the challenges related to the research question criteria. Specifically relevant to this thesis are the recommender systems that use some sort of collaboration with other users to help learn an individual's preferences (Section 2.5.1.2 and Section 2.5.1.3). For completeness, this section also includes a brief discussion of directly executing behavior without explicitly predicting the intermediate context information or preferences (Section 2.5.2). Finally, a summary of how recommender systems relate to this thesis' approach to context prediction is presented at the end of this section.

2.5.1 Recommender Systems

Recommender systems suggest items to users depending on the user's preferences for that item [114]. Examples of items are products such as books or movies, activities such as visiting a restaurant or taking a guided tour, or documents such as web search pages or news items. In recommender systems, preferences are captured as item-based ratings, where items with higher ratings are more likely to get recommended than ones with lower ratings [4]. For example, a book with the highest rating for a user is recommended or news items are ranked depending on their rating assignment for that specific user.

Figure 2.10 presents a taxonomy of recommender systems. Each of the next three sections presents the second level of that taxonomy, and the sections contain representative works from those topics. Issues related to the research question criteria are highlighted, and the approaches recommender systems use to deal with these issues are discussed.

2.5.1.1 Content-based Recommenders

Content-based recommenders recommend items that are similar to ones that the user liked in the past [114]. For these recommenders, the similarity of items is determined

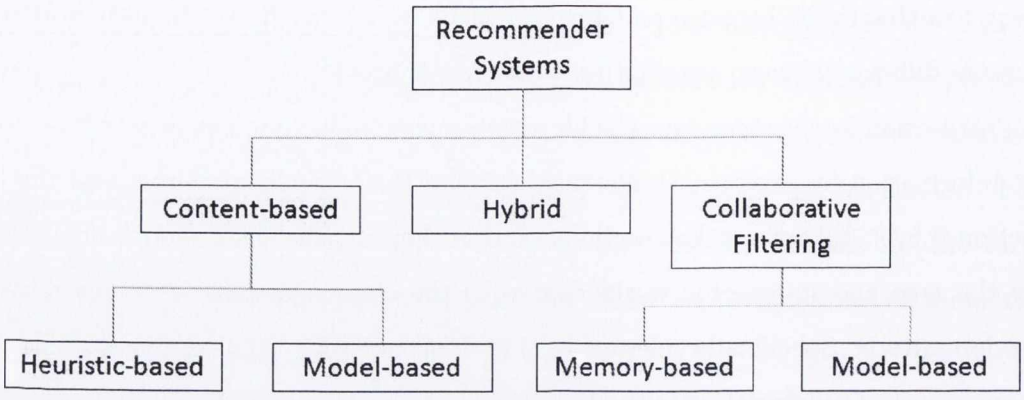

Figure 2.10: A taxonomy of recommender systems.

by the similarity of features that describe the item. Consequently, content-based recommenders are mostly used for recommending text-based items [4]. This is not a requirement, however, as non-text-based items can be annotated with metadata, e.g., movies can be annotated with genre and featured actors descriptors.

There are two types of content-based recommenders: heuristic-based and model-based [4]. Heuristic-based recommenders use mostly information retrieval methods, e.g., the term frequency/inverse document frequency measure (TF-IDF). Model-based recommenders learn a model from the underlying data using statistical and machine learning techniques. Model-based recommenders employ methods such as decision trees, nearest-neighbor, linear classifiers, and naive Bayes [105].

As a representative example of a content-based recommender, Syskill &Webert is an intelligent agent that uses a model-based approach to recommend web sites [103]. The feature space is made up of several Boolean features each indicating whether the page contains a specific word, and each page is labeled as "hot" or "cold". The word choice for the features is chosen using expected information gain directly from the training pages. The authors ran an evaluation of several learning methods and measured their accuracy. The naive Bayes classifier, multilayer perceptron, and Rocchio's algorithm (a method using TF-IDF to weight words) performed the most accurately overall. The naive Bayes classifier was ultimately selected for Syskill & Webert because of its low knowledge requirement and fast performance.

There are a number of issues facing content-based recommenders. First, the *new user problem* occurs when a user is first introduced to the system, and the user has

not rated any items [48]. As content-based recommenders direct their suggestions by a user's item ratings history, this makes any personalized recommendations impossible until the user has rated several items. Second, *overspecialization* is an issue caused by the fact that a recommender system can only suggest items that are similar to the ones the user has rated [4]. For example, if a user has rated only horror films, the system would not be able to recommend any comedies. Finally, the *limited content analysis problem* affects content-based recommenders as they are limited by the features that are explicitly assigned to items [4]. This creates a knowledge acquisition problem for non-text-based items, as automatic feature extraction is difficult for some items, e.g., video and audio files. All three of these issues are at least partially addressed by the collaborative approach of the next section.

2.5.1.2 Collaborative Filtering

Collaborative filtering [49] uses the known preferences of a group of users to make recommendations for items with unknown preference for a user [133]. It accomplishes this by identifying users that have rated past items similarly to the current user and suggesting items to the current user that those similar users have rated highly.

Collaborative filtering approaches are divided similarly to machine learning approaches in that they are either memory-based (instance-based learners) or model-based (eager learners). Memory-based algorithms calculate the similarity between users (e.g., the Pearson correlation or vector cosine-based similarity) and then provide a prediction on an item by taking a weighted average based on the user-similarity of the other users' ratings on that item [118]. Model-based algorithms learn patterns of rated items from the data of other users to predict new items via statistical and machine learning techniques, e.g., Bayesian methods, clustering, and dimensionality reduction algorithms [133]. Memory-based algorithms do not scale well, so most practical algorithms are model-based or combine some sort of pre-computation with memory-based techniques [118].

For example, Su and Khoshgoftaar [132] compare four collaborative filtering approaches to predict ratings of movies using the MovieLens database.[1] The memory-based algorithm uses the Pearson correlation to measure the correlation of two ratings.

[1] http://www.grouplens.org/node/73

The three model-based algorithms are a simple naive Bayes classifier (NB-CF), a naive Bayes classifier using extended logistic regression (ELR) to accommodate incomplete training sets (NB-ELR), and a tree-augmented naive Bayes classifier with ELR (TAN-ELR). The focus of their experiments is the robustness of the recommenders when the user-item matrix is sparse. Results indicate that the approaches with ELR perform with the most overall accuracy when dealing with very sparse user-item matrices.

Collaborative filtering techniques address the feature acquisition and overspecialization problems of content-based recommenders. First, there are no features in collaborative filtering as only the ratings are compared on items that the user and the group of other users have rated. Second, the user is not limited to receiving recommendations only from items that are similar to ones rated previously. Users that rated items similarly to how the current user rated them can recommend any item that they have rated highly, regardless of whether those items are completely different to the ones the current user has rated previously.

While addressing some issues of content-based recommenders, collaborative filtering does introduce a few new issues. First, the *new item problem* describes the lack of ratings for an item when that item is first introduced into the system [9]. This problem does not exist in content-based recommenders, as the similarity there depends on the features of an item and does not depend solely on the rating as it does in collaborative filtering. Second, the *sparsity problem* occurs when there are not enough users and ratings to identify similar users or recommend relevant items. This means that early recommenders receive little value for their rating contribution [65]. Finally, the new user problem also exists for collaborative filtering but combining collaborative filtering techniques with other recommender approaches can ameliorate this problem, as discussed in the next section.

2.5.1.3 Hybrid Recommenders and Emerging Topics

There are several other approaches to recommender systems [114] (e.g., knowledge-based, community-based, and demographic-based), but most works at least use some elements of content-based and collaborative filtering. This section contains a description of hybrid recommender systems as well as an introduction to some recent emerging topics in recommender systems, i.e., community-based recommendation and context-

aware recommendation.

Works that combine more than one of the recommender approaches to improve recommendation performance are called hybrid recommender systems [17], and they are usually employed to deal with the new user and new item problems [18]. One such hybrid approach for dealing with the new user problem is to combine collaborative filtering with demographic filtering. Demographic filtering uses information describing the user, e.g., age, location, and sex, to learn a relationship between an item and the demographic information of people that like that item [104, 121]. While not as highly personalized as consideration of a user's specific tastes, this approach allows a more personalized approach when faced with the new user problem.

Another prominent hybrid approach is based on combining content-based and collaborative filtering techniques. This chapter has already introduced a distributed, hybrid recommender system in the CCBR framework [84] (see Section 2.3.3). Rather than a product, here the items recommended are the initial sub-routes that make up the building blocks of the final route. If the previous experience of the user is enough to handle the query, the CCBR framework does not use collaboration at all but recommends the sub-routes in the store of previously travelled routes strictly through content-based techniques. If it cannot handle the query by itself, the framework uses a content-based similarity measure to identify other users that are similar to the current user, i.e., users with similar route sub-features such as locations and paths. It then recommends the sub-routes of the users with similar route preferences, and the current user utilizes these sub-routes to generate the new route.

A further approach to recommendation is community-based recommendation which utilizes the similarity within a group of users to make recommendations [126]. This is similar to the social network-based approach of the Mobile Sensing Group [68] where users within a social group are considered to have more relevant context than those outside it. As an example of community-based recommendation for web search, HeyStaks [125] is a search utility that works with existing search engines and is focused on organizing and sharing search experiences among friends and colleagues. A *stak* is a type of folder that records and influences search experiences. When a stak is created for a topic of interest, it can be shared among the group. When that stak is active, a search is run as normal except that certain results are moved up in the search list ("promoted")

based on results that other users of the stak found useful for a similar query. These result promotions function as automatically generated recommendations from other members of the community formed by the stak.

Finally, context-aware recommender systems combine the context of the user in addition to the items and ratings to make context-aware recommendations [5]. An example of this is the Support Vector Machine (SVM) approach of Oku et al [95]. They present three binary SVM classifiers to compare in their evaluation. The first uses an SVM classifier to classify an instance with a given number of features. Their second classifier, C-SVM, extends that feature space to also include context information. The third classifier, C-SVM-CF, is a context-based, collaborative filtering approach that takes recommendations from other similar users. Users are considered similar if the combined classification percentage of a user's instances using the other user's model is above a threshold. They run an experiment using a restaurant recommender to compare the approaches. The results show that the collaborative filtering approach suggested more satisfying restaurants than the C-SVM classifier and that both the classifiers that consider context information outperform the SVM approach without context features.

2.5.2 Predicting Behavior Directly

While it is often the case that intermediate values must be predicted in order to execute the behavior of an application, this is not always the case. For example, sometimes preferences are not explicitly generated and just the recommendation is output [77]. This can be generalized where explicitly creating the intermediary missing context information is skipped in favor of predicting the behavior or a higher-level context directly.

Methods that accomplish this direct computation by being tolerant of missing values are common in machine learning. For example, CART [13] is a tree-based algorithm that uses a surrogate split when classifying an instance with missing values. A surrogate split substitutes an existing attribute type that splits the dataset similarly to how the missing attribute type does. As another example, self-organizing maps (SOM) are unsupervised learners that provide data visualization for high-dimensional spaces which can help with identifying clusters. Without some way to handle missing data, potentially valuable information could be lost by discarding incomplete instances. Wang

[139] proposed a fuzzy SOM that could function in the face of missing data by treating missing data as fuzzy variables, i.e., assigning a weight for each possible value of an attribute, and having the outputs be fuzzy as well. These outputs are then presented as a histogram-style graph differentiating crisp outputs from fuzzy outputs while still allowing clusters to be identified. These examples show that it is sometimes possible to deal directly with the ultimate behavior rather than perform the prediction of the intermediary values. However, often these intermediary values are needed for the behavior, e.g., information presentation or for a specific learning method that cannot handle missing information, and these situations comprise the focus of this thesis.

2.5.3 Recommendation Systems Summary

Preference learning is an important component of any context prediction approach as preferences are a common type of context information. Recommender systems provide one popular way to predict preferences, and there are similarities between the issues of recommender systems and the context prediction challenges this thesis attempts to address. Specifically, both the approach of this thesis and collaborative filtering attempt to use the information of other users in order to deal with unfamiliar situations and overspecialization, respectively. The problems of overspecialization and unfamiliar situations both hinder the capability of applications to handle situations beyond the previous experience of the other user.

There exist definite relationships between the research question criteria and the issues in recommender systems. Knowledge autonomy is the main goal of learning preferences rather than always eliciting them, but there are difficult problems facing autonomy such as the new item problem and the sparsity problem in general. The issues of the sparse instance set criterion are very similar to those raised by the sparsity and new user problem. Recommender systems as a whole handle these quite well via the use of dimensionality reduction approaches, hybrid recommender systems and demographic information. Instance set selection and situational relevance are stressed less as recommender systems are less likely to be distributed and context-awareness is only an emerging topic in recommender systems, although the CCBR Framework [84] is an example of both. The evaluation validity is quite high in general as there are several databases with actual information from users, e.g., the MovieLens datasets for

movie recommendations and the Book-Crossing dataset[2] for book recommendations.

The approach in this thesis is most analogous to a hybrid recommender approach combining content-based and collaborative filtering techniques as it utilizes information from other users but also learns based on features of the instances, which is analagous to context information for the approach of this thesis. However, even though preferences are context information, recommender systems are not quite as useful for predicting context in general. The similarity between the concepts of item and situation is not perfect because of the dynamic nature of situation instances over time. This also makes non-hybrid collaborative filtering approaches less useful. A particularly important distinction for the approach in this thesis is that systems using collaborative filtering techniques also require other users to have an extensive previous experience, which this thesis does not assume (see Section 3.1). Finally, while predicting behavior directly is often possible, in many cases the prediction of the intermediate context information is key to the behavior, e.g., displaying the temperature or indicating the user's location along a route in a navigation system.

2.6 Missing Value Prediction

The final section of works in this chapter deals with missing value prediction in general. While the last section dealt with predicting one specific type of context information, i.e., preferences, this section generalizes to predicting any type of missing value.

There are several different approaches when confronted with missing values [46, 127]. Figure 2.11 shows the four different topics that these approaches fall under. Most of the approaches presented in this thesis have fallen under the missing data imputation topic where a missing value has been directly predicted. Case deletion and machine learning methods for handling missing data are not as useful to this thesis, as they do not predict the missing values. Case deletion utilizes either only instances that are not missing values for any variables (casewise deletion) or only instances that contain values for the subset of variables that the specific calculation needs to function (pairwise deletion). Learning methods that handle missing data were covered in the previous section's discussion on predicting behavior directly (Section 2.5.2). Model-

[2] http://www.informatik.uni-freiburg.de/~cziegler/BX/

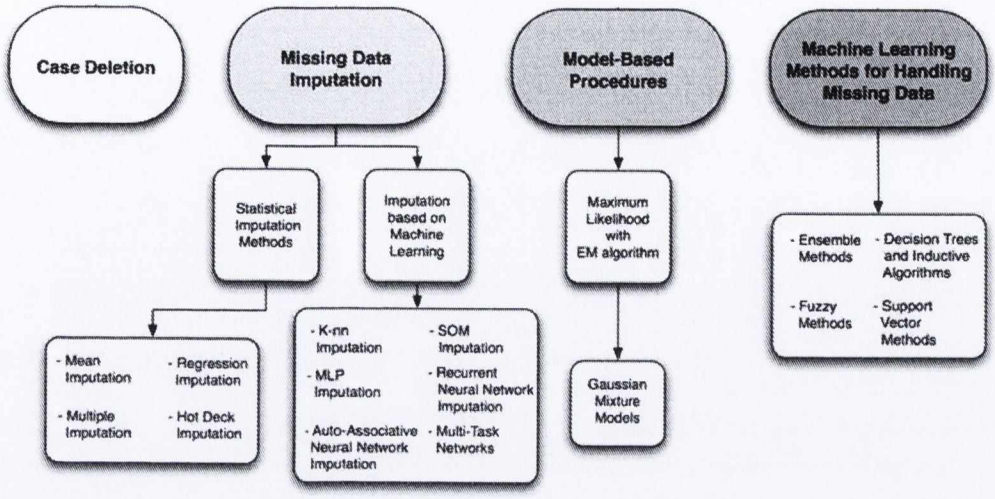

Figure 2.11: General topics for handling missing values (from [46]).

based procedures such as the expectation-maximization algorithm are not knowledge autonomous as they require assumptions about the distribution in order to build the model.

Section 2.6.1 discusses approaches of general data imputation for missing values. The remaining two subsections present topics in missing value prediction that deal with specific aspects of the research questions of this thesis. Transfer learning aids in predicting missing values when the source and target tasks are different (Section 2.6.2), and Section 2.6.3 presents common approaches for predicting missing data in sensor-based applications. As the overall topic of this section is so broad and the individual topics are so diverse, the degree to which each individual topic addresses the research question criteria is discussed within each subsection.

2.6.1 Data Imputation

General data imputation approaches [46, 127] are designed to predict multiple missing members of a dataset so it can be analyzed or used for a further means. The dataset contains rows of attribute vectors where some of the attribute values are missing. For some analysis, the missingness mechanism must be known in order to use a certain method. The *missingness mechanism* is the reason why the data are missing, i.e., missing completely at random, missing at random (the pattern is dependent on other attributes' values), and not missing at random (the pattern depends on the value of

the missing variable). The approaches are divided into imputation based on machine learning methods and statistical imputation.

Statistical imputation methods can vary in complexity. The simplest method is *mean imputation*, and it assigns every missing value the average of its attribute's non-missing values. *Regression imputation* trains a regression model for the missing attribute, and *hot deck imputation* uses the most similar complete row in the dataset to predict the value. In multiple imputation [127] a missing value is estimated M times according to a model that represents the uncertainty about the value to impute. The model can be quite complex and it usually has some relationship to the statistical method being used to analyze the dataset.

Imputation using machine learning methods can be done by any supervised learning methods reviewed in this chapter. Training becomes more complicated if there are multiple missing values, however. If a selected imputation method cannot predict the value with missing predictive attributes, then these values need to be imputed or the method must use only instances (rows in the dataset) that have a value for all attributes. K-nearest neighbor, neural network-influenced methods, and multi-task learning are all examples of imputation methods using machine learning. Multi-task learning [22] is closely related to transfer learning which is described in the next subsection. In multi-task learning, information from two different tasks is used to learn both tasks. In transfer learning, just the target task is learned.

Most of the imputation methods can be used to provide situational relevance. Especially relevant are the supervised machine learning methods that are also pervasive in the rest of the latter sections of the related work, and the performance of some of these will be explored in the thesis evaluations. There are a few reasons why the data imputation topic is not a great fit for this thesis. One of the main differences is that the methods are designed to predict multiple missing values. The methods can be used for predicting a single value, but this is not the primary function for which they were designed. There are also fundamentally different goals between imputation and predicting a missing context value. Imputation is primarily used to fill in the missing values in a dataset so that it can be analyzed. This means researchers have to take into account why the variables are missing (the missingness mechanism) from a dataset in order to select an appropriate imputation approach. Multiple imputation is a prime

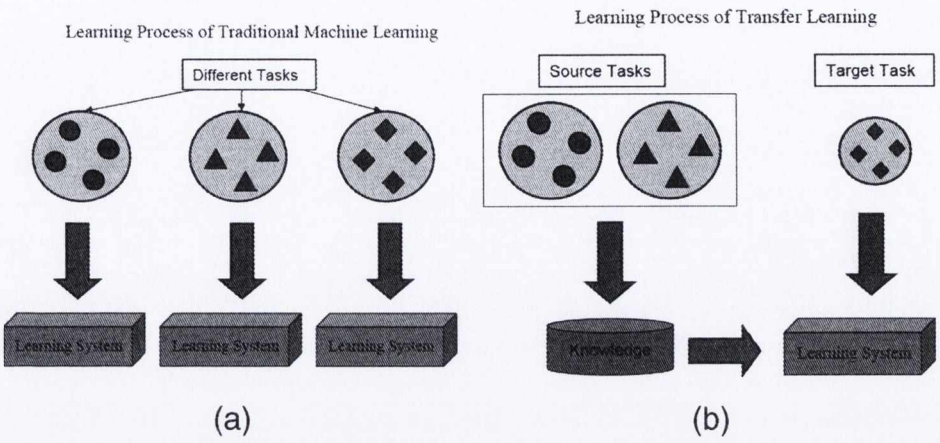

Figure 2.12: Machine learning processes for (a) traditional machine learning and (b) transfer learning (from [100]).

example of the difference in the topics. Multiple imputation focuses on generating several different datasets so that they can be analyzed using the model that generated them. That is, predicting a missing value is not the goal but instead the goal is to have a number of estimates of the uncertainty of that value in order to perform statistical analysis.

2.6.2 Transfer Learning

The research questions of this thesis ask how useful knowledge from one task (tasks of other users or of a user's previous experience) is for predicting a missing value in a new task (the user's current situation). Transfer learning [100] takes place when knowledge learned from one source task is extracted into a different target task. The underlying idea is that some of the knowledge gained from learning and labeling in a previous task should be useful to a new unfamiliar task that is somewhat related. The difference in processes of transfer learning and traditional machine learning (Figure 2.12) is due to the fact that the distributions of the source and the target tasks in transfer learning are not assumed to be the same. The ability to reuse knowledge from previous tasks and identify which instances are most useful for the current task should be especially helpful for the knowledge autonomy and instance set selection criteria, respectively.

There are three relevant types of knowledge that can be transferred [100]. *Instance-transfer* conveys knowledge in instances through instance importance sampling or

reweighting. *Feature-representation-transfer* attempts to learn a low-dimensional feature representation that can benefit both the source and target tasks (with an emphasis on performance for the target task). Finally, *parameter transfer* learns hyperparameters or priors that can be transferred from the source task and used by the target task, e.g., reusing the number of hidden nodes for a neural network or prior probabilities in a Bayesian network. This thesis will focus on works that perform instance-transfer as this is the most common approach in pervasive environments.

TrAdaBoost [32] is a boosting algorithm that modifies the AdaBoost algorithm [43] to perform transfer learning. Boosting is an iterative ensemble method that reweights the importance of an instance at each iteration depending on if that instance is misclassified or not (see Section 3.3.7). TrAdaBoost is trained on labeled instances of both the source task and the target task. Through each training iteration, instances of the source task set that are misclassified by the new learner have their weights decreased. The goal for the last learners is to be influenced only by the target task instances and the source task instances that have the same distribution as the target task. Results of their evaluations show that TrAdaBoost improves error over traditional learning methods when the target instance set is sparse and the errors become similar as the number of the target task instances increases.

The Multi Home Transfer Learning method [112] is an example of transfer learning applied to pervasive computing. The method attempts to use labeled activity context information from other smart homes to label activities in a new smart home in order to save on data collection and annotation time and effort. The method begins by forming "activity templates" that summarize the different activities. Activity templates are formed by either summarizing all activities with the same label or summarizing all activities in a cluster using incremental clustering if the data is unlabeled. A model representing each source smart home learns its activity templates' relationships to the target home's activity templates based on similarity of activity start time, duration, location, and sensors. The probability that a source activity template is the same as a target activity template is learned using a "semi-EM framework" that updates the matrix of inter-activity probabilities until it is stable. The label of the source activity template with the highest probability for an unlabeled target activity template is then selected. The labels from all source homes are combined using a voting ensemble

method [36] that is weighted by the similarity between the source and target homes. Experiments were performed using labeled datasets from six smart homes using a leave-one-out methodology. The results showed that the error rate was around 40% or less for five of the six smart homes showing that labeled data from other smart homes is useful for learning missing activity labels for a target smart home.

Adaptive Transfer learning [21] attempts to automatically determine how much knowledge to share based on similarities in the target and source tasks. If the tasks are too dissimilar, transferring knowledge could result in negative transfer. *Negative transfer* is when using knowledge from the source task hurts the target task's performance [116]. The authors present an Adaptive Transfer learning algorithm based on Gaussian processes (AT-GP). AT-GP learns the similarities between the source and target parameters and assigns a correlation parameter (λ). If λ approaches zero, then the tasks are not seen to be similar and only the shared parameters in the kernel are passed. When the magnitude of λ is in the interval $(0, 1)$ then the training instances of the source task are also shared though with an influence proportional to λ. When λ is one, the source and target tasks are the same and instances of the source and target tasks are weighted equally. The authors test AT-GP on a synthetic dataset and three real-world datasets. For example, one real-world dataset involves Wi-Fi-based location prediction based on received signal strength from Wi-Fi access points. The training and test data were recorded at different times, causing a distribution change. AT-GP was more accurate than the approach using all the instances and the approach using none of the instances from the source task set.

The research questions of this thesis do involve the applicability of knowledge from one source task (those of other users or of previous experience) to another target task (the current situation). The works surveyed deal with the instance set selection criterion as they attempt to discover which instances are most helpful and how much instances from a source task should be weighted, respectively. However, for the problem of this thesis there is not a set of instances representing the current target task (labeled or unlabeled) available to learn the relationship of the source and target tasks or do instance set selection. As transfer learning requires these set of instances, it is of limited use to addressing the research question criteria.

2.6.3 Sensor-based Applications

The research questions of this thesis deal with predicting missing context information for pervasive applications. Almost all pervasive applications deal with some form of sensor data. This subsection presents general approaches that aid sensor-based applications in predicting missing sensor data without the restrictions of being necessarily people-centric and utilizing the current context information from other users as in the main works of the first part of this chapter.

The Multi-Home Transfer Learning method [112] reviewed in the transfer learning subsection is an example of a sensor-based application in a home, i.e., a smart-home. A smart home is one example of a sensor network where multiple, potentially redundant sensors are spread throughout the environment. In general sensor networks, sensors also act as nodes in a multi-hop network to pass their data through the network for aggregation and other processing. When a sensor runs out of battery or otherwise fails, not only is future data from that sensor lost but any communication paths that run through that sensor node are disconnected. Due to the potentially high volumes of sensor streams, unreliable communication, and potentially redundant nature of the sensors, often it is more efficient to estimate a sensor's missing data than to add more redundant sensors or increase energy consumption and query response time by requesting that the sensor resend the data [136, 50]. There are several approaches to predicting missing sensor data, and most involve using sensor data streams where each sensor's event is a member of a time series.

The data estimation framework MASTER-M [50] uses association mining rules to estimate missing data in a multi-hop network. MASTER-M uses both the data stream of the sensor being estimated as well as the data streams of other sensors to employ a multi-attribute, temporal approach that exploits the inherent redundancy of sensors in the network. The framework includes a data structure called MASTER-tree [29] that stores the history of each sensor's readings. Each node in the MASTER-tree represents a sensor and a path through the tree is an association rule among sensors where continuous values are discretized in order to become the discrete "items" association rules require. There is a MASTER-tree for each cluster of related sensors. When a missing sensor value needs to be estimated the trees with that sensor's node as a consequent are searched and the rule with the most confidence is returned. MASTER-M extends

the applicability of MASTER-tree by dynamically re-clustering the sensors when a change in phenomena occurs. As well as grouping sensors whose values are correlated, the clustering method of MASTER-M also provides robustness to node and path failure by favoring groups of sensors which are less likely to be missing at the same time. The evaluation showed that MASTER-M outperformed related approaches and also lowered energy consumption.

The work of Rodrigues and Gama [115] provides a similar approach to predicting sensor streams for the next k timestamps. Variables are also clustered by correlation in an online clustering algorithm, Online Divisive-Agglomerative Clustering (ODAC), although variables here are only allowed to be a part of one cluster. To confront concept drift, diametric statistics are examined periodically and if two leaf clusters in the cluster hierarchy reach a threshold relating to their parent's diametric statistics when the clusters were formed, the two clusters are merged and possibly re-split with a new variable distribution. Instead of association rules, the predictions at each cluster are performed by a feed-forward neural network predictive model that learns the first-order differences. The predictions for k time units in the future are the sum of the k output differences added to the last known value of each variable. In addition, predictions are made for several timestamps in the future so that the error computed when the actual data is received can be back-propagated into the neural network to keep the weights up-to-date.

There are several other means of estimating missing values in data streams, e.g., Kalman filters [136] and auto-regression with the aid of principal component analysis [101]. While sensor networks are useful for recording phenomena in an environment, the research question in this thesis focuses on people-centric sensing. Additionally, most of the techniques require sensor data streams which the approach of this thesis does not assume (see Section 3.1).

2.7 Chapter Summary

The works presented in this chapter represent the state of the art in acquiring current context from remote sources in pervasive environments and of missing value prediction in general. A unique view on how context should be acquired within the restrictions

of these environments is provided by the works of the first four sections. Context selection methods (Section 2.1) attempt to handle the varying level of quality in context information and its efficient retrieval. Proximity-based acquisition methods (Section 2.2) attempt to address the situational relevance limitations of context selection methods by prioritizing values from the devices nearest them. CBR methods for acquiring context from other users in pervasive environments (Section 2.3) handle situational relevance more generally by considering the similarity of all types of context, not just location. Multi-agent CBR methods (Section 2.4) are useful for context acquisition even though not all of them explicitly deal with pervasive applications as their concepts are relevant to increasing knowledge autonomy and performing evaluations with high validity. The reviews of recommender systems (Section 2.5) and general missing value prediction (Section 2.6) expand the scope of works outside pervasive applications even further to consider how a specific type of context information is predicted, i.e., preferences, and how missing data prediction is handled in general.

The extent to which the works of each category support the research question criteria presented in the introduction is summarized in Table 2.2. Instance set selection was generally not dealt with, i.e., the set from which to acquire information was predefined. Exceptions include social circle-prioritization of the Mobile Sensing Group (Section 2.2.1), the similarity threshold-based set selection of MCBR (Section 2.4.1), similar-history comparison of the CCBR framework (Section 2.3.3), and the transfer learning works (Section 2.6.2). However, the former assumes that social network implies relevance without considering the context of the situation. The middle two are ineffective in dynamic environments with unfamiliar situations and concept drift as these works prioritize previous experience. The last exception does not apply to the problem of this thesis, because transfer learning requires a set of target task instances to function.

Knowledge autonomy suffers for most works because of a failure to create an initial, relevant instance set and lack of automatic calibration and recalibration of their underlying methods. In the first part of the chapter, MCBR is the lone exception as it concentrates on acquiring and modifying instances from other users. MCBR also includes autonomous calibration techniques that learn the parameters for their methods from relatively few data instances. However, there is no discussion of learning the

	Instance Set Selection	Knowledge Autonomy	Situational Relevance	Instance Set Sparseness	Evaluation Validity
Context Selection Methods					
Huebscher and McCann	-	-	-	-	-
IST-CONTEXT	-	-	-	-	-
Preuveneers and Berbers	-	-	-	-	-
Proximity-Based Acquisition Methods					
Mobile Sensing Group	O	-	-	-	-
Collaborative Context	-	-	-	-	-
CBR in Pervasive Environments					
The LISTEN Project	-	-	O	-	-
AmICREEK	-	-	O	O	-
CCBR Framework	X	-	O	O	-
Multi-Agent CBR					
MCBR	O	O	O	X	O
CoopCBR	-	-	O	-	O
Recommender Systems	-	-	O	O	O
Missing Value Prediction	O	O	O	-	O

X = supported, O = limited support, - = no support

Table 2.2: Summary of the state of the art for context acquisition from other users.

relative importance of attributes and their static domain does not require run-time recalibration. This limits the applicability of their approach to knowledge autonomy for pervasive applications. Transfer learning from the missing value prediction section deals quite well with knowledge autonomy as it automatically transfers knowledge from another task to the current one. Again, the requirement of multiple instances from the target task hinders its applicability for the research questions of this thesis.

Situational relevance is only handled by the approaches using CBR techniques (Sections 2.3 and 2.4) and some of the works in the preference learning and missing value prediction sections (Section 2.5 and Section 2.6). However, they are all focused on previous experience and not on the current state of the user. As mentioned above, disregarding the timeliness of data leads to inappropriate acquisition in dynamic situations involving concept drift. In addition, the CBR methods require that other users also have a case base, which restricts the devices that the application can utilize. The context-aware recommender systems stress the importance of situational relevance but the area is still emerging and preference prediction does not generalize well to other context types. Finally, almost all the missing value prediction methods focus on situational relevance, and some of the general learning methods employed are reused for the evaluations of this thesis (see below). Most of the approaches are quite general though and the sensor-based applications focus mostly on predicting a value from iterations of previous values over time. This thesis does not assume any structure to its history or that other users will make historical values available, so approaches that depend on data time series are not viable.

Instance set sparseness is considered by subsets of the CBR and recommendation system approaches. AmICREEK (Section 2.3.2) uses knowledge to aid in combating sparseness, but the lack of knowledge autonomy and the static nature of the knowledge makes it inappropriate for general pervasive environments. The CCBR Framework uses the experience of other users if their previous experience instance set is too sparse and uses their own previous experience if no other users in the environment can solve the problem. Unfortunately, their approach to identifying users that can solve the problem requires non-sparse instance sets for both the local agent and the remote agents. This is also a consequence of using one type of context, i.e., the route type. As mentioned in the knowledge autonomy paragraph above, most of MCBR's benefits are due to its

performance in the face of sparse instance sets. However, the calibration of the method to perform the acquisition requires that other users have a case base, further reducing the available users' devices from which to acquire context information and limiting the environments in which they can operate. Recommender systems deal quite well with sparsity, but their solutions do not transfer well to the pervasive computing domain. Content-based recommenders get most of their features directly from text, and other approaches use other users along with demographic information.

Finally, most of the evaluations have limited external validity for general pervasive environments. The majority are simulations with seemingly arbitrary generation of data. Those that do use data sensed from pervasive environments only perform evaluations for one, often very simple, problem. The multi-agent CBR evaluations of Section 2.4 were of high validity, but their approaches were not applicable to pervasive environments. The proximity-based acquisition methods of Section 2.2 avoided some of the problems of external validity that plague the remaining works by using real data sensed from pervasive environments. However, their overall external validity is still limited due to testing a single dataset representing one problem and the experimental setups that do not generalize well beyond their problem. The validity of the surveyed recommender system and missing value prediction works is quite high as each domain has many realistic datasets available; however, the datasets do not have the properties necessary to answer the research questions in this thesis and minimal statistical analysis was provided overall.

As stated in the introduction to this chapter, the degree to which the related works address the research question criteria is the degree to which the research questions of this thesis have already been addressed. As the related work provides little or no support for the criteria, the conclusion is that the research questions have not been addressed. In addition, the specific manner in which the related works are limited in their handling of the criteria will influence the study inputs, experiment design, and result analysis choices in the subsequent chapters.

The approach suggested by this thesis acquires more accurate context by acquiring from both the current context information of other users and the application user's previous experience. The approach addresses the limitations of existing related works and is specialized to function in pervasive environments. Learning methods that exhibit

situational relevance are evaluated. Situational relevance is handled by using all the available attributes for the selected learning methods. Instance set selection is flexible as it can use both other users' information and the previous experience of the user. As the current context of other users is utilized, the difficulties presented by concept drift over time are minimized. There are negligible knowledge requirements as the approaches learn from existing instance sets, whether from the previous experience of the application's user or the current situations of other users. Learning can be repeated at any interval to incorporate more recent instances so that concept drift has less effect on accuracy. Additionally, the other users are not required to have a set of previous experience like the case base requirements of the multi-agent CBR approaches in Section 2.4. This relaxation expands the number of user devices from which to acquire information. The approach in this thesis manages sparse environments by using the previous experience of the application's user if there are not enough users in the environment. Finally, the evaluations of the approach maximize validity within the logistical restrictions of producing datasets in pervasive environments. Similar to the dataset combination in the evaluation [58] of Collaborative Context Recognition (Section 2.2.2), the evaluation of this thesis combines single user datasets sensed from real pervasive environments to form a multiuser dataset (see Section 3.2.4), increasing external validity. Unlike their work, the external validity of the majority of the data is not compromised by only having datasets with users performing the same exact activity at the same time. To further increase external validity, the evaluation tests across several datasets representing different problems, runs several experiments within each dataset, and analyzes the accuracy of estimating values for every attribute type in a dataset (see Section 4.1). Additionally, the accuracy of the different alternatives is compared using statistical analysis to maximize conclusion validity (see Section 4.3).

The closest competitors are those works in the multi-agent CBR section. However, two of the three works are not focused on pervasive environments and the third (CCBR) concentrates on a very specific, single-context approach, i.e., personalized route planning. The approach of this thesis meets all the limitations of these works, as shown in Table 2.2 and discussed above.

This chapter also provides an account of the learning methods used by the related works to answer the questions deemed most similar to the ones of this thesis.

CBR, support vector machines, decision trees, ensemble methods, neural networks, and Bayesian learners are all common approaches in the state of the art covered in this chapter. These six learning algorithms will be explored in the next chapter, and the ones that are most suited for answering the research questions will be selected for the evaluations.

The next two chapters describe the approach presented by the study in this thesis in detail. Chapter 3 discusses the inputs into the study, i.e., the datasets and learning methods used in the evaluation, and Chapter 4 discusses the design of the experiment.

Chapter 3

Study Inputs

The last chapter presented the degree to which the related work satisfied the research question criteria. These criteria represent the degree to which the research questions presented in Section 1.2.3 have been addressed in the related work, and the previous chapter concluded that they have not been sufficiently addressed. As the related work is unable to sufficiently answer the research questions, this is the focus of the thesis.

The structure of the next three chapters is guided by the system design cycle presented in the second edition of the classic textbook by Duda et al. [37] extended to suit the particulars of this study. The original intention of Duda et al.'s design cycle is to guide the design of a system that selects a learning method implementation and data on which the learning method is trained to best meet the requirements of that system. As an extension of Duda et al.'s design cycle that reflects the terminology used in this thesis, the *learning method evaluation process* is introduced. The process is sufficient as a guideline to designing this thesis' study, and its steps are presented in the leftmost column of Table 3.1. The research question criteria addressed by each step of the process and the section in which they are addressed are listed in the subsequent columns. The criteria relate directly back to the research questions (as summarized in Table 2.1). When the last step of the process is completed, evaluations will have been performed and valid conclusions will have been drawn that fully answer the research questions.

By the end of this chapter, the selection of the inputs into the study will be explained and defended. Specifically, this chapter describes the selection of the data on which the experiments are run and the machine learning methods used to evaluate the

Process Step	Research Question Criteria Addressed	Thesis Section Addressed
Select data	Evaluation validity (external validity).	Multiuser Datasets (3.2)
Choose attributes	Evaluation validity (external validity).	Existing Datasets (3.2.2)
Choose learning method	Situational relevance. Knowledge autonomy. Instance set selection. Sparse instance sets.	Selected Learning Methods (3.3) Learning Method Implementations (3.4)
Train model	Instance set selection. Sparse instance sets.	Dataset Partitioning (4.1) Study Evaluation (5)
Evaluate trained model	Evaluation validity.	Analyzing Results (4.3) Study Evaluation (5)

Table 3.1: Criteria addressed by each step of the learning method evaluation process and each step's corresponding thesis sections.

utility of other users' information in pervasive applications. First, the scope of the evaluations and the assumptions of the environments in which they run are presented in Section 3.1. Then, the "select data" and "choose attributes" steps are completed by the selection of the datasets and attributes that best satisfy external validity in Section 3.2. Next, the single-user conversion procedure used to generate multiuser datasets is presented in Section 3.2.4 and the threats to the validity of the procedure are listed and addressed in Section 3.2.4.1. The procedure defined and defended in this section successfully addresses the first research question (RQ-1) of this thesis (see Section 1.2.3). The other five research questions will be addressed by the evaluations of Chapter 5. Lastly, the "choose learning method" step is performed by surveying existing learning methods in Section 3.3 and by providing the implementation details of the learning methods that help best meet the remaining criteria in Section 3.4.

The final two process steps will be addressed in the experiment setup chapter (Chapter 4) and study evaluation chapter (Chapter 5). That is, the "train model" step applies the selected learning method's algorithm on the selected datasets to train the associated model. The partitioning of the dataset (Section 4.1.1) for use in training the model addresses the instance set selection and sparse instance set criterion. Additionally, the "evaluate trained model" step will run the evaluations and analyze the results to maximize validity while addressing the remaining research questions.

3.1 Environment Assumptions and Evaluation Scope

To increase the external validity of the evaluation results, this thesis assumes as little as possible about the devices of other users and the environments in which the users operate. These assumptions affect the selection of the datasets and learning methods that compose the remaining sections in this chapter.

- *People-centric*: People-centric sensing is sensing where the user is the focus of the sensing and the sensing is done by devices and sensors on the user [20]. This is in contrast to infrastructure-based sensing where the sensors and reasoning are in the environment, e.g., a smart home [55]. This thesis assumes a people-centric

view, allowing the application to be free from the necessity of being located in an area with a sensing infrastructure.

- *Heterogeneous devices*: Pervasive applications can run on heterogeneous devices [10]. To generalize the results of the evaluations as much as possible, this thesis assumes very little about the devices of other users in the environment.

  - *Heterogeneous learning and reasoning methods*: There is no assumption of the type or presence of reasoning or learning methods on other users' devices. For example, the multi-agent CBR approaches described in Section 2.4 assumed that all the other agents were running a specific CBR approach. Assumptions like these limit the devices from which the application can receive useful information.

  - *No previous experience assumed for other users*: There is no assumption on the structure or presence of previous experience on other users' devices as there is in the multi-agent CBR systems (Section 2.4). The only requirement is the availability of a single instance representing the current situation of the user.

  - *Heterogeneous communication types*: As long as devices are able to communicate in some manner, this thesis does not restrict the method of communication, e.g., restricting communication to Bluetooth or Wi-Fi. For example, the proximity-based methods of Section 2.2 assume devices that communicate are in the same location and situation as they are communicating using short-range communication technology.

- *Homogeneous representation of instances*: This thesis assumes that the learning methods receive the same representation for all the training instances, i.e., the same attributes with the same representation for each instance. This does not require that the other users have the same representation or that they have identical sensors, however. Common methods to unify different representations are to use ontologies which relate similar concepts (e.g., the AmICREEK approach [64]) or to convert the different lower-level context information into higher-level context information of the same representation (e.g., the Mobile Sensing Group's

approach [68] to deriving location from several different lower level location sensors). In addition, transfer learning [100] can be used to transfer knowledge from two tasks with different attribute types (see Section 2.6.2). To deal with missing sensors or attribute values, common techniques are learning with the smallest intersecting subset of attributes (e.g., as suggested by the Mobile Sensing Group's approach), data imputation techniques [46, 127] (see Section 2.6), and using learning methods that are tolerant to missing values (see Section 2.5.2).

The scope of the evaluations is focused on improving the accuracy of context acquisition. There are several other issues that are equally important to pervasive applications that are beyond the scope of this thesis and these are discussed in the future work of Section 6.2:

- Logistical issues such as the efficiency and scalability of the approach, e.g., performance with increasing number of users and training instances.
- Privacy and trust issues for both sharing situation instances with other users and using other users' information.
- The uncertainty of sensed and derived context information [52].

3.2 Multiuser Datasets

The external validity of the datasets used in the evaluation is the focus of this section. The first two steps from the learning method evaluation process listed in Table 3.1 are also performed, i.e., "select data" and "choose attributes". The properties that the selected datasets must have to maximize external validity while also allowing the research questions of Section 1.2.3 to be addressed are listed in Section 3.2.1. A survey of existing pervasive datasets is performed in Section 3.2.2 along with an examination of how well each dataset satisfies the required properties. The section also divides the datasets into the scenarios they perform and discusses which attributes of the datasets are selected for the evaluations, completing the "choose attributes" step. Finally, the datasets selected for the evaluation are justified in Section 3.2.3 and the procedure that transforms the selected single-user datasets into multiuser ones is presented in Section 3.2.4. The threats to validity introduced by this procedure are also presented in this

Property	Definition
Multiuser, simultaneous	A dataset with multiple users performing at the same time.
Multi-attribute	A dataset where each situation is described by more than one attribute type.
Multi-scenario	A dataset where users encounter different situations and perform different activities than each other at any particular time.
Chronological	A dataset that records each user's situation at a specific time interval over a duration of time.
Inter-attribute relationships	A dataset where the relationship among the different attribute types is realistic.
Multi-run	A dataset with multiple runs. These are not necessarily recorded simultaneously.

Table 3.2: Definitions of the dataset properties.

section (Section 3.2.4.1) and further addressed in a post-evaluation discussion (Section 5.7). This completes the "select data" step of the learning method evaluation process. The single-user conversion procedure addresses the first research question (RQ-1) of this thesis (see Section 1.2.3), which relates to identifying a procedure for generating realistic, multiuser datasets.

3.2.1 Dataset Properties

To maximize the validity of the evaluation, the following properties of the dataset should be satisfied where possible. The scarcity of multiuser datasets requires inclusion of the last two properties (multi-run and inter-attribute relationships), which would not be required if the first two properties were satisfied, i.e., if the datasets were multiuser and multi-attribute. The dataset properties and the motivation for their inclusion are:

Multiuser, simultaneous Ideally, to evaluate the utility of other users' information, the datasets should contain multiple users acting in the same pervasive environment at the same time. This aids in external validity as the shared environment

context information, e.g., temperature and humidity, is similar. In addition, the probability of a user performing a particular activity at the same time that another user is performing a particular activity is actually observed. Multiuser, simultaneous datasets are rare because they are prohibitively expensive to deploy (see Section 1.2.2).

Multi-attribute Almost all the works discussed in Chapter 2 tested their approach with multi-attribute datasets, i.e., there were multiple types of context information. This more accurately reflects pervasive environments where many types of context are available from sensors and data stores. The presence of multiple attributes also allows a richer description of the situation. Additionally, the use of multiple attribute types allows applications to function even if a source for one attribute is unavailable. For example, one of the few works discussed that had a single context type was the CCBR framework [83] in Section 2.3.3. This required a specialized similarity measure and made acquiring personalized routes impossible when previous experience was sparse.

Multi-scenario To aid in external validity, there should be multiple scenarios within the dataset to reflect the typical pervasive environment where at least some of the users are in different situations. For example, the generalizability of the evaluation [58] of Collaborative Context Recognition in Section 2.2.2 suffered from only testing the approach when all users were in the same situation, i.e., performing the same activity.

Chronological The existence of previous experience is necessary to answer the research questions of Section 1.2.3 comparing using other users' information to instance sets including previous experience (RQ-3, RQ-4, and RQ-6) and the instance set selection between the sets (RQ-5). Specifically, the dataset must be separable into two subsets, i.e., the previous experience described by earlier instances and the later test instances representing potentially unfamiliar situations. Realistically modeling how situations change over time aids external validity by allowing the estimation of current context from the experience that actually preceded it, i.e., a user's current situation is realistically plausible given the user's experience preceding the current situation.

Inter-attribute relationships A major impediment to the external validity of simulations is the lack of realistic relationships among attributes (See Section 1.1.3). This property is satisfied with a multi-attribute, multiuser, simultaneous dataset.

Multi-run Because multiuser, simultaneous datasets are prohibitively expensive to create, multiple runs taken at different times can be considered to take place simultaneously for testing purposes. This was the case for the evaluation [58] of the Collaborative Context Recognition in Section 2.2.2. Unfortunately, with this simultaneous assumption, the external validity is not as high as with a multiuser, simultaneous dataset for reasons discussed in Section 3.2.4.1.

3.2.2 Existing Datasets

The ability to replicate an experiment plays an important role in furthering the understanding of a domain, both for verifying an experiment and for evaluating new methods against existing ones [61]. The datasets utilized in experiments must be made available to other researchers in order for the goals of replication to be met. For evaluating machine learning methods in general domains, the UCI Machine Learning Repository [8] maintains almost 200 datasets across several domains on which different approaches can be evaluated. Unfortunately, exacerbating the already difficult task of deploying a pervasive application to produce a dataset, the researchers of a majority of the applications that are deployed do not publish their datasets. An initial attempt to rectify this problem for pervasive environments is the collection of datasets called the Context Database.[1] This collection was introduced in a workshop at Pervasive 2004: Benchmarks and a Database for Context Recognition.[2] Most of these datasets track the context information of a single user performing a scenario, such as travelling along a path or performing a sequence of tasks. Often the user repeats the scenario a number of times, with each repetition referred to as a run. The dataset then either has a single user performing the different runs and scenarios, or it has multiple users doing so (although not simultaneously). All five datasets available in the Context Database are

[1] http://www.pervasive.jku.at/Research/Context_Database/

[2] http://www.pervasive2004.org/program_w5.php

Time	Gyroscope	Air Pressure	Resistor #1	...	Activity Label
0	-0.305	969.4	-1.8171	...	1
1	-0.188	969.4	-1.8183	...	1
2	0.073	969.4	-1.8159	...	1
...	...	...	...	...	...
38500	0.258	969.4	-1.8122	...	1

Table 3.3: Portion of a run from the Locomotion dataset.

reviewed in this subsection as well as additional datasets from the Helsinki University of Technology and the Massachusetts Institute of Technology (MIT).

For each dataset, there is an overview followed by a description of how the dataset would be used if it were selected for the evaluations. This is accomplished through a description of the available attributes, selection of a subset of those attributes for the evaluations, and information on how the runs and scenarios are assigned for the evaluations. The selection of the attributes completes the "choose attributes" step of the learning method evaluation process.

The datasets in this subsection are people-centric (see the environment assumptions in Section 3.1). For this reason no datasets for smart environments (e.g., datasets from smart homes in [134][3] and [59][4]) or location tracking were included in the survey (e.g., [137][1]).

3.2.2.1 Locomotion Dataset

Overview The Locomotion dataset [60] from the Wearable Computing Lab is the first of the datasets to be reviewed from the Context Database. The users in the dataset were fit with sensors attached to their body (e.g., a gyroscope, air pressure sensors, and accelerometers) and a run recorded a single user following a predefined path. There are four users but they do not perform the tasks at the same time, so the multiuser, simultaneous property is violated. The dataset contains multiple runs for each user, however, so the multi-run property is satisfied.

A portion of one of the runs in the Locomotion dataset is shown in Table 3.3.

[3]http://courses.media.mit.edu/2004fall/mas622j/04.projects/home/

[4]http://architecture.mit.edu/house_n/data/PlaceLab/PlaceLab.htm

The names of the attribute types are listed across the top and the values at each time iteration are listed below them. A new iteration occurs every 100th of a second. The presence of a snapshot of the values for all the attributes at each time iteration coupled with the fact that there are multiple attributes means that the chronological, multi-attribute, and inter-attribute relationships properties are all satisfied.

Attribute Description and Selection All the attributes are continuous real numbers except for the activity label. The activity label is categorical and represents level walking (1), ascending (2), and descending (3). The numerical attributes are gyroscope reading, air pressure, two pressure sensors under the right foot, three accelerometers above the right knee, and three accelerometers on a belt. Each sensor in the trio of accelerometers measures acceleration in a different orthogonal direction, i.e., front-to-back, left-to-right, and top-to-bottom. The timestamp is not included in the attributes, so there are 11 attributes total.

Scenario and Run Information

- The dataset has only one scenario for all the users. However, variety in the dataset is maintained as each user has a distinctive manner of walking.
- In total there are 4 users and each user has two runs except for user number 4 who has seven runs. There are therefore a total of 13 runs in the dataset.

3.2.2.2 Nokia Context Data

Overview The Nokia Context Data dataset [41] included in the Context Database contains 43 different recorded runs of the same user over the course of several weeks as that user travels to and from the workplace. The data was recorded by sensors carried around by the user in a sensor box, a laptop, and a mobile phone. The subject uses many different forms of transportation and occasionally varies the route slightly or drastically.

Like the Locomotion dataset, each run contains a snapshot of all the attributes for each second. The Nokia dataset satisfies all of the dataset properties that the Locomotion dataset satisfies in addition to the multi-scenario property.

Attribute Description and Selection There are several temporal attributes in the dataset, e.g., day, hour, and second. Location is given by the current Cell ID and the current Area Code in which the mobile phone is located. The remaining attributes are from sensors: Temperature, Relative Humidity, Dew Point, Pressure, Device Orientation, User Activity (measured by an accelerometer), and Audio Level.

All the attributes that are not temporal were selected for a total of nine attributes. All attributes are represented as integers, where most are numerical but some are categorical (i.e., Device Orientation, Cell ID, and Area Code). The attributes are also not in their original scales but are transformed instead, e.g., temperature is represented with the coldest value as 1 and the warmest as 10.

Scenario and Run Information

- One of the runs is significantly shorter than the rest (i.e., run 34). This run was omitted from the evaluation dataset in order to keep a uniformity in the dataset partitions described in Section 4.1.
- The dataset is divided into two scenarios: *From Work* and *To Work*. The 22 runs used for the *To Work* scenario take place in the morning and are the odd numbered runs from runs 1-19 and runs 22, 24, 25, 27, 29-33, 36, 38, and 41. The 20 runs used for the *From Work* scenario are all the other runs except for run 34, which was omitted as discussed in the previous bullet point.

3.2.2.3 Other Datasets from the Context Database

The Locomotion and Nokia datasets come from the Context Database. The remaining three datasets from that database do not fit the properties as well as the first two. Their descriptions and reasons for exclusion are briefly summarized as follows:

- *Augsburg dataset*: This dataset from the University of Augsburg tracks the location of several users through an office building for both the summer and the fall seasons [137]. The dataset only uses one context type (i.e., location) and so is not suitable for the evaluations of this thesis.
- *Lancaster dataset #2*: This dataset from the University of Lancaster tracks the activity of a single user who has 40 accelerometers attached to his trousers

[67]. While there are several attributes, the dataset contains a single user who performs only one run. This violates the multi-run dataset property necessary for the evaluations of this thesis.

- *Lancaster dataset #1*: This dataset is similar to Lancaster dataset #2 except this dataset uses only 2 accelerometers instead of 40 and is likewise not suited for the evaluations of this thesis.

3.2.2.4 Helsinki Dataset

Overview The Helsinki dataset [89][5] is from the Neural Networks Research Centre at the Helsinki University of Technology. The dataset contains five scenarios that include several actions (i.e., walking, stopping, sitting, using an elevator or stairs, and talking) performed in a specific sequence and at specific locations. The raw data comes from a custom-built sensor box containing a three-dimensional accelerometer, illumination sensor, thermometer, humidity sensor, skin conductivity sensor, and microphone.

Like the Locomotion and Nokia dataset, each run contains a snapshot of all the attributes for each second. The Helsinki dataset satisfies all the properties that the Nokia dataset satisfies. The dataset is different than the others in that its attributes are fuzzy variables. These fuzzy variables are considered as regular attributes whose confidence must be predicted. This is not much different from the transformation to a different scale in the Nokia dataset or to the higher-level context in the Locomotion dataset, i.e., the activity label.

Attribute Description and Selection There are 29 attributes that do not describe the time, identify the run, or identify the scenario. These attributes are used to display confidence in the categories of ten different variables. For example, the Temperature variable has the attributes representing the confidence in the categories Hot, Warm, Cool, and Cold.

Several attributes were removed and there were two reasons for doing so. Firstly, two of the attributes do not change at all for the entire dataset. The Temperature attributes Hot and Cold always have confidences equal to zero. Secondly, some attributes are direct complements (i.e., the values always sum to one) of each other and

[5] http://www.cis.hut.fi/~jhimberg/contextdata/

the knowledge of the values of a subset of the attributes would automatically allow the exact prediction of the remaining attribute's value. For example, the Stability variable's attributes Stable and Unstable always sum to one. There are four such variables, so the attributes Unstable for Stability, Cool for Temperature, Dry for Humidity, and Loud for Sound Pressure were removed. This leaves 23 attributes on which to run the evaluations.

Scenario and Run Information

- There are five different scenarios each of which is run between 44 and 50 times for a total of 241 runs.
- There runs are performed by two different users, but the runs are not labeled by user so runs are not discernible by user.

3.2.2.5 MIT's Reality Mining Dataset

Overview MIT's Reality Mining Dataset [39][6] represents approximately 450,000 hours of context information about 100 subjects at MIT over the course of the 2004-2005 academic year. Unlike the previous datasets, this is a multiuser simultaneous dataset with users recorded at the same time although not necessarily at the same location. Data was collected through a user survey and from a Nokia 6600 smart phone which recorded the users' activities. It contains a wide variety of different types of users (incoming freshman students, incoming master's students, continuing students, and faculty) in a variety of different situations (e.g., studying, working, and extracurricular activities).

Attribute Description and Selection While there are several static attributes, there are very few dynamic attributes. Location (given by the mobile cell id) is the only dynamic attribute available for the entire time the dataset is being recorded. The rest of the attributes are only available when events occur, e.g., caller ID or type of communication when communicating using the device. This gives a collection of binary attributes (i.e., whether an event is taking place) and a set of attributes that rarely get set. The dearth of interesting dynamic attributes and the fact that there are no

[6] http://reality.media.mit.edu/

environmental attributes means that the multi-attribute property is not completely satisfied.

The event-related attributes are Using Software, Device in Bluetooth Proximity, Phone Number of Communicating Device, Type of Communication, and Direction of Communication. 21 static attributes describe the users' details such as Subject Name, Phone Number, Device ID, Device Name, Position (at the university), Home Neighborhood, Normal Work Hours, and other personal assessments from a survey, e.g., Forgotten Phone Frequency and Illness Frequency. There is also location information given by Cell Tower ID and associated Cell Name (if available). The total number of attributes for the MIT dataset is 28.

Scenario and Run Information

- There are 100 people who participated in the study for nine months, so there are potentially very many runs depending on how a run is defined.
- A portion of the data was corrupted, and several users did not always have the fully charged phone with them. This leaves some variation in the number of runs for each user.
- Unlike the other datasets, no specific task was set for the users of the devices. There are still a potentially large number of scenarios although they are not currently categorized.

3.2.3 Dataset Selection

The goal of this subsection is to justify the selection of a small group of the datasets presented above on which to run the evaluations in Chapter 5. None completely satisfy all the properties, but simulation is not an option because of complexity of how the attributes change over time, the uncertainty of the values, and the relationships among the multiple attributes at a specific point in time (Section 1.1.3). A modification to how the dataset is interpreted that will allow the selected datasets to meet all of the required dataset properties is presented in the next section. The procedure to modify the datasets also addresses RQ-1 of Section 1.2.3.

	Multiuser, simultaneous	Multi-attribute	Multi-scenario	Chronological	Inter-attribute relationships	Multi-run
Locomotion Dataset	-	X	O	X	X	X
Nokia Dataset	-	X	X	X	X	X
Helsinki Dataset	-	X	X	X	X	X
MIT Dataset	X	O	X	X	X	X

X = supported, O = partial support, - = no support

Table 3.4: Summary of dataset applicability.

While the MIT dataset is the only dataset that is "multiuser, simultaneous", the lack of interesting dynamic attributes for context acquisition make it less ideal for answering the research questions in this thesis. Specifically, there are no shared environmental attributes and, excepting location, there are only binary dynamic attributes that rarely get set.

The first three datasets were selected as inputs into the evaluation of this thesis. These three datasets satisfy the dataset properties better than the MIT dataset as they have environmental sensors and several dynamic attributes. For example, the ambient temperature or accelerometer readings of one user are interesting for another user attempting to predict the temperature or activity. The exception here is the Locomotion dataset which violates the multi-scenario property. However, it does have multiple users and the specific task being measured is very specific to a user (the user's gait), so the different users exhibit quite different sensor readings which gives it the quality of multi-scenario datasets.

The use of these three datasets places some limitations on the properties of the learning methods required to run evaluations with these datasets. A summary of each dataset is shown in Table 3.5. All three of the datasets have multiple attributes, so

Dataset Name	Scenarios	Runs	Average Iterations	Attributes (Categorical)	Description
Nokia	2 (commute direction)	42	5,132.1	9 (3)	User commutes
Locomotion	4 (user)	13	34,307.5	11(1)	User travels fixed path
Helsinki	5 (action sequence)	240	191.9	23 (0)	User performs a sequence of actions

Table 3.5: Information for each dataset.

a learning method must handle multivariate input. Two of the three datasets have multiple types, so a learning method must be able to handle inputs and outputs for both categorical and numerical attributes. These requirements are a subset of those used to select the learning methods for the evaluation (see Section 3.3).

3.2.4 Single-User Conversion Procedure

In addition to the fact that none of the three selected datasets is multiuser, simultaneous, they also contain no static attributes. The lack of static attributes removes a property that would be interesting to evaluate, e.g., for the difficulties arising from the learning of the relative importance of an attribute that rarely or never changes (see Section 6.2.3).[7] At the same time, the lack of static attributes also has the advantage of allowing each run to be treated as a different user, which is how this thesis addresses the scarcity of acceptable multiuser datasets.

The procedure to create a multiuser dataset from several single-user runs is to assume that each separate run of a dataset is a different user. It is also assumed that the users perform simultaneously in a single environment by assigning each user

[7]The degree to which users are willing to share static information such as name and age may make this information unavailable for sharing in any case. See Section 6.2.1 for further discussion.

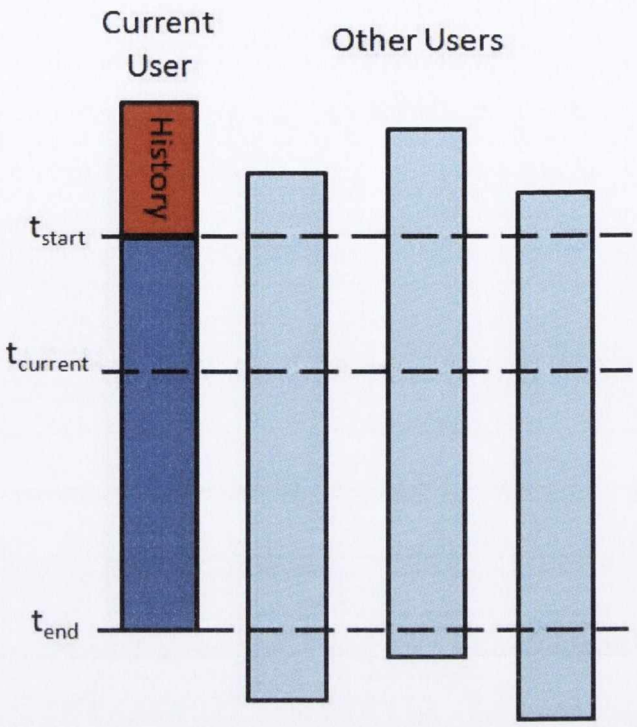

Figure 3.1: Other users' runs are randomly shifted relative to the current user's run.

a time within the run as a start time and the same time iterations for recording attribute information. For example, the situation of each user at time iteration 3 is represented by the context attributes in the third row after its assigned start time for each corresponding run.

To increase the validity of the datasets, the start time of the runs other than the one being tested is shifted by a random percentage. Figure 3.1 shows how the runs of other users are randomly assigned an earlier start time, while the run being tested keeps its start time (t_{start}). This random shift in start time attempts to eliminate any bias that might exist if all the runs started simultaneously. This randomization scheme decreases this bias and allows the maximum number of test iterations. It also keeps each run consistent with any situations that came before the current one, i.e., it preserves the chronological changes in attributes that are difficult to validly model in simulations. This thesis refers to treating the single user runs as different users acting simultaneously and then randomizing the runs' start times as the *single-user conversion procedure*.

The procedure overcomes the three difficulties of simulations presented in Section 1.1.3. The external validity of simulations is hindered by a lack of realistic inter-attribute relationships, chronological changes in attributes, and modeling of the uncertainty of those attributes. These simulation disadvantages are not problems in the single-user conversion procedure, as the within-user attribute relationships were taken from actual users' sensors in pervasive environments recorded over time. As mentioned in Section 1.2.2, solutions to these three problems are imperative in order to run valid evaluations that answer the research questions of this thesis, and they have been significantly counteracted by the single-user conversion procedure. As such, the single-user conversion procedure addresses the research question of how datasets can be created that do not suffer from these three problems (i.e., RQ-1).

3.2.4.1 Threats to Validity

As detailed in the last subsection, there are many reasons why the single-user conversion procedure has better external validity than randomly simulated multiuser datasets. However, there are still reasons why they are not as externally valid as a multiuser dataset. These threats to validity all concern potential biasing (favorably and unfavorably) the accuracy of methods that use the set of other users. This is also the case when we consider both the set of other users and history combined, as any potential bias from other users is also present in this combined set. It is important to again note that due to the result of applying the single-user conversion procedure, the set of previous history is biased neither favorably nor unfavorably in the evaluations.

Specifically, the threats to validity come from the three artificial aspects of the dataset conversion procedure. First, the situations being sensed are taken from different times but the runs are assumed to happen at the same time. Second, the combination of runs assumes that at least some of the users are performing the same scenarios at the same time. Third, the procedure assumes that all the runs describe different users when all or some of the runs are potentially from the same user.

In the remainder of this section, these three threats to validity are presented and the degree to which they bias the set of other users for the evaluations in this thesis is shown through discussion. The section also introduces experiments run on the datasets to discover the degree of the bias introduced by the conversion procedure.

These experiments and the implications of their results to the validity of the specific evaluations of the thesis are discussed in the post-evaluation Dataset Validity Analysis section (Section 5.7).

Shifting the Time of the Runs The single-user conversion procedure takes runs performed at different times and different days, and treats them as runs that occur at the same day and time. The shifting of the time of the runs biases the set of other users in two ways.

Firstly, treating runs from different times as if they occurred at the same time creates an artificial disparity in the shared environment attributes. If the runs were actually being recorded by users in close proximity who were experiencing the environment at the same time, the shared environment attributes such as temperature and air pressure would tend to be very similar. However, because the runs are not recorded simultaneously, the users are not experiencing the same phenomena, and the difference in sensor readings between each run reflects that disparity.

For the evaluations of this thesis, this disparity leads to a problem in both predicting the value of similar environment attributes and using those attributes to predict the values of other attributes. This creates a bias against the set of other users' information. However, any disparity in shared environment context in the datasets derived from the single-user conversion procedure is seen as a worst-case scenario for the set of other users. As the output and input attributes will be further off than would normally be the case in an actual multiuser environment, any results that favor the remote sources for accuracy involving these attributes are conservative and do so in spite of this unfavorable bias towards the set of other users.

Secondly, there is a potential bias of shifting the time of the users because the procedure might not accurately model the interaction between the users, i.e., how users affect their environment and each other's situation. While interactions are interesting phenomena to study, they are not important to a number of tasks. This bias is not a great concern for the evaluations of this thesis, as the scenarios of the datasets involve minimal interaction with other users, i.e., performing a sequence of individual actions and travelling to and from work.

Users Performing the Same Scenario The second aspect of the dataset con-

version procedure that creates a potential bias for the set of other users is that the collection of runs contains users performing the same scenario. Users that perform the same scenario starting at around the same time would be more likely to have similar values for attributes, e.g., the accelerometer readings of two users walking in the Helsinki dataset are potentially more similar than when one user is walking and the other is climbing stairs. The supervised learning methods that use the input-output attribute relationships to predict the output attribute's value would tend to make more accurate predictions if these input-output relationships in the test instance also existed in the training instances.

The potential similarity of attribute values between the current user's situation (test instance) and other users' situations (training instances) is the motivation for using other users' information to improve accuracy. This similarity between the scenarios of the current user and at least some of the other users is what the evaluations in this thesis assume. The presence of similar users doing similar scenarios is not an anomaly among everyday tasks. For example, common group activities include spending time with friends and family, collaborating with coworkers, or travelling along the same route as strangers. In the Nokia dataset, the commuting user is in communication with other commuting users as would be the case on any weekday morning on a busy city street.

It is important, however, to ensure that an artificial similarity does not exist between users that would not be present in reality. As users rarely do the exact same thing in perfect synchronization, the runs of the same scenario should have similar values for their attributes but the attributes' values should not be identical. One way to test this is to explore the average difference of attributes between users running the same scenario.

Appendix C.3 presents experiments that show how similar the runs of the same scenario actually are in the generated datasets. These experiments were run for all three datasets resulting from the single-user conversion procedure without the randomization of start time. That is, all the runs start at timestamp 0 and continue on. This provides a worst-case scenario for demonstrating the average difference, as all the scenarios start at the same time, instead of randomly as the single-user conversion procedure dictates. The graphs show the average difference of attributes for other users

running the same scenario, different scenarios, and all scenarios. The graphs also show the average difference from the history of the current user as well as the difference from its most similar run.

For all datasets, the results show that the difference is much greater than zero for runs of the same scenario. This is the case even when the randomization of start time is not present and when each run is only compared to its most similar run. This indicates that while two runs may be a recording of the same scenario, they have far from identical attribute values which could be the case in most groups performing roughly the same task.

Finally, the objection to the presence of similar runs is also mitigated by the presence of runs from other scenarios, i.e., there are several users doing potentially very different tasks which brings a variety to the new multiuser dataset common to most real-world environments. For example, there will likely be other users doing similar and different tasks in an office building at the same time, so the multiuser datasets generated from the single-user conversion procedure for the Locomotion and Helsinki datasets represent plausible situations. The graphs of the average differences between runs in Appendix C.3 reflect that the average difference between runs increases when other scenarios are added, i.e., the TOTAL alternative's error is greater than the IN_SCENARIO's alternative. While the Locomotion dataset has only one scenario, the Nokia and Helsinki datasets have multiple ones. This is an improvement on the evaluation for Collaborative Context Recognition (Section 2.2.2) where the evaluation of the utility of other users' information comes only from users that are performing the same activity [58]. For a discussion of how the introduction of other scenarios affects overall accuracy, see the experiments in Appendix C.4 and the discussion in Section 5.7.

Treating the Runs of the Same User as Runs from Different Users As stated in the last subsection, similar users performing similar tasks is not anomalous in many everyday situations, e.g., group activities with friends and coworkers. The experiment presented in that subsection showed that even when the users are performing the same scenario, there is a high variation of attribute values between runs (Appendix C.3).

As an added challenge to validity of the single-user conversion procedure, the other

users in the datasets are not only similar to the user being tested, but in some cases they are the exact same user. The other runs from a user technically represent the previous experience of that user. For the three selected datasets, the Locomotion dataset has four different users, the Helsinki dataset has two users (although they are not labeled and indiscernible), and the Nokia dataset has only one user. Only the Locomotion dataset has multiple, discernible users, and the tests with this dataset have removed the runs of the same user from the evaluation. Therefore, this threat is eliminated from the Locomotion dataset.

The danger of treating runs from the previous experience of the same user as runs from different users is that the same users would potentially have similarities between runs that would be more pronounced than if they were different users performing the same scenario. For example, the user in the Nokia dataset might always choose the same path to work each day, causing attribute similarity via user-specific regularities in means of transport and repeated visits to certain locations. As a consequence, there is a danger the results from these datasets would potentially not apply to real-life situations with different users doing the same task because they show that a single user does a similar thing in a similar way rather than showing that different users do similar things in similar ways.

To show that runs from the same user can be treated as runs from different users, these similarities between runs of the same user should be minimized. Unfortunately, it is hard to separate out effects of the same users from the effects of the same scenarios. Two properties of the selected datasets help to decouple the attributes describing runs from the users performing the runs. Firstly, as mentioned earlier, there are no static attributes that can be used to identify the same users or artificially increase attribute similarity in any of the three datasets. Secondly, the scenarios themselves for the Helsinki and Nokia datasets are not particularly specialized for each user. These two datasets are mostly made up of shared environment attributes that have more to do with where the users are than who they are. The Nokia dataset has only two non-shared environment attributes: an imprecise accelerometer reading and device orientation. The Helsinki dataset has device attributes and attributes describing an imprecise mode of travel as the only non-shared environment attributes. On the other hand, the Locomotion dataset is tightly bound to a particular user as it measures how

a user walks, i.e., the attributes are gyroscopes, accelerometers, and pressure sensors. In fact, the Locomotion dataset's only shared environment attribute is air pressure. Appendix C.5 runs the tests of this thesis without these attributes to get a better idea of how much these non-shared attributes bias the set of other users. The results of this experiment are discussed in Section 5.7.

In addition, the Environment Assumptions section (Section 3.1) contains an aspect of the experiment design for the evaluations of this thesis that helps to ameliorate the threats of users performing the same scenarios and treating the same users as different users: no previous experience of other users is assumed so only other users' current information is considered. This means that for any test instance from a run, only other users' instances from the same timestamp are considered. Any regularity in attribute values around that timestamp snapshot but not at that timestamp snapshot will not be considered. The implications of only using a single snapshot for other users are that even identical runs with the same user need to be aligned exactly in time to be identical. However, the runs of even the same users performing the same scenarios are not identical for the selected datasets. As seen by the results of the average difference experiments in Appendix C.3, even when the runs all start at the same timestamp, there is significant variation between the attributes. When the start time randomization of the single-user conversion procedure is added, this should result in even more variation.

Another implication of the single snapshot nature of the evaluations is that the set of other users at each timestamp are anonymous users that are doing similar tasks to the test user at that moment. There is nothing to identify a user from one timestamp as a user from another timestamp. To put it another way, the probability of another user doing the exact same task as the current user over a long period of time is unlikely, but having other users doing the same task at any specific time is potentially very probable in many tasks. For example, at any particular time a current user from the Nokia dataset would find it common during a walk to work that there were other users on the street experiencing the same shared environment context information that the current user is experiencing. The collection of other users around at any particular time does not have to correspond to the set of users at any other time. Figure 3.2 shows an example of this, as the current user (the square) is walking across a busy

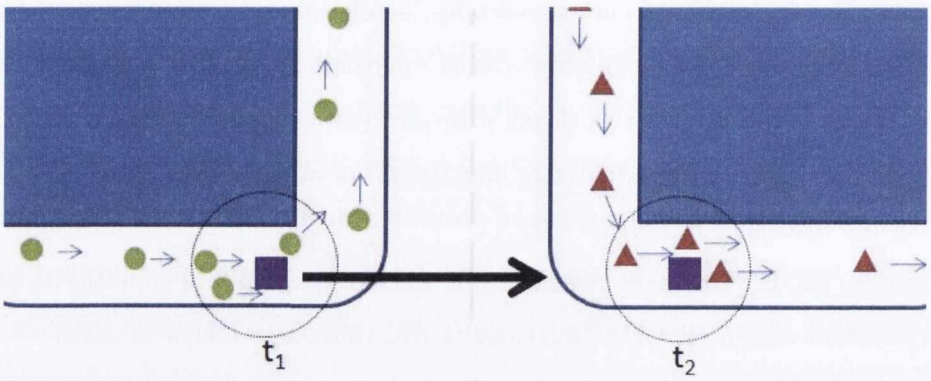

Figure 3.2: A user (square) walking across a busy intersection.

intersection. At time t_1 the current user is in range of three users (the circles), but at time t_2 the user is in range of three other users (the triangles). This is true of all the scenarios in the datasets, which record a user while doing common tasks in public areas such as walking and sitting.

The tasks and attributes that make up the selected datasets and the single snapshot nature of the evaluations decrease the bias of the set of other users due to treating runs from the same user as runs from different users. However, there is still some potential that a bias exists as the similarity due to the same user is difficult to isolate. In an effort to further reduce artificial similarity due to the presence of the same users and the same scenarios, an additional experiment is carried out in order to remove runs that are the most similar to the current user's run. For the Nokia and Helsinki datasets, this is done by clustering the runs per-scenario and removing the runs within-cluster from the set of other users during the evaluation of that run. For the Helsinki dataset in particular, it is hypothesized that if there is a same user bias in the dataset, the clustering algorithm will cluster runs of the same user together. The results are then compared to the evaluations with the full collection of runs to see if there is any significant difference in accuracy due to these similarities. The experiment is more fully described in Appendix C.6, and the results' implications on the validity of the conversion procedure are discussed in the post-evaluation Dataset Validity Analysis section (Section 5.7).

3.3 Machine Learning Methods

As discussed in Section 1.1.2, machine learning methods allow the performance of a computer program to automatically improve with experience [87]. Machine learning methods have both learning algorithms and models that those algorithms assume [7]. For example, a decision tree learning method uses learning algorithms such as entropy-based attribute splitting and pruning to train its model, i.e., a decision tree. Generally, a model ($g(x|\theta)$) returns an output value given an input (x) and parameters (θ). A learning algorithm assigns values to the parameters by induction on the training instances. When the parameters are assigned, the trained model is a reasoning method that estimates the function underlying the data [7].

This section details the "choose learning method" step of the learning method evaluation process. Selection of a learning method is a domain- and problem-specific process. Even a method that uses random guessing will not always be inferior to any given method [37]. This points to the importance of selecting the learning method that is most appropriate for the problem at hand.

The six learning methods surveyed in this section are among the most common in supervised learning. Their pervasiveness in almost every work in the previous chapter reinforces both their popularity and their appropriateness for the study evaluations. Additionally, all but neural networks were named among the top ten algorithms in data mining in a 2008 paper by Wu et al. [146].

This set of learning methods is a sufficient collection of diverse methods for the purposes of this section as completeness is not a goal. Instead, the goal is to select a reasonable number of learning methods with their associated learning algorithms and models that are appropriate for use in the evaluations of Chapter 5, where "appropriate" means those that meet the requirements introduced in the next subsection. Learning methods that are not able to meet these requirements at all are not presented. For example, hidden Markov models are not covered as they require sequential instances and this assumption is not valid for other users' information in this thesis' evaluations (see Section 3.1).

The next subsection presents the learning method selection requirements (Section 3.3.1) which reflect the research question criteria as well as other choices made in

the previous two sections. It then surveys the six learning methods identified in the previous chapter (Section 3.3.2 to Section 3.3.7) and finally selects the learning methods that best meet the selection requirements (Section 3.3.8).

3.3.1 Learning Method Selection Requirements

The requirements in this subsection are an accumulation of the research question criteria (Chapter 2), the properties of the environment that this thesis assumes (Section 3.1), and the properties of the selected datasets (Section 3.2.3) on which the experiments will be run in Chapter 5. The dataset properties must be considered in the selection of a learning method for two reasons. The first is a practical concern for the experiments to ensure that the learning methods are able to function with the selected datasets. The second is to maximize external validity, as the diverse properties of the selected datasets are representative of the diversity encountered in pervasive environments in general.

The requirements' definitions and their source are as follows:

Supervised learning Supervised learning methods are selected as the acquisition problems require an output attribute. The datasets selected in Section 3.2 also have values for each of the attributes, i.e., the training instances are labeled. Any learning other than supervised learning, i.e., with unsupervised or reinforcement learning, would also mean a failure to use the information provided by the dataset to its full potential.

Single snapshot This thesis assumes heterogeneous devices with no assumption of learning methods or previous experience (Section 3.1). This means that no learning method or structure of previous experience is required for other users' devices. In addition, no temporal structure is required for the previous experience of the application's user beyond having a collection of past instances from which to learn, i.e., chronological instances are not required. As such, the learning method must be capable of learning from instances with no fixed temporal relation and not require multiple, chronological instances over time. This also allows evaluations to be run that uniformly test different approaches to instance set selection,

Requirement	Definition
Supervised learning	The learning method must be able to take in labeled training instances.
Single snapshot	The learning method must be able to function with a training set that is not necessarily chronologically ordered over time.
Multivariate input	The learning method must handle multiple attributes as input.
Multi-type input and output	The learning method must manage both categorical and numerical values as input and output.
Knowledge autonomous	The learning method must not require additional feedback for training the model beyond the information available in the training data.
Sparse instance sets	The learning method must offer good performance if training with a small number of instances.
Incremental learning	The learning method must be able to easily update its model to include new or remove obsolete training data.

Table 3.6: Learning method requirements and their definitions.

i.e., any of the three different training instance sets can be used to train the same learning model.

Multivariate input In order to handle the selected multi-attribute datasets and to determine the full extent of a value's situational relevance (see Section 1.2.1), the learning methods should be able to function with multivariate inputs.

Multi-type input and output The selected datasets have both categorical and numerical attributes, so the learning methods must be able to handle both types as input. Additionally, the learning methods must be able to perform regression and classification in order to estimate the value of each context type.

Knowledge autonomous The learning methods must not require feedback or much additional information beyond a labeled instance set. Knowledge autonomy is important for an application to function in pervasive environments (see Section 1.2).

Sparse training sets A learning method for pervasive applications must function as well as possible when presented with sparse instance sets (see Section 1.2.1).

Incremental learning In order to handle the situational concept drift (see Section 1.1.2) and assign more importance to recent training data, the learning method should ideally be able to easily include newer data instances and remove ones that are no longer relevant. Lazy learners are more capable of incremental learning than eager learners, as the former reason about the instances directly rather than learning a function from the instances from which to reason.

It is important to note that only a subset of research question criteria are represented in the above requirements. These are requirements for the manner in which the learning methods are implemented and how they derive an output from a given instance set. These concerns are orthogonal to selecting the input instance set and the manner in which the learning methods are used in the experiments. Consequently, the instance set selection and evaluation validity research question criteria are not fully represented in the requirements of this section. The external validity of the datasets used in the evaluation is addressed in Section 3.2 of this chapter and is addressed

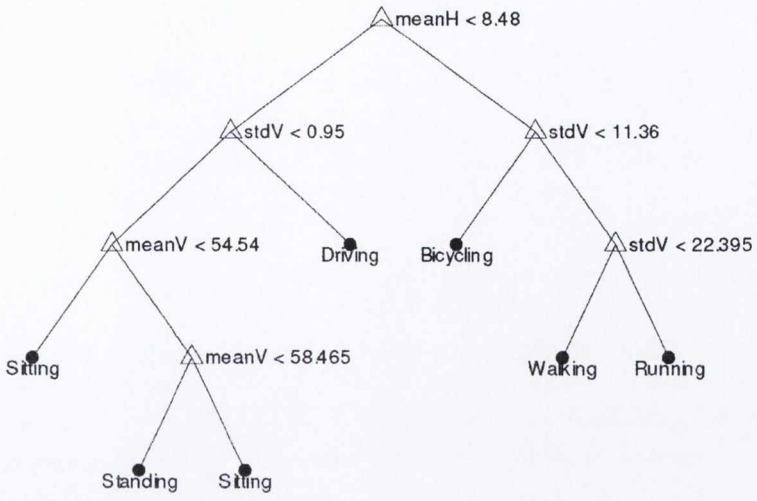

Figure 3.3: Decision tree classifier for a pervasive application from [148].

for the experiment design in the next chapter. The instance set selection criterion is evaluated directly in the experiments addressing RQ-3 to RQ-6 in Chapter 5.

The next six subsections detail the learning methods. For each subsection, the learning methods' associated models and learning algorithms are briefly summarized. Examples of pervasive applications that use the learning method are then listed, and the degree to which the learning method meets the requirements introduced in this subsection is assessed.

3.3.2 Decision Trees

Decision trees [7] are hierarchical tree structures that use the values of the attributes of the input problem to determine the path in the structure that leads to the problem's corresponding output. A simple example of a decision tree classifier [148] with numerical attributes from accelerometers is shown in Figure 3.3. A decision tree is a tree structure composed of internal decision nodes and terminal leaves [7]. The internal decision nodes contain a number of branches and depending on the output of the node's test function given the problem's attribute values, one of the branches is taken to either another internal node or to a terminal leaf. The terminal leaves contain the output for either a classification or regression task. In the decision tree classifier of Figure 3.3, the test functions are inequalities with numerical attributes and the terminal nodes are activity classes.

Decision tree learning describes how the structure of a tree is generated from the data contained in a labeled training set. There are many ways to structure a decision tree, many ways to assign test functions, and many different test functions from which to choose. If there is even one continuous attribute, there are an infinite number of decision trees that can be generated.

For classification trees, the function at each internal node is selected that best minimizes an impurity measure. An impurity measure is minimized if, when the training data is run on the tree, a node splits the data so that each branch contains only instances labeled with the same class type, i.e., the split is pure. If the split is pure, then the branches lead to a terminal leaf node with the label of the class assigned to it. A pure split occurs in the rightmost test function (triangle) in Figure 3.3, i.e., the tree is split into two separate classes, i.e., "Walking" and "Running". If the split is not pure, then the training data should be split such that the impurity is minimized. An impure split occurs in the leftmost test function (triangle) in Figure 3.3 and the activity class of "Sitting" can travel either branch. The right branch must be split further to achieve purity. Some examples of measures of impurity are entropy, Gini index, and misclassification error. Research has shown there is no significant difference between these three measures [7].

The construction of regression trees is almost the same as for classification trees, except that the impurity measure for classification is replaced by one appropriate for regression. This is based on the error of the regression output, i.e., the difference between the value of the terminal leaf node and the output value of the training instance. If the error is within a certain bound for all training instances, the learning algorithm is complete. If the error is not acceptable, more nodes are added that split the instances until the error is acceptable.

The challenge in decision tree learning is to generate a decision tree with acceptable performance that is not overfit to the training data. Overfitting is a problem as, unless there are identical problems with different labels, a decision tree can be generated that can replicate the outputs for all the training data exactly. To counteract overfitting, a small tree is desirable as it makes the tree more general [7]. Pre- and post-pruning of the tree accomplishes this goal, although this requires setting part of the data (i.e., the pruning set) in the training set aside to assist in judging optimal pruning.

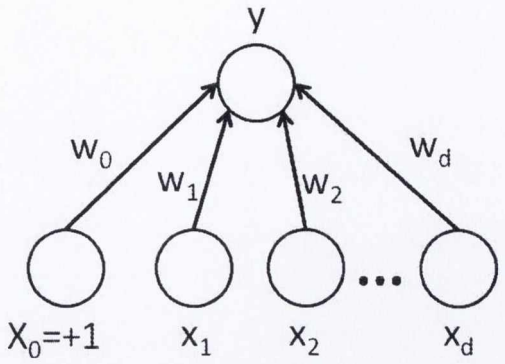

Figure 3.4: A simple perceptron derived from [7].

Appropriateness for Study Evaluation

Decision trees are used for many pervasive applications, e.g., [147, 68, 150, 19]. Decision tree learning cannot handle a few of the requirements laid out in Section 3.3.1, in particular, incremental learning and sparse instance sets. If a newly encountered training instance is added that is misclassified by the tree, the tree must be modified and in the worst case be regenerated. Additionally, battling overfitting through pruning requires a pruning set, which is difficult to allocate from the training set without affecting performance with sparse instance sets. Decision trees do have the additional benefit of functioning in the presence of missing attribute values [13] and can be converted into If-Then rules which are an easy concept for application users to understand [7].

3.3.3 Artificial Neural Networks

Like decision trees, artificial neural networks [7] are hierarchical structures that are able to perform classification or regression. They are based on a model of the human brain where simple individual neurons are connected to perform highly complex processing. The unit of an artificial neural network is called a perceptron (Figure 3.4) and it is made up of inputs (x_j) and outputs (y) connected by arcs that each have a weight (w_j) attached.

The node at the receiving end of the weighted arcs combines the inputs with a basis function, also incorporating the arcs weights. For regression, the simplest example of

a basis function is one describing a hyperplane [7]:

$$y = \sum_{j=1}^{d} w_j x_j + w_0 \tag{3.1}$$

The basis functions can also be much more complex. In the case of a classification task, it is often a sigmoid threshold function whose output represents two different classes depending on the side of the threshold on which an output falls. A perceptron has a bias unit that always has the value +1 (x_0 in Figure 3.4) which serves as the intercept when a basis function is linear and a threshold when the basis function is a sigmoid. Additionally, although only a single output is shown in Figures 3.4 and 3.5, a neural network can have multiple outputs with their own weighted arcs from a subset of the inputs. If the Outputs layer and its incoming arcs were removed from Figure 3.5 and the Hidden Units layer was renamed Outputs, this would be an example of a neural network with multiple outputs. Specifically, there would be three parallel perceptrons.

The weights are assigned by learning algorithms like linear regression through batch learning, which means that all the training instances are used at once to train a structure. This is the approach taken by the decision tree learning discussed in the last subsection and the Bayesian learning approach discussed in the next subsection. Alternatively, perceptrons can be trained through online learning, starting with random weights. The weights are then updated as each single training instance is presented to the neural network, where the error is based on individual training instance error rather than the error function over the entire sample as in batch learning. Online learning is appealing in dynamic domains where the problems change over time. In neural networks, this allows the weights to gradually converge to ones offering the best performance with a bias towards the latest training instances.

Neural networks that have a single layer of weights are only able to approximate linear functions. To approximate nonlinear functions (for regression) and nonlinear discriminants (for classification), one or more hidden layers are added to the network (Figure 3.5). The hidden layers are made up of connections from the inputs or other hidden layers where each hidden unit has a basis function (e.g., the simple function in Equation 3.1) and the incoming arcs are assigned their own weights. Each hidden unit is essentially its own perceptron. The hidden layers have nonlinear basis functions,

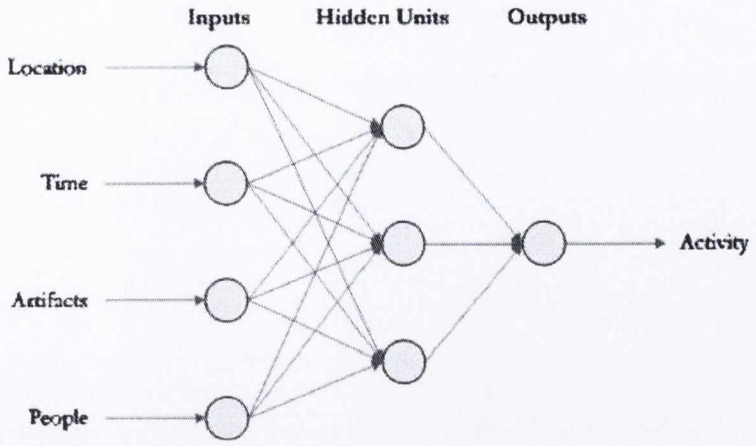

Figure 3.5: A neural network with hidden nodes from [40].

such as the sigmoid function, in order for the network to approximate a nonlinear function or discriminant.

Training a multilayer neural network is accomplished through the backpropagation algorithm. The error propagates back through the network from the output based on gradient descent. Gradient descent is local, so the weight updates only depend on the inputs and output of each perceptron along with the error from the subsequent perceptrons. The training process is run several times (with each training iteration called an *epoch*) for the inputs so the weights converge to those with the best performance. The number of epochs is determined via performance testing with a validation set, as overtraining hurts the general performance by overfitting the model to the training data.

Tuning the appropriate number of hidden layers, number of perceptrons within those layers, and assigning connections between those perceptrons can also be learned. This is done constructively by adding connections or nodes or destructively by removing them [7]. A validation set is also used to maximize general performance for structure.

Appropriateness for Study Evaluation

Several pervasive applications use artificial neural networks to aid their approaches, e.g., [40, 137]. Artificial neural networks address most of the requirements of Section 3.3.1. Sparse environments are not handled well both in the training of the network weights through backpropagation and for learning the network structure which requires a separate validation set. In addition, the incremental evolution of online learning helps deal with concept drift. However, there are still remnants of the obsolete training

instances in the network and identifying and removing effects of specific instances that are no longer valid is not possible without retraining. So while there is incremental learning, it does not offer the same control as the Bayesian learning methods and the instance-based learning methods of the following two subsections. This problem is associated with another major criticism of neural networks: they are black boxes. The If-Then rules derived from decision trees or the probabilities of Bayesian learning methods are easily understandable by users, but the weights and hidden nodes do not translate well to explanations for users.

3.3.4 Bayesian Learning Methods

This subsection discusses three approaches that use Bayesian learning: general Bayesian classifiers, Bayesian belief networks, and naive Bayes classifiers.

General Bayesian Classifiers Bayesian decision theory [37] uses probability to determine the most likely output value. Mostly used in classification, it can also be used in regression by discretizing a numerical output into intervals and selecting the value that represents the most probable interval, e.g., the mean of the most probable interval. The same discretization is also common in handling numerical attributes.

Bayesian decision theory uses Bayes' rule:

$$P(c_i|a_1, a_2, \ldots, a_n) = \frac{p(a_1, a_2, \ldots, a_n|c_i) \times P(c_i)}{p(a_1, a_2, \ldots, a_n)} \tag{3.2}$$

Where c_i is an output class and each a_j is an attribute type with an assigned value. The equation can also be written informally as [37]:

$$posterior = \frac{likelihood \times prior}{evidence} \tag{3.3}$$

For use in multi-attribute classification, Bayesian decision theory predicts the probability of the class given an assignment of attributes (posterior). It does this by using information that is available from labeled training data: the likelihood of an assignment of attributes given a class (likelihood), the probability that a class is seen regardless of the attributes (prior), and the probability that the attributes are assigned as such without knowing the class (evidence). To find the output class, the class with the maximum posterior probability is selected.

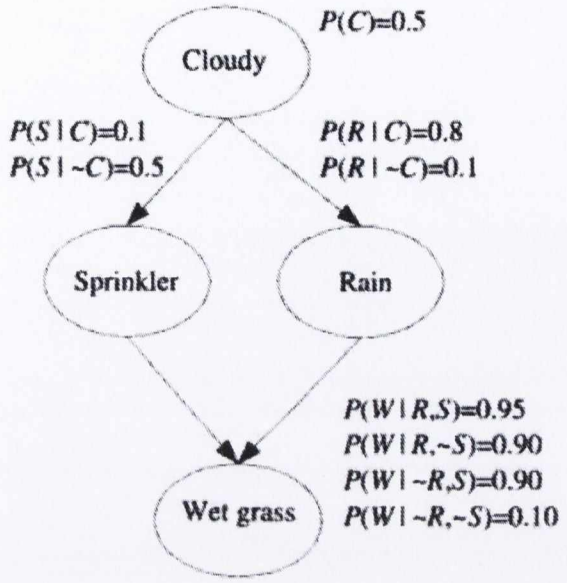

Figure 3.6: A Bayesian belief network from [7].

Since the evidence is a constant in the search for the maximum posterior probability, it is ignored. This leaves only likelihood and prior probabilities to learn. General Bayesian learning uses each training instance to increase the count of the appropriate assignment and prior. For example, for each labeled training instance, the class's prior count is increased by 1 and is divided by the total number of training instances seen at the end of the learning process. This reflects the probability that a class was seen at all, i.e., the prior probability. A similar counting takes place for the likelihood, where each instance for a class and its associated attribute assignment is increased by one. Unfortunately, this requires every attribute assignment and class combination in the problem space to be seen several times in order to get the appropriate relative probabilities [87]. Some assumptions that alleviate this limitation are seen in the Bayesian belief networks and the naive Bayes classifiers discussed below.

Bayesian Belief Networks Like general Bayesian classification, Bayesian belief networks predict the class likelihood given a set of assigned attributes. Bayesian belief networks differ in that only certain variables are designated as having a direct influence on a specific variable. This designation is shown in graphical form through a Bayesian belief network (Figure 3.6). Bayesian belief networks are a probabilistic

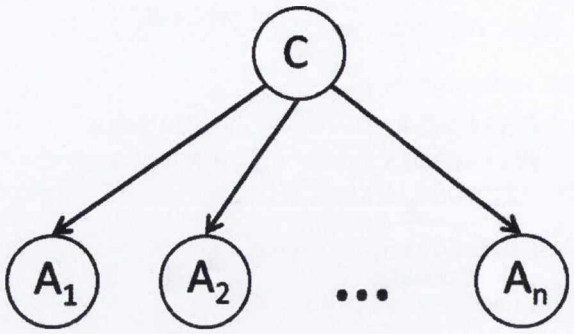

Figure 3.7: Naive Bayes classifier.

graphical model where connections between random variables (nodes) denote conditional dependence. The arrows connecting nodes denote the influence from parent to child node, e.g., from the attributes Sprinkler and Rain to the output Wet grass in Figure 3.6. The nodes are directly influenced only by their parents and the posterior probabilities (the conditional probabilities in Figure 3.6) for these connections must be specified before inference is able to be carried out with the network. Instead of Equation 3.2, the equation for the posterior probability for any random variable a_1 in a Bayesian belief network is now:

$$P(a_1|a_2, a_3, \ldots, a_n) \propto P(a_1, a_2, \ldots, a_n) = \prod_{j=1}^{n} P(a_j|parents(a_j)) \tag{3.4}$$

Where $a_2 \ldots a_n$ are ancestors of a_1. Bayesian belief networks require more knowledge than general Bayesian classifiers in order to determine which variables directly influence others (i.e., defining the structure of the Bayesian belief network), but the presence of the structure allows the learning of the posterior probabilities with far fewer training instances if the structure is a sparse network [87].

Naive Bayes Classifiers The idea of minimizing conditional dependence between random variables is taken to the furthest extreme by the assumptions of the naive Bayes classifier [87]. Naive Bayes classifiers assume conditional independence for all their random variables, giving the simple structure in Figure 3.7. This is used when the dependency relationships of the attributes are unknown and in practice performs well even if the attributes are not actually conditionally independent [37]. The simplified

equation is:

$$P(c_i|a_1, a_2, \ldots, a_n) \propto P(c_i) \times \prod_{j=1}^{n} p(a_j|c_i) \tag{3.5}$$

This further reduces the size of the training set necessary to learn the posterior probabilities from the reduction from general Bayesian classifiers possible with Bayesian belief networks [87]. Where general Bayesian classifiers would need to see training instances for the entire attribute space for each class multiple times, learning in naive Bayes classifiers increases the total count of a class for each attribute with just one training instance. It also does so without requiring the knowledge of conditional dependence required by Bayesian belief networks (although at the cost of assuming conditional independence).

Appropriateness for Study Evaluation

Bayesian decision theory is used in several pervasive applications, using both Bayesian belief networks [6] and naive Bayes classifiers [66, 91]. The general Bayesian classifier cannot function with sparse training sets, so it is not suitable for the evaluations of this thesis. Bayesian belief networks require less training data than the general Bayesian classifier but still necessitate the knowledge required to build the Bayesian belief network structure. The structure can be learned, but a validation set is required to do so [6]. The naive Bayesian classifier requires even less training data than Bayesian belief networks. Bayesian learning methods also function incrementally, as when a new training instance is added or removed, only the counts need to be updated (and no change to the structure is necessary). The conditional independence assumption for classifiers often performs well even when the assumption does not hold [37]. Using the naive Bayes classifier for regression leads to less ideal performance when the independence assumption does not hold, although it performs comparably to linear regression [42].

3.3.5 Instance-Based Learning

Instance-based learning [87] differs from the other learning methods presented in this section as it waits to construct the classifier or regression function until the problem is presented. The delay in constructing the approximating function is why instance-based learning methods are often called "lazy learners" and the other learning methods

described in this section are called "eager learners." The delay allows the learner to use the information describing the problem to customize the approximation function to that particular problem. This often leads to a local approximation function that performs well in the portion of the problem space containing the posed problem but does not have good overall performance across the entire problem space [87].

This thesis has already presented a description of an instance-based learning method in Sections 2.3 and 2.4 of the related work, i.e., Case-Based Reasoning (CBR) methods [33, 2]. Here, k-nearest neighbor learning is presented as an example of instance-based learning when the problem can be posed in an n-dimensional Euclidean space. This subsection will then delve further into the CBR techniques that allow the learning method to meet the requirements of Section 3.3.1.

3.3.5.1 k-Nearest Neighbor Learning

The k-nearest neighbor learning algorithm selects the k nearest training instances to the problem in order to solve it. Once the k nearest instances are identified, they are combined to give the answer to the problem. In the case of regression, this is normally an average of all the instances' output values. For classification, a majority function is used to select the appropriate class.

"Nearest" is defined by a distance function which combines the distances between each attribute of the problem space and the corresponding attribute for a training instance into a single number. The distance function is often Euclidean distance but can be implemented with several other functions, e.g., city-block distance, other Minkowski distances, and Mahalanobis distance [80].

There are several variations on the k-nearest neighbor algorithm. Distance-weighted nearest neighbor weights the contribution to the final output value of each of the k instances by their distance from the problem. For regression, this produces an output based on a weighted average where the nearer instances have more influence. For classification, an instance has a more influential vote based on being nearer to the problem. Another variation is locally weighted regression, where only the k nearest instances are used to learn a multivariate regression function. The problem is then input into the function obtained from the regression and the output of that function is returned.

A drawback for the above implementations of the k-nearest neighbor algorithm is its reliance on numerical attributes, violating the multi-type input requirement. It is difficult to generally define the distance between two categorical attributes. Another major drawback for these implementations is the lack of support for the relative importance of each attribute. For example, a small physical distance between two users is more important than a small distance in a user's age for predicting ambient temperature. One of the main research focuses in CBR is using relative importance of attributes to create more accurate reasoners [128, 28].

3.3.5.2 Case-Based Reasoning

As discussed in Section 2.3, CBR is a technique that reuses the solutions of previously solved problems (i.e., cases) that are similar to the current problem. To relate this terminology to k-nearest neighbor algorithms, "most similar" is analogous to "nearest." Rather than being restricted to representing instances in an n-dimensional Euclidian space like k-nearest neighbor algorithms, cases in CBR are able to contain rich symbolic attributes that have complex attribute and aggregate distance functions [87]. These distance functions are called similarity measures where attributes and instances are more similar when the distances are smaller.

CBR techniques employ a global similarity measure to determine the overall similarity of a training instance to the problem. A *global similarity measure* is a weighted sum of the outputs from local similarity measures for the set of attributes. A *local similarity measure* compares the similarity of a single attribute, e.g., comparing the similarity of two locations or two roles. There can be multiple and varied implementations of local similarity measures depending on the type of attribute and application domain, e.g., distance-based functions, such as Euclidean or city block distances, or via a similarity table for categorical attributes [76].

Both the local similarity measures and the weights of the global similarity measures must be specified in order for inference to be performed. Learning the local similarity measures can be carried out a number of ways. Local similarity measures are often equivalent to the distance functions described previously in the k-nearest neighbor learning subsection [28]. They are often normalized to make comparison between similarities of different attributes easier, e.g., each similarity must be in the range from

0 to 1. For categorical attributes, a local similarity measure can be an equivalence relationship (e.g., 1 if the attributes are in the same category, 0 if not) or can be based on the degree of similarity between two categories (e.g., similarity based on a function of the length of a path between concepts in an ontology [62]). Complex local similarity measures may also be learned via relative utility feedback searching the space using evolutionary algorithms [129].

The weights of global similarity measures can likewise be learned. Stahl's algorithm [128] performs an iterative search for the weights that best minimize the average error function for a training set using a conjugate gradient algorithm. This approach was found to perform quite well with sparse training sets [128]. Similarly, Cheng and Huellermeier [28] learn the weights of the global similarity measure by using active learning techniques that require the feedback of which training instances in a subset are more similar than the others in the subset.

Appropriateness for Study Evaluation

There are many pervasive applications that use instance-based learning techniques like CBR, some of which were detailed in Section 2.3. The CBR approach meets all of the requirements set out in Section 3.3.1. Incremental learning is accomplished by simply adding an appropriate new instance or removing an obsolete one. Finally, learning with sparse instances sets and knowledge autonomy are also facilitated by using weight learning algorithms like Stahl's algorithm described above.

3.3.6 Support Vector Machines

A support vector machine (SVM) is a linear classification and regression approach that uses a weighted subset of the instances (so called support vectors) to calculate its output [7]. Classification with SVM is presented in the first subsection (Section 3.3.6.1), progressing from binary classification to multiclass classification. Regression with SVM is then introduced in the following section (Section 3.3.6.2).

3.3.6.1 Classification with SVM

Binary classification learns a model that outputs whether the test instance is a member of one class or the other. SVM accomplishes this by defining a hyperplane discriminant

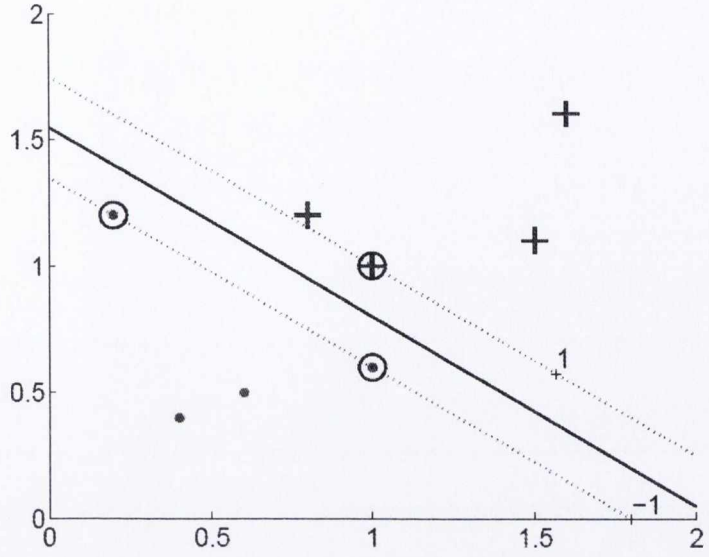

Figure 3.8: Binary classification with support vectors represented by the instances within circles and the margin around the solid line representing the hyperplane is indicated by dotted lines (from [7]).

to separate the two classes. A test instance is assigned to a class depending on which side of the hyperplane the instance falls. To increase the generality of classification, the hyperplane is defined such that it optimizes the margin between the closest instances to the hyperplane. Any instances on the margin of the hyperplane are the support vectors. Figure 3.8 shows the margin as dashed lines surrounding the solid line representing the hyperplane. The support vectors are the circled instances that lie on the margin.

If the instances are not linearly separable, the feature space can be converted into another higher-dimensional space using non-linear basis functions, e.g., polynomials, radial-basis, and sigmoidal functions. The optimization problem is written using Lagrange multipliers and maximizing those using quadratic optimization methods. The training instances with non-zero multipliers are the support vectors. This optimization problem does not require performing the computation to convert the feature space directly into the higher-dimensional one, but instead it replaces the inner product of the basis functions with a kernel function that allows the computation to be done in the original feature space.

If the instances are still not separable, a soft margin can be introduced. This draws the hyperplane such that any deviation from the margin (both for incorrectly classified instances and correctly classified instances that are within the margin) is minimized. The error is introduced into the optimization problem and the sum of the errors is assigned a penalty factor, C. For a soft margin approach, the support vectors are instances within the margin, on the margin, and on the wrong side of the discriminant.

For multiclass problems, the most common approach is to learn K different binary SVMs, where K is the number of classes. Each SVM takes one class and assumes that all the other classes make up one other class. During classification, the SVM with the largest output is chosen as the output class.

3.3.6.2 Regression with SVM

Regression with SVM is very similar to that with classification. Instead of a hyperplane acting as a discriminant, it indicates the value of the output. There is still a margin to the hypeplane, but instances within the margin do not count towards the error. That is, the error of all differences from the hyperplane value up to an accuracy parameter (ϵ) are considered zero. Consequently, the support vectors are any instances that are on the boundary of the margin or outside of it.

Appropriateness for Study Evaluation

There are many pervasive applications that use SVMs [151, 95]. SVMs are considered the best, off-the-shelf learners for their computational advantages and the fact that there is no optimization iteration necessary as there is a unique solution to the optimization problem [7]. Sparsity of training data is managed by maximizing the margin between what test instances are available, and incremental learning with SVM is also possible [24].

3.3.7 Ensemble Learning

Ensemble methods combine the outputs of several base learners in an attempt to get a more accurate overall output [36, 7]. The base learners used in an ensemble must be accurate and diverse in order for the ensemble to be more accurate than any of its

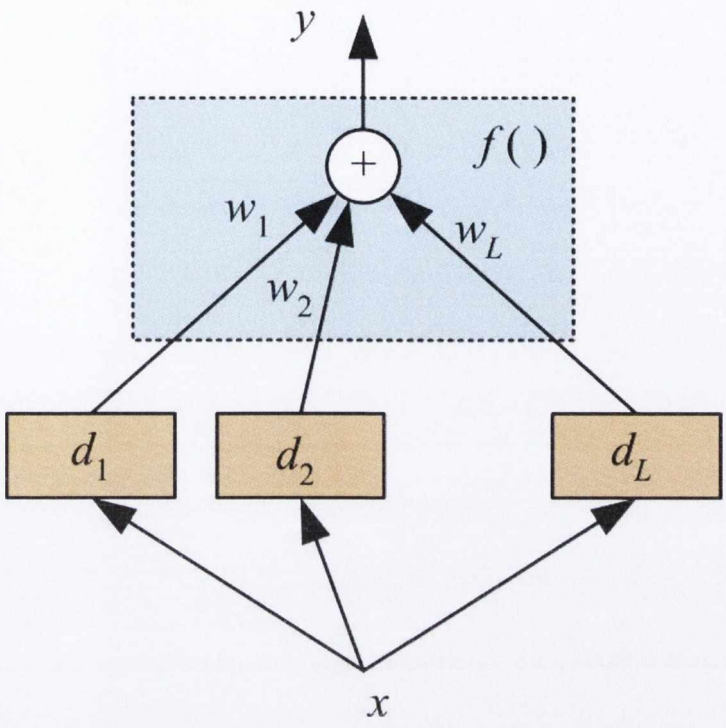

Figure 3.9: A voting scheme where a linear combination of outputs ($f(\,)$) from the learners (d_is) form the output (y) of the ensemble for the test instance (x) (from [7]).

individual members. If the base learners are not sufficiently accurate, their combination will increase the overall error rather than decrease it. Two base learners are diverse if they make different errors on new instances [36]. The next two subsections discuss how the outputs of the base learners are combined (Section 3.3.7.1) and how diverse learners are created (Section 3.3.7.2).

3.3.7.1 Output Combination

In most ensemble methods, the outputs of base learners are combined through weighted voting. For example, each output from a regression learner can be averaged together with equal weight or the output can be weighted by how well the learner performed on the training set or a validation set (see Figure 3.9). Similar voting schemes can be created for classifiers voting on the class of the instance.

Examples of other approaches to using the collection of base learners are to select a subset of the learners based on properties of the test instance, have the combiner ($f(\,)$ in Figure 3.9) be a (possibly non-linear) learner as well, or cascade increasingly more

complex learners until an answer with acceptable confidence is achieved.

3.3.7.2 Creating Diverse Learners

Creating diversity between learners of an ensemble method is a key to the success of an ensemble method. According to Brown et al. [15], there are three ways to create diversity. The first method is to influence the starting point in the hypothesis space by varying the initialization parameters, e.g., the weights of a neural network. This is the least effective way of achieving diversity. The second method is to influence the set of accessible hypotheses. This can be achieved by manipulation of the training set or by manipulation of architectures. Manipulation of the training set means to create different subsets of either the instances, features, or both. Manipulation of architectures can be accomplished by using learners with variations of the same type, e.g., neural networks with different numbers of hidden nodes, or of completely different types of learners, e.g., using both decision trees and neural networks. The third method of creating diversity is to affect how the hypothesis space is traversed. This is generally performed by penalizing a branch through the hypothesis search space that would lead to similar behavior to a learner already in the ensemble.

Each of these three methods for creating diversity can be implicit or explicit. Implicit methods generally mean that the manipulation is done at random, whereas explicit methods directly manipulate the learner and dataset to ensure a level of diversity. Here we will focus on representative cases that manipulate the training set: bagging (implicit) and boosting (explicit). Dataset manipulating approaches are covered as Duin and Tax [38] found that classifiers with different datasets are more useful for accurate predicting than different classifiers using the same dataset.

The word *bagging* comes from the combination of the words bootstrap and aggregating [12]. Bagging is a voting ensemble technique that creates diverse learners by randomly sampling with replacement a single dataset and training each learner on a different sample.

The original version of boosting trained only on instances which were misclassified by the previous learner. This allowed diversity between the learners but required a very large dataset. AdaBoost [43] is a specific implementation of boosting that functions with sparse datasets. AdaBoost.M1, a variation of AdaBoost, is a sampling method like

bagging except that AdaBoost.M1 is an explicit method that is more likely to sample instances that were incorrectly classified previously. Instead of training subsequent learners only with the subset of instances that the previous learner incorrectly classified, the misclassified instances are given a higher weight and the correctly classified instances are given a lower weight. The classifiers are then combined through weighted voting, where the weights are a function of how well they perform on the training set, i.e., greater weights are assigned to classifiers with lower error. Approaches for regression perform similarly where the "misclassifications" are for values that fall outside of an error margin.

Appropriateness for Study Evaluation

There are infinitely many types of ensemble methods, and any of the learning methods covered in this section in any combination could be used. Ensemble learners are used in several pervasive computing applications [112, 79]. There also exist bagging and boosting approaches that perform incremental learning [99]. For example, online bagging distributes any new instance among the different base learners according to a Poisson distribution and then relies on an online version of the base learners to update themselves on the new instance.

The variation of AdaBoost is appropriate for this study as it meets most of the requirements of the evaluation, including dealing with sparse datasets through resampling. A version of AdaBoost whose base learner is a decision tree algorithm is described in the next section. While the multi-type input and output capability is not ideal, a discretization can be performed (similar to Bayesian methods) that will allow the method to perform with numeric inputs.

3.3.8 Selected Learning Methods

Table 3.7 provides a summary of how well each of the six learning methods meet the requirements. Based on the number of requirements met or partially met, the learning methods selected are the naive Bayes classifier from Bayesian learning, a CBR approach from instance-based learning, an SVM approach, and a boosting approach from ensemble methods. The selection also allows the evaluation of both an eager and lazy learning methods. Decision tree learning and artificial neural networks have limited

Learning Method	Supervised Learning	Single Snapshot	Multivariate Input	Multi-Type Input and Output	Knowledge Autonomous	Sparse Instance Sets	Incremental Learning
Decision Tree Learning	X	X	X	O	X	-	-
Artificial Neural Networks	X	X	X	X	X	-	O
Bayesian Learning	X	X	X	O	X	X	X
Instance-Based Learning	X	X	X	X	X	X	X
Support Vector Machines	X	X	X	X	X	X	X
Ensemble Learning	X	X	X	O	X	X	X

X = supported, O = limited support, - = no support

Table 3.7: Summary of learning method applicability.

or no support for sparse instance sets and incremental learning. Bayesian learning, ensemble learning, and decision tree learning also have limited support for multi-type outputs, as they both require discretization for regression. The version of the ensemble method (boosting) uses a decision tree as its base learner algorithm. While decision trees are not used by themselves, the boosting technique still allows them to be included in the experiments. Finally, Bayesian learning is not ideal for regression, but overall it still performs with results similar to linear regression [42].

3.4 Learning Method Implementations

As discussed in the introduction to Section 3.3, the following implementations of the learning methods are used for the comparative evaluations of this thesis. It does not matter whether the learning algorithms are completely fine-tuned to the task or whether there are other algorithms that perform a little better. The goal of the evaluations is to study the comparative value of using these learning methods to select other users' information, not to maximize their individual performance. It is only important that they best meet the learning method requirements laid out in this section. The emphasis away from fine-tuning the learning methods is analogous to the approach of Leake and Sooriamurthi in the comparative evaluations for the MCBR framework [72] described in Section 2.4.1.

In order to answer the second research question (RQ-2) concerning whether the four selected learning methods are more accurate than existing methods for context acquisition in pervasive environments (e.g., [57, 27, 111]), this section also presents an implementation of a context selection method. The context selection method is considered a learning method as well, although it is a very simple one.

This section is devoted to the implementation of the five selected learning methods. First, the parameters needed by the model and learning algorithm are listed along with any dataset-specific considerations. Next, the algorithms to learn the parameters that are not predetermined are described. Finally, the manner in which inference is performed on the learned model is presented.

3.4.1 Equal Width Discretization Naive Bayes

The Equal Width Discretization (EWD) approach [23] was selected as the naive Bayes classifier for the evaluations as it is simple and offers reasonably good performance [54]. However, as noted in [148], EWD can lead to suboptimal performance when the number of discretized bins is not selected with respect to the size of the training set. Customizing the learning method to the training set size is important in pervasive environments as the number of instances can vary greatly, e.g., the different number of other users in full and sparse environments. The implementation of EWD in this thesis uses Yang's proportional discretization [149] in order to dynamically determine the number of intervals according to the size of a given training set.

Parameters

There are a number of parameters that need to be estimated for the model. All the parameters described below are used to deal with the numerical attributes, both input and output:

- *Number of bins*: This parameter is set to be the square root of the size of the training set [148]. This allows the algorithm to dynamically configure the number of intervals according to the size of a given training set.

- *The value of each bin*: This parameter is applicable only for output variables. Once an output bin is selected as the most probable, this value is returned from the learning method as the estimated value. For the evaluations of this thesis, the value of a selected bin is equal to the median of all the values placed in that bin during training.

- *Minimum and maximum values for an attribute*: These parameters define the full range to be divided into bins. For the evaluations of this thesis, the maximum and minimum values were assigned for the maximum range present for that attribute over the entire dataset.

A note for the last parameter: The predefinition of estimates for the minimum and maximum values of an attribute requires minimum knowledge, as many attribute types have a natural range, e.g., ambient temperature or ambient pressure. If an estimate

cannot be easily acquired, the range can be learned from the training data. In either case, if a new problem is outside the range, it can be assigned to the closest bin. If several new instances far outside the original range for an attribute are encountered, the bins should be restructured and the learning algorithm run again.

Learning algorithms

Equal Width Discretization (EWD) [149] divides the range of data into k equally sized bins, and then treats those bins equivalently to classes for categorical attributes if a value from a training instance falls within the bin's range. k is determined dynamically as described in the "Number of bins" explanation in the parameter list. Once the bins are defined for each attribute variable, a pass through the training set is done to update the counts of the class and bin prior probabilities ($P(c_i)$) and the counts for the corresponding likelihood ($P(a_j|c_i)$) needed for the inference in Equation 3.5

A further note about how the model is trained for the evaluations of this thesis: Even though it is an eager learning algorithm, a new model is trained each time for tests involving other users. This is for evaluation purposes only and if this approach were deployed in an actual pervasive environment the model would be updated with the new training instances instead of retrained.

Reasoning method inference

When a new problem instance is presented, inference with the trained Bayesian network uses equation 3.5 to find the most probable value. The equation is run for each class or bin in the output, and the class or bin with the largest result for that equation is the most probable. The class or the value representing the most probable bin is then returned as the output of the classification or regression function. If there are multiple bins with equal probability, one bin is selected at random.

3.4.2 CBR Approach

The CBR approach uses a global similarity measure to identify the most similar instances. The global similarity measure used is a weighted sum of each attribute's local similarity measure. The outputs of the global similarity measure and the local similarity measures are in the range $[0, 1]$.

Parameters

Local similarity measures: All local similarity measures are in the range $[0, 1]$, where 0 indicates no similarity between two values of the same attribute and 1 indicates the two values are identical.

- *Categorical attributes*: The local similarity measure for categorical attributes is equality. If the value of an instance's categorical attribute is equal to the value of another instance's categorical attribute, the similarity is 1. Otherwise it is 0.

- *Numerical attributes*: The local similarity measure for numerical attributes is normalized linear distance. That is, the similarity of two values v_1 and v_2 of an attribute is $sim(v_1, v_2) = max(0, 1 - \frac{|v_1 - v_2|}{max(v_i - v_j)})$ where the denominator represents the maximum range for that attribute. This ensures that the similarity for numerical attributes is always in the range $[0, 1]$.

  - *Maximum range*: Like the minimum and maximum values for an attribute in the naive Bayes implementation, the range can be estimated or learned from the training instances. If the attribute values of two instances are further away than this interval, the similarity is still 0. For the evaluations of this thesis, the interval was assigned for the maximum range present for that attribute over the entire dataset.

Weight learning algorithm: The weight learning algorithm used for the evaluations is from Stahl's case-order algorithm [128] introduced in Section 3.3.5 and described more fully below. It is an iterative method that attempts to find the weights that minimize an instance ordering error function. The instances are ordered according to their similarity to the problem. The error function calculates the difference between the ordering of the instance-problem similarities derived from the global similarity measure using the current iteration's weights and the actual given ordering.

- *Maximum number of training instances*: This parameter is the maximum number of training instances used in Stahl's algorithm. The minimum number of history instances for a dataset is the Helsinki dataset with 21 (see Section 4.1). In the interest of allowing all datasets to have the same number of weight training instances, this parameter is set to 20.

- *Initial and default weights*: The set of initial weights passed into the weight learning algorithm is the uniform weight vector, i.e., $w_i = \frac{1}{n}$ where n is the number of predictor attributes. This equal weighting method is also used when the weights cannot be learned from the training instances, i.e., when all the output values are identical and there is no way to order the cases.

- *Degree of disorder (α)*: This parameter controls how much disorder in the cases impacts the new weights. For the purpose of the evaluations in this thesis, the value of α is set to 1 which is the same as that of Stahl's experiments [128].

- *Maximum number of iterations*: This is the number of iterations of the conjugate gradient method. The algorithm in [128] only uses this parameter if the stopping condition is not reached by the time the maximum number of iterations is reached. The specific maximum number of iterations is 100.

- *Learning rate* (λ): This parameter controls the rate at which the algorithm converges to the minimum error. As with the implementation in Stahl's evaluations [128], the initial learning rate is set to $\lambda = \frac{\Delta_{max} w}{max\{\Delta w_i\}}$ where the numerator is the maximal change in weights parameter (see sub-bullet) and the denominator is the initial maximum partial derivation of the error function with respect to the ith attribute's weight w_i, i.e., $\Delta w_i = \frac{\partial E}{\partial w_i}$.

  - *Maximal change in weights* ($\Delta_{max} w$): This parameter sets a limit on the maximum change from the initial weights that the algorithm will assign. For the purpose of the evaluations in this thesis, the parameter is set to 0.1 which is the same as that of Stahl's own evaluations [128].

- *Stopping condition*: This parameter determines if the learning algorithm should stop before the maximum number of iterations has been reached. For the purposes of the evaluations in this thesis, the stopping condition is a minimal change in the error function, which is the stopping condition used in Stahl's experiments [128]. The specific stopping condition is a minimum change in the error function of .0001.

Learning algorithms

Stahl's weight learning algorithm performs an iterative search for the weights that best minimize their average error function using a conjugate gradient algorithm [128]. The error function is based on the difference between the ideal ordering for a set of queries on a case base and the ordering that the weights given by the current iteration return. The ordering is based on similarity, where the most similar case is first, the second most similar case is second, and so on. The ideal ordering is given by a "teacher" that is able to order cases according to their utility in solving each query. The "teacher" is deliberately vague as it is assumed to be application-specific which means it can be anything from a known utility function to feedback from a human domain expert.

To eliminate the necessity for user feedback, the learning methods for the evaluations of this thesis use the fact that the output attribute also has a local similarity measure and that the value of the output value for the training instance is known. This similarity measure is used as an assessment of how well the learned model is performing, i.e., it is the "teacher". To automatically perform the weight learning algorithm, a leave-one-out approach is used where the removed instance from the training set is the query and the remaining instances are judged based on their usefulness in handling that query.

Reasoning Method Inference

After the weights have been learned from a subset of the training set using Stahl's algorithm, inference involves selecting the instance from the training set that is most similar to the problem and returning that instance's output value as the solution. A global similarity measure between the problem instance and a training instance is calculated for each instance in the training set. The value from the training instance with the largest output from the global similarity measure is the solution. If there are multiple instances with identical similarity values, one instance is selected at random.

3.4.3 ν-SVM

ν-SVM implementation [120] was selected for the SVM learning method as it performs both classification and regression. The ν-SVM learner adds a parameter (ν) that allows

control of the number of support vectors. Specifically, ν is a lower bound on the fraction of support vectors among the training set and an upper bound on the number of errors. This is considered more intuitive than the other parameters and automatically minimizes the accuracy parameter (ϵ) in regression and removes the penalty factor (C) for classification. The specific implementation uses LibSVM (Version 3.1) [25] and a LibSVM wrapper (Version 1.0.0) from the Weka machine learning toolkit (Version 3.7.3).

Parameters

There are a number of parameters needed to build the SVM model:

- *Number of training instances*: For the Locomotion dataset, the size of the history instance set is 7312 instances. ν-SVM is intractable at this size, so a random subset of 1000 instances was selected. The negligible effect of this limitation is discussed in the evaluations of Section 5.6. The full complement of instances is used for the remaining datasets and instance sets.

- ν: The ν parameter controls the fraction of support vectors. The default ν parameter of .5 was used under normal circumstances. However, the ν parameter is occasionally too large to handle the specific data in the instance training set, making the calculation infeasible. In this case, ν is divided in half until this is no longer the case.

- C: The penalty factor was set to the default value of 1. Note that this is only needed for the regression case.

- *Kernel type*: The kernel type used is the radial basis function. This is the default kernel type in LibSVM, and all the other default parameters describing this function are used.

Learning algorithms

Whether classification or regression, the model is learned for the hyperplane that best minimizes the error of the training instances. There are two instances where the classifier/regression function cannot be built. First, if the output attribute is the same value

for all instances, that value is automatically returned. Second, if there are any predictor attributes where all instances have the same value, that attribute is removed when building the classifier for that test timestamp. The latter approach is an enhancement for ν-SVM, but it is necessary for the MultiBoostAB method to function at all when presented with these "unary" attributes.

Reasoning method inference

When a new problem instance is introduced, the classification algorithm chooses the binary SVM classifier with the greatest value or the regression algorithm returns the value of the function of the hyperplane defined in the model.

3.4.4 MultiBoost

For the ensemble methods, a form of MultiBoosting [141] was selected as it allows each learner to consider every instance of the dataset instead of possibly missing some with a resampling-based technique. MultiBoosting is a form of AdaBoost that combines bagging with boosting and performs with less error over a majority of datasets than AdaBoost or bagging by itself. The specific form of bagging used is called *wagging*. Wagging is short for weight aggregation, and instead of sampling from the dataset, it randomly assigns weights to each instance. In MultiBoosting, n subcommittees of base learners are formed with each committee given the dataset created by performing wagging on the original training data, i.e., each subcommittee gets the same dataset but with a different set of initial weights randomly assigned to its instances. For each subcommittee, a base learner is trained on the dataset that attempts to minimize the error with respect to the distribution of weights, i.e., misclassifying an instance with greater weight would have more effect on the error. The weighted error on the training set is saved for the classifier. The classifier's error is calculated by averaging the current weight of each misclassified instance. For the next base learner training iteration, the instance weights are decreased less for misclassified instances of the current iteration than they are for correctly classified instances of the current iteration. When all the iterations are complete, there are n subcommittees with n base learners for a total number of $n \times n$ learners. The output for the ensemble is then a weighted voting approach where the weight is a function of the error of the base learner on the training

set. Regression is performed exactly the same except output classes then represent subintervals from discretized numeric attributes.

Parameters

There are a number of parameters that need to be estimated for the model:

- *Base learner*: The base learner used for the MultiBoosting implementation is the decision tree variation called a decision stump. This is the default base learner and it is an unstable algorithm that maximizes the variation of the learners that make up the ensemble [43]. Unstable algorithms are those that exhibit large variation in results with a small variation in input.

- *Number of instances*: The Locomotion dataset's history instance set size is 7312 instances. Like ν-SVM, MultiBoostAB is intractable at this size, so a random subset of 1000 instances was selected. The negligible effect of this limitation is discussed in the evaluations of Section 5.6. The full complement of instances was used for the remaining datasets and instance sets.

- *Number of iterations*: This is the total number of base learners generated, and it is approximately the square of the number of subcommittees in the MultiBoosting ensemble. Ten iterations are used for the evaluations in this thesis which is also the default in the implementation of MultiBoostAB. Webb posits that ten iterations is sufficient, as most of the benefits of MultiBoosting are gained with the first few iterations [141].

- *Equal frequency discretization*: Unlike the equal width discretization approach used in the naive Bayes implementation, an equal frequency approach is used in MultiBoosting to discretize the numeric outputs. This is the default form of the discretization method and performs reasonably well.

Learning algorithms

Specifically, the implementation is MultiBoostAB from the Weka machine learning toolkit (Version 3.7.3). Like ν-SVM, the classifier/regression function cannot be built if the output attribute or any predictor attributes are the same value for all instances.

If they are all the same, that value is automatically returned or the predictive attribute removed, respectively. In addition, the MultiBoostAB implementation cannot handle non-numeric or multi-class output. To counteract these deficiencies, the Weka wrappers of MultiClassClassifier and RegressionByDiscretization are used. The MultiClassClassifier uses the binary-classifier-to-multiclass-classifier approach described for SVM classifiers (Section 3.3.6). For RegressionByDiscretization, the output value is the mean class value of the most probable discretized output interval.

Reasoning method inference

When a new problem instance is introduced to the ensemble learner, a weighted voting approach is used to select the output value or class. This weight is a function of how well the base learner performed on the dataset on which it was trained.

3.4.5 Context Selection Method

Context selection methods (Section 2.1) focus mainly on maximizing selection parameters such as timeliness, uncertainty, trust, and logistical properties. There are no such concerns with the datasets as all of these instances are identical with respect to these parameters. With these parameters all equal, the limitations of the context selection methods become more apparent. In this situation, context selection methods randomly choose a value from the instance set.

As context selection methods select randomly in these situations, the implementation of this learning method is simply a selection of an instance from the training set at random. As such, there is no need to define in great detail the parameters, learning algorithm, or reasoning method inference.

3.5 Chapter Summary

This chapter completed the first three steps of the learning method evaluation process and at least partially addressed four of the five research question criteria. The remaining criterion, instance set selection, will be addressed directly by the evaluations. Three different datasets were chosen to increase the external validity of the results (the "select data" step) and the datasets' attributes used for the evaluations were chosen

(the "choose attributes" step). The single-user conversion procedure answered the first research question (RQ-1) which allows the evaluations to run on realistic multiuser datasets (Section 3.2.4). Four learning methods that aid situational relevance were also chosen in order to best answer the research questions of this thesis (the "choose learning method" step). The implementations of all five learning methods, including the random context selection method, were also specified (Section 3.4). The random context selection method serves both as a representative of existing approaches to context acquisition from other devices and as a baseline against which to evaluate the other learning methods.

In the next two chapters, the remaining two steps in the learning method evaluation process are completed and the research question criteria more fully addressed. The manner in which each experiment will be implemented, the measure representing the overall accuracy by which the different evaluation alternatives will be judged, and how the results of those experiments will be analyzed are presented in the next chapter.

Chapter 4

Experiment Design

The previous chapter defined the inputs into the evaluations, i.e., the learning methods to be used and the datasets on which they will be trained and evaluated. This chapter focuses on how the evaluations will be run in order to answer the research questions introduced in the first chapter (see Section 1.2.3). The overall goal of the evaluations is to demonstrate the utility of using other users' information to improve the overall accuracy of context acquisition. This is accomplished through experiments that compare the overall accuracy of different approaches to using other users' information, including not using other users' information at all. Depending on the overall accuracy of the alternatives used for the specific research question being addressed, a conclusion is drawn about which method performs best. This chapter defines the evaluation setup that allows these claims to be made with a high degree of validity.

The alternatives tested in the experiments and the names used to refer to those alternatives are listed in Table 4.1. Each of the five learning methods introduced in the last chapter (see Section 3.4) are combined with the different instance sources, i.e., the application user's previous experience, other users, and a union of both these sets, to form fifteen different experiment alternatives. For example, the CBR-HISTORY alternative is the name of the alternative using the CBR approach applied only to the previous experience of the application's user and the BAYES-BOTH alternative is the name of the alternative that trains the naive Bayes classifier implementation of Section 3.4 on a combination of both other users' information and the previous experience of the application's user.

This chapter continues to follow the learning method evaluation process presented

Learning Method	Instance Sources		
	Other Users	Previous Experience	Both
Context Selection Method	RANDOM-OTHERS	RANDOM-HISTORY	RANDOM-BOTH
CBR Approach	CBR-OTHERS	CBR-HISTORY	CBR-BOTH
Naive Bayes Classifier	BAYES-OTHERS	BAYES-HISTORY	BAYES-BOTH
ν-SVM	SVM-OTHERS	SVM-HISTORY	SVM-BOTH
MultiBoosting	BOOST-OTHERS	BOOST-HISTORY	BOOST-BOTH

Table 4.1: Names of experiment alternatives.

in the introduction to Chapter 3. Specifically, the "train model" step is completed in this chapter and the "evaluate trained model" step is continued here and in the next chapter. While the last chapter selected the study inputs based on all the research question criteria, this chapter will proceed through the final two steps by focusing only on the research question criterion of validity. This facilitates the maximization of the validity of the experiments and the analysis of their results. The steps of the learning method evaluation process and the specific types of the validity addressed by each of the next three sections are shown in the overview of Table 4.2.

The manner in which the dataset is divided into the training and test sets is presented in the dataset partitioning section (Section 4.1). This section completes the "train model" step by defining the situation instances on which the learning models are trained (see Section 4.1.1). The training sets are defined so that the research questions address both full and sparse instance sets, i.e., small numbers of other users and limited previous experience. The section also addresses a part of the "evaluate trained model" step by defining the portion of the dataset to be used as the test set (see Section 4.1.2).

Section of Chapter	Process Steps	Validity Types
Dataset Partitioning (Section 4.1)	Train model Evaluate trained model	External Conclusion
Measuring Overall Accuracy (Section 4.2)	Evaluate trained model	Construct Conclusion
Analyzing Results (Section 4.3)	Evaluate trained model	Conclusion

Table 4.2: The learning method evaluation process step(s) and the threat(s) to validity addressed by each section.

Both the training and test sets are defined to maximize the validity of the evaluations. Specifically, this section addresses external validity by maximizing the diversity of the users, scenarios, situation instances, and attribute types and scales tested [145]. It also confronts conclusion validity threats by maximizing the number of observations, thereby increasing the power of the statistical tests used to analyze the results. A more powerful test is better able to spot differences between alternatives than a less powerful test [90].

Continuing the "evaluate trained model" step, the definition of "overall accuracy" and how it is measured is discussed in Section 4.2. This requires good construct validity, i.e., matching the theoretical construct of "overall accuracy" to a quantity that can be precisely measured. A specific threat to conclusion validity (i.e., "violated assumptions of statistical tests" [31]) is also addressed.

Also addressing the "evaluate trained model" step, Section 4.3 defines the manner in which the measurements of overall accuracy returned from the experiments should be analyzed in order to maximize conclusion validity. The section discusses both parametric and non-parametric statistical approaches and the assumptions that the measured data must meet in order to correctly use them.

4.1 Dataset Partitioning

The goal of this section is to define how the inputs of the last chapter are utilized in the experiments. The selected datasets are partitioned into training sets and test sets in order to train and test, respectively, the selected learning method inputs [87]. The training set divisions correspond to the three instance sources listed in Table 4.1. The test sets contain individual test instances, each of which is used by the alternatives to estimate the value of a specific attribute type for that test instance.

For all the datasets, the situation instances have the following format:

$$T_x : a_1, a_2, \cdots, a_k \tag{4.1}$$

Equation 4.1 shows a situation instance with k attributes, i.e., k different types of context information. a_i represents the value for the ith context attribute and T_x is the time that the situation instance represents, i.e., the time at which the values for all the context attributes were recorded.

In addition to partitioning the datasets to address the research questions, they are also divided in such a way as to maximize conclusion and external validity. These are maximized by increasing both the number of observations and the diversity of users, scenarios, situations, and attributes from which these observations are drawn. The conclusion validity threat of "low statistical power" is concerned with avoiding Type II error [31], i.e., falsely concluding that two alternatives are not different when they are, (see Section 4.3). Type II error is minimized by increasing the number of observations.

Increasing the number of observations also aids conclusion validity. Matching the diversity of the users and situations tested to better reflect the diversity of the general population to which the experiment conclusions are applied aids external validity [145]. Specifically, the external validity threat "interaction of setting and treatment" is mitigated by varying the setting of the experiment [31]. The selection of multiple datasets from different pervasive applications in Section 3.2.3 also aids the goal of varying the setting [85].

This goal is further advanced by increasing the diversity of the training and test sets within the dataset. Increasing diversity is aided by including many different:

- *Scenarios*: The different scenarios being recorded in a dataset.

Dataset Name	Previous Experience	Other Users	Both Instances	Test Iterations	Iteration Interval
Nokia	592	41	633	178	10
Locomotion	7312	12	7324	220	100
Helsinki	21	239	260	64	1

Table 4.3: Dataset partition statistics for each dataset.

- *Users*: Users within those scenarios.[1]
- *Situation instances*: Different situations at different times in the run of a single user.
- *Attribute types and scales*: Each situation has several attribute types, e.g., location and temperature, that are both estimated and used as predictors. The collection of attribute types across the three datasets also has different scales, i.e., categorical and numerical attributes.

The variance of scenarios, users, situation instances, and attribute types and scales will all be addressed in the remainder of this section. Increasing the number of observations in general will also be a focus.

4.1.1 Training Sets

Partitioning the datasets into training sets is necessary to complete the "train model" step. In order to keep the experiments as consistent as possible within a dataset despite any variation in the run length between users, all users' runs are truncated to be equal to the user with the shortest run time. For example, the user with the shortest run in the Helsinki dataset has 85 time iterations and any iterations beyond 85 are ignored for all users of that dataset (See Table 4.3).

The datasets are divided into three training sets, all of which are necessary to answer the research questions in Section 1.2.3:

[1]This is not strictly true for all the users of the datasets in these evaluations as a result of using different runs from a single user to represent several different users. However, as each run is not identical, there is still variety between each run (see Section 3.2.4.1 and Appendix C.3).

Other users' information (OTHERS) This is the situation instance set that describes the current situation of all the other users that are available to the application's user. The size of this set depends on the number of other users that are in communication range of the device and that advertise the desired output context type. There are no assumptions on the structure of the previous experience of other users (see Section 3.1). As a consequence, the only situation instances that are assumed to be available from other users are their current ones. To reflect this, only the situation instances for the current time are available from the subset of other users under consideration. The current situation of other users that make up this instance set is reflected in Figure 4.1 for the test instance's current timestamp T_m. The start time for each other user run is randomized so that the activities of each run are less likely to be synchronized. The randomization shifts the start time of each run up to 10% of the total run's length. This essentially brings latter instances from what would be used for that run's previous experience into the set of the other user. For example, the other users' timestamps of T_{m-27} and T_{m-13} line up with the current user's timestamp (T_m) in Figure 4.1, indicating that the other users' start times are 27 and 13 instances, respectively, before the current user's start time. A subset of runs is employed to address sparse environments (RQ-6), i.e., when a smaller number of other users are available for context acquisition. An approach similar to that used by the "sparse case-base" experiments in MCBR [71, 72] is used to determine the subset of other users, i.e., the subset is randomly chosen at each timestamp from the full collection of other users.

Previous experience (HISTORY) This is the situation instance set that describes the situations previously encountered by the application's user. The size of this set depends on the extent of the user's past situation instances with the desired output context type that has been saved onto the device. The previous experience of a user was chosen to be the first 25% of instances from that user's run. For example, each user in the Helsinki dataset has 21 instances representing the previous experience, i.e., around one fourth of the 85 instances. The size of one fourth of the instances was chosen in order to give a previous experience (the first subset) that covers a variety of situation instances (addressing the external

validity threat) while also maximizing the size of the instance set available for the test set (the second subset). Maximizing the size of the second subset enables an increase in the variety of situation instances encountered (addressing the external validity threat). A large range of time for the test set also emphasizes the potential differences between the situations of the previous experience and that of the test set as the differences in timestamps of the test instances and those of the previous experience increase. This difference reflects the changing situations and unfamiliar environments common to pervasive applications (see Section 1.2.1). Finally, just as subsets of the other users set can be used to test sparse environments, randomly selected subsets of the instances that make up the previous experience can be used to test situations where the size of the previous experience set is limited.

Union of previous experience and other users' information (BOTH) The union set is the combination of an application user's previous experience and other users' information available in the environment. The union set is used to aid sparse instance sets and to augment the application user's previous experience. The degree to which the union set is useful for each dataset is explored in the latter research questions (RQ-4, RQ-5, and RQ-6).

The comparison of approaches using these instance sets is similar to the evaluation of MCBR [71, 74]. Although not an evaluation of pervasive applications, it compared prediction accuracy for instance sets from two different sources, i.e., two separate case bases, as well as the accuracy for those sources combined.

As a final note, the CBR approach has an additional requirement that the other learning methods do not, i.e., it needs a training set to configure the weights. For the selected datasets and training partitions, abbreviated exploratory tests revealed that the instance set source used to train the weights did not affect the results in a meaningful way (see Appendix B).[2] That is, most differences were not statistically significant. For the few that were, the difference between instance source tests was a small fraction of the difference between the overall alternatives, i.e., the difference was not meaningful for the evaluations of this thesis. Given that there is not a meaningful difference

[2]For a brief discussion on possible causes of the consistency across instance sources, see Section 6.2.3

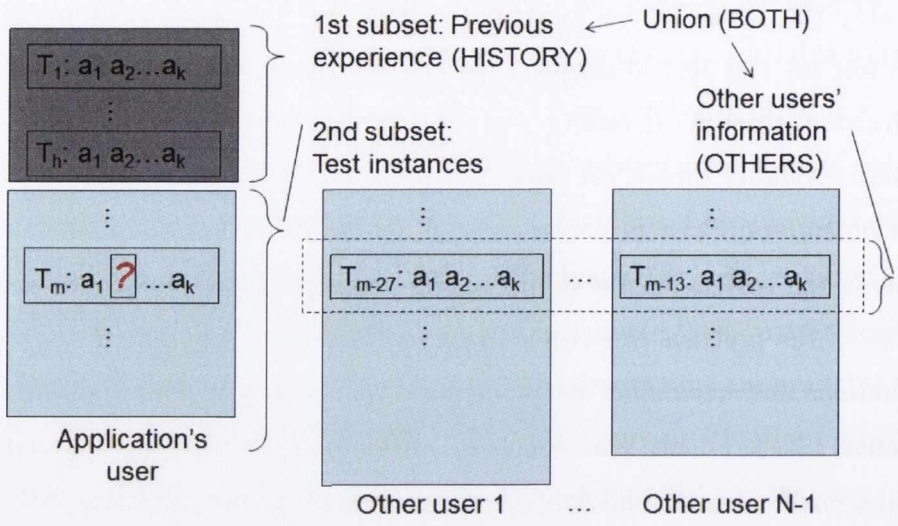

Figure 4.1: Dataset partition summary.

between instance source sets, the previous experience instance set was decided upon for the weight training set. The use of the previous experience training set for weight training enables an increase in the diversity of the tests while keeping the evaluation computation times manageable. That is, this choice allows more test instances to be performed for the evaluations as the weight training process only needs to be performed once per attribute for each user. Unlike the previous experience instance set, the other instance sets do not remain constant throughout each test instance in a test subset and as such would require the relatively time-consuming weight training step to be repeated at each test instance.

4.1.2 Test Sets

Partitioning the datasets to form the test sets begins the "evaluate trained model" step. The division of a user's run between previous experience and instances available for the test set has already addressed some of the validity requirements for the test set. That is, it attempts to maximize the number of observations as well as the diversity of the situation instances encountered. The final decisions to be made with regard to dataset partitioning are choosing which of the second subset's situation instances within a user's run should be used as the test set and selecting which user runs should be in the test set.

To fully utilize the diversity of the situation instances, the test instances are selected by intervals from the entire range of the second subset. That is, the situation instance at every *ith* timestamp is included in the test set. The time iteration intervals chosen for each dataset are shown in Table 4.3. While the entire second subset is used for the Helsinki dataset, a fraction of the respective second subsets are used for the remaining two datasets. This is mostly in the interest of mitigating excessive experiment computation time.[3] However, as a collection of test instances from the entire range of the test set is selected, diversity is still maintained. It is also important to note that by defining the situation instances for the test set, this fully defines the corresponding situation instances for the instance set of other users' information. That is, the timestamp for each instance in the test set is the same as the timestamp for the situation instances used for the other users' information instance set minus the random variation in start time (see T_m, T_{m-27}, and T_{m-13} in Figure 4.1).

To fully generalize the evaluations across all attribute types in a dataset, each test instance is used to estimate a value for each of the k attributes. Each test instance at a specific time interval in a run (e.g., T_m in Figure 4.1) is treated as k different test instances, each one used to estimate a different attribute type's value. As a consequence of evaluating across all attribute types for the three datasets, an alternative is also tested across the different scales of those types, i.e., categorical and numerical (see Section 3.2.2).

There are a number of approaches available to determine how user runs should be assigned to the test set and to the training set, e.g., a single random partition, k-fold cross-validation, and leave-one-out cross-validation [87, 7]. Leave-one-out cross-validation is a special case of k-fold cross-validation with k equal to the number of users in the dataset. While there is an increase in the evaluation run-time compared to the other approaches, leave-one-out cross-validation satisfies a number of the identified requirements for conclusion and external validity. It maximizes the number of observations by having a set of observations for every user's run. This also has the effect of maximizing the diversity of observations for both the users and the scenarios

[3]It is worth noting that the power of the statistical tests will not be affected by this decision. While the diversity may be slightly affected, the test instances are a large sample from the full range of the second subset.

those users are executing. It also presents an advantage over the other approaches for datasets with a smaller number of users [87] (such as the Locomotion dataset, which only has 13 total users).

An illustration of using one user's run to evaluate the alternatives is shown in Figure 4.1. This is then repeated for the other $n-1$ users in order to get n sets of observations. The evaluation in MCBR [71] uses a leave-one-out cross-validation approach similar to this where the data instances are travel bookings rather than users.

4.2 Measuring Overall Accuracy

The last section identified several test instances over the duration of a run for each user to make up the test set. These instances were further divided by their attributes and each of these attributes are estimated by each of the fifteen learning method-instance set combinations that make up the alternatives (see Table 4.1). This results in fifteen different estimations of the attribute's value for each test instance. The totality of test instances represents different users, scenarios, attributes, and situations. The last section concentrated on providing the evaluations with this wide variety of test instances to maximize external validity.

This section continues the "evaluate trained model" step of the learning method evaluation process by taking the diverse set of estimated values from the last section and creating an overarching measure of accuracy that will allow the comparisons of the different alternatives. The first subsection determines which measure best reflects the individual measures of accuracy for a single estimated value. The degree to which a researcher validly relates the variables of the experiment (e.g., individual accuracy measures) to the theoretical construct (e.g., "accuracy") concerns construct validity [31] (see Section 1.1.3). The second subsection aggregates the individual measurements in order to obtain the performance measure termed *average estimation error* that measures the theoretical construct "overall accuracy" required by the research questions in a manner that maximizes conclusion and construct validity.

4.2.1 Individual Measures of Accuracy

The accuracy of a measurement is how close it is to the actual phenomenon it describes. In the evaluations of this thesis, the accuracy of the value estimated by the learning methods reflects how near it is to the actual value. Along with the estimation of an attribute's value, the actual value is also available, i.e., the value of the attribute type already given in the situation instance in the original dataset. These two values are used to calculate the error of the estimation for both categorical and numerical values. The alternative with the lowest overall error is then the most accurate overall. The fact that the most accurate alternative is identified by the alternative with the lowest overall error means that the measure of error reflects good construct validity.

For the categorical attributes, the classification error rate is used to reflect the overall accuracy. The *classification error rate* is the percentage of the test set assigned to the wrong category and is the simplest measure of classifier performance [37]. To facilitate aggregating the individual measures of accuracy into the classification error rate, the individual measure used for categorical attributes is a binary one that is very similar to the categorical local similarity measure of the CBR approach (see Section 3.4). The error is the value 0 for a correct classification and the value 1 for any misclassification. The mean of these individual errors for a test set gives the classification error rate for the alternative. It is worth noting that like the local similarity measure, there is no differentiation between two misclassifications. For example, even though one cell tower is much closer in proximity to the true cell tower than another one for the Nokia dataset, both incorrect cell tower IDs receive the same error, i.e., the error value of 1.

For the numerical attributes, there is more information than whether the estimated value is the correct one and that information is inherent in the scale of the attributes. This thesis uses a measure of error that reflects the magnitude of the difference between the estimated value and the actual value. The average of the estimation error for a single numerical attribute's test set is this average error.

The magnitude, or absolute value, of the error was used instead of a difference that indicates direction for two reasons. First, if the estimations are unbiased, the means of the estimation errors would not be a reflection of the overall accuracy (as this would approach zero as the number of estimations increased). Second, the answers

to the research questions do not concern whether the learning methods overestimate or underestimate the value, so magnitude is the only property of the estimation error that is interesting for the research questions being addressed.

4.2.2 Overall Accuracy

With the definition of the individual measures of accuracy for each attribute type across each situation instance in the test set defined, all that remains is combining them into one measure that reflects the "overall accuracy" for each alternative. This measure of overall accuracy is selected to maximize both construct and conclusion validity.

As will be seen in Section 4.3, all of the statistical tests used to analyze the results require that the observations are independent. This is not the case for the estimated error for all the situation instances in a single run for a user. Particularly with the slow changing attributes, like temperature and pressure, of the Nokia dataset, performance in one instance is closely correlated to performance in instances that precede and follow that instance. In order to minimize the conclusion validity threat of "violated assumptions of statistical tests", this dependence must be removed. This is achieved by averaging the values of estimation error for a specific attribute over time. The means exhibit good construct validity as well, as there is now one value per each attribute that reflects the "overall accuracy" for that attribute.

This leaves an average estimation error for each attribute type for each user. For the answers to the research questions of Section 1.2.3, the knowledge of which individual attributes are best estimated by each alternative is not necessary. Rather, the manner in which the alternative performs across all attributes is needed. In order to get a single value that reflects the "overall accuracy" for a user, the estimation error values of the different attribute types must be combined as well. Averaging the numerical values together will not give the desired results, however, as the attributes have different dimensions. Even if the numerical attributes are normalized, there is still the problem of how to compare the binary values of the categorical attributes to the numerical ones of the numerical attributes.

A solution to these problems is suggested by Gelman [47]. In his work, it is suggested that binary and numerical values are directly comparable by dividing the numerical attributes by twice their standard deviation. The standard deviation for each

attribute is calculated from the values for that attribute over the entire dataset. This division provides a way to directly combine the values across attribute types.

The combination of all the estimation errors for each user results in one measure for the "overall accuracy" of each alternative for each user. This measure of "overall accuracy" is termed *average estimation error*, and the measure is used for the statistical analysis described in Section 4.3. The average estimation error has good construct validity as an alternative with a lower error has the better "overall accuracy".

4.3 Analyzing Results

Drawing valid conclusions from the results of the evaluations continues the "evaluate trained model" step. The application of statistical methods to the results is used to maximize conclusion validity. The degree to which the difference between the measures of two alternatives are a reflection of the general population under study and not an artifact of the specific dataset used for testing are conveyed by comparative statistical methods. For example, the statistic that the evaluations of this thesis are comparing is the mean of the average estimation error of an alternative for each user. If the difference between the means of the two alternatives is 1 unit but both alternatives have a standard deviation of 4 units, the difference in the alternatives could very easily be an artifact of the particular sample used for testing. This means that a difference in the means of the two alternatives cannot be safely concluded about the general population by considering these results.

The concept of statistical significance employs probability theory to determine the likelihood that a difference is just an artifact of the data being evaluated rather than of the general population from which the data was drawn. *Statistically significant* is a term applied to an observed effect that is so large that it would rarely occur by chance [90]. It is accompanied by a *significance level* (α) which is the probability that the test predicts that there is an actual difference between two alternatives when there is not one. For example, a standard value for alpha is .05 [90]. If that significance level is employed in a test, then one out of twenty times the test will show that there is a statistically significant difference between alternatives when there is none. Statistical significance is directly related to the threats of "low statistical power" and "fishing and

error rate" discussed below. The experiments of this thesis use a repeated measures design which is an experiment where each subject is used to test all the alternatives. This is an experimental design used to increase the statistical power of a test by removing variability between subjects from the experiment [143].

Almost all the threats to conclusion validity [145, 31] have to be dealt with by the evaluations of this thesis. Consequently, the following lists each threat and how they are minimized by the experiment design of this thesis:

- *Low statistical power*: This occurs when there is a Type II error, i.e., when an analysis concludes that there is no statistically significant difference between alternatives when there actually is one. As noted above, increasing the number of observations decreases the chance of Type II error. Additionally, this threat is decreased by the selection of more powerful statistical tests to analyze the data. A more powerful test is better able to spot differences between alternatives than a less powerful test. For example, a parametric test is generally more powerful than an equivalent non-parametric test [14]. In addition, the repeated measures design increases statistical power by decreasing the error variance from independent-group designs [143]. This allows the test to focus on the variability of the alternatives rather than confounding it with the variability inherent between different users.

- *Violated assumptions of statistical tests*: While parametric tests are generally more powerful, their use is more restricted as any data they are used to analyze must meet the assumptions of the test's model. For example, one assumption of both repeated measures tests described below is that the data is independent, i.e., that each subsequent observation is not affected by the observation that preceded it [90]. To minimize the violation of the independence assumption, the measure described in the last section averaged all the observations for a single user's run (see Section 4.2). In the evaluations of this thesis, the observations are analyzed to ensure they meet all of the assumptions of the statistical tests, and that the most powerful test for which these assumptions are valid is used.

- *Fishing and the error rate*: The "error rate" refers to the possibility of making a Type I error, i.e., when there is no statistically significant difference between

alternatives but a difference is concluded anyway. A pairwise test that compares two alternatives cannot be used multiple times with the same significance level and still be assumed to maintain the overall significance level. For example, making three comparisons with a significance level of .05 means that the total significance level is $(1 - .05)^3$ which is .14. This total significance level is much higher than the intended significance level, i.e., .14 is much greater than .05. In the evaluations of this thesis, the threat is mitigated by the repeated measures tests used (i.e., repeated measures ANOVA and the Friedman test) and the post-hoc multiple comparison tests used (i.e., the Tukey test and Wilcoxon signed-rank test with Bonferonni correction). Unlike the Tukey test, the Wilcoxon signed-rank test does not adjust the significance level based on the number of comparisons, so it is used in conjunction with a conservative error correction (i.e., the Bonferonni correction).

- *Reliability of measures*: This threat is eliminated as the outcomes are derived by computer programs and are exactly the same whenever the alternatives are reapplied (given the same random seed).

- *Reliability of treatment implementation*: This threat is eliminated as a processor applies the alternatives and so they are applied in exactly the same manner each time and for all users.

- *Random irrelevancies in experimental setting*: This threat is eliminated as there are no elements outside the experiment runs that affect the results.

- *Random heterogeneity of subjects*: This threat is minimized as the users are blocked in the repeated measures design [143]. Blocking controls for experimental factors that are not of primary interest, e.g., the inherent variability between users. It is used to eliminate the undesired effect so it does not interfere with the study [145].

A lack of statistical analysis was noted as a limitation to most works in the state of the art in Chapter 2. The experiments in [109] were the only ones that employed statistical analysis (using the non-parametric Wilcoxon signed rank test). Without the proper application of statistical analysis, these approaches are in danger of making

conclusions about the specific sample used in their experiments rather than that of the general population under consideration.

The remainder of this section briefly presents the statistical tests used in the analysis of the evaluations in this thesis. Specifically, both the parametric and non-parametric tests are summarized as well as the assumptions that each of the tests make about the data.

4.3.1 Parametric Tests

The parametric tests in this section have an underlying linear model. This model is shown in Equation 4.2, where y_{ij} is the actual average estimation error of the ith alternative for the jth user, μ_i is the mean of the average estimation errors for all users for the ith alternative, and e_{ij} is the error associated with the jth user for the ith alternative, i.e., how far off the mean of the estimation error for the alternative (μ_i) is from the actual estimation error (y_{ij}).

$$y_{ij} = \mu_i + e_{ij} \tag{4.2}$$

To use these parametric tests that assume an underlying linear model, the following assumptions need to hold [90]:

1. *Independence of measurements*: Each subsequent observation (y_{ij}) is not affected by the observation that preceded it.

2. *Normality*: The distribution of the residuals (the $e_{ij}s$ in Equation 4.2) is normal.

3. *Equal variances*: The variance of the data is the same for all alternatives.

The "independence of measurements" assumption has been previously addressed as the estimation errors were averaged across the entire range of time in the run for each user. This means the experiments only need to test the "normality" and "equal variances" assumptions. Normality can be observed via a normal quantile plot by ensuring that the data points form an approximately straight line. Moore et al. [90] and Weiss [143] argue that testing for normality is less important for large sample sizes, which is a direct result of the central limit theorem [90]. A "large sample size" is one that is at least 40 in Moore et al. and at least 30 in Mitchell [87]. This thesis uses 40 as the

cutoff point for a large sample size as it is more conservative, so there is less chance for a Type I error. For the "equal variances" assumption, Moore et al. also suggest a rule of thumb that if the alternative with the largest standard deviation is less than twice the alternative with the smallest standard deviation, equal variance may be assumed.

4.3.1.1 One-Way Repeated Measures ANOVA

One-way repeated measures ANOVA (Analysis of Variance) [143] is "one-way" because it has only one factor it is examining (i.e., the alternatives) and is a "repeated measure" test because each alternative is applied to the same subject in the same manner. ANOVA assumes the null hypothesis unless there is evidence to reject it. The *null hypothesis* (H_0) theorizes that all means are equal while the alternative hypothesis (H_A) theorizes that not all the alternative means are equal:

H_0:$\mu_1 = \mu_2 = \cdots = \mu_n$

H_A: Not all of the μ_is are equal

The μ_is are the means of the average estimation error for each alternative for all subjects. If the null hypothesis is not rejected, there is no perceived difference between the alternatives. If the null hypothesis is rejected, a post-hoc multiple comparison test is used in order to determine which alternatives differ from each other. It is important to note that performing the multiple comparison tests only makes sense when the null hypothesis is rejected.

4.3.1.2 Post-hoc Multiple Comparison: Tukey Test

If the null hypothesis is rejected in one-way repeated measures ANOVA, the Tukey test is run. The Tukey test allows the pairwise comparison of any two alternatives' means without affecting the experiment-wide significance level [143]. This allows conclusions to be drawn about the alternative that performs best while minimizing the conclusion validity threat "fishing and error rate". The hypotheses for each comparison are:

H_0:$\mu_a = \mu_b$

H_A:$\mu_a \neq \mu_b$

If the null hypothesis is rejected and the means of the two alternatives being compared are not equal, the means are compared to see which is less.

A smaller mean indicates an alternative with a lower average estimation error which means the alternative has better overall accuracy.

4.3.2 Non-parametric Tests

If the data do not meet the assumptions of the parametric tests, a non-parametric test is used instead. Non-parametric tests are generally less powerful than parametric tests, so a parametric test should be used if possible to minimize the "low statistical power" conclusion validity threat [14]. The Friedman and Quade tests are both options for the non-parametric test, but the Friedman test is the more powerful test when there are six or more alternatives [30]. The Friedman test was selected for this thesis as the evaluation tests fifteen alternatives.

The hypotheses for non-parametric tests are not phrased in terms of means as with the parametric tests. Particularly with skewed data, the mean is not a good measure of central tendency [90], and this is reflected by the use of median as a measure of central tendency in the outputs of the non-parametric test in Chapter 5. Instead of dealing with the average estimation error directly, the non-parametric tests in this section deal with ranks. That is, the data are placed in an ordered list according to their magnitude and then are replaced by their rank in the list. That is, the smallest value is replaced by 1, the second smallest value is replaced by 2, etc. Any significant difference between alternatives is reflected by differences in the sum of the ranks for an alternative.

The non-parametric tests discussed in this section remove the normality and equal variance assumptions from the parametric tests discussed in Section 4.3.1. The only assumptions for the non-parametric tests are that the observations are independent and that they are able to be ranked according to some criterion of interest [30]. Neither assumption needs to be tested in the evaluations of the next chapter as the measurements are already all independent and are also real numbers which can be naturally ranked.

4.3.2.1 Friedman Test

The Friedman test is the non-parametric equivalent to one-way repeated measures ANOVA [30]. As stated above, it is based on analysis of the ranks rather than means.

The hypotheses for the Friedman Test are as follows:

H_0: Average estimation errors have the same distribution for all alternatives

H_A: Average estimation errors are systematically higher for some alternatives than for others

4.3.2.2 Post-hoc Multiple Comparison: Wilcoxon Signed-Rank Test with Bonferonni Correction

The Wilcoxon signed-rank test is the non-parametric equivalent of the paired t-test. If the Friedman test rejects the null hypothesis, the Wilcoxon signed-rank test may be run. However, unlike the parametric Tukey test, a pairwise test like Wilcoxon must use an error rate correction as the significance level is only valid for one comparison [7]. The Bonferonni correction divides the overall significance level by the number of pairwise comparisons being performed [143]. This allows the comparisons to be performed while also combatting against the Type I errors of the conclusion validity threat of "fishing and the error rate".

The hypotheses for each pairwise comparison are:

H_0: Average estimation errors for the two alternatives do not differ

H_A: Average estimation error for one alternative is systematically larger than the other

4.4 Chapter Summary

This chapter completes the description of the design of the experiment, which was accomplished by partitioning the dataset into training and test sets, defining the average estimation error measure to reflect the theoretical construct of "overall accuracy", and specifying a way to analyze the differences between alternatives while maximizing conclusion validity. Specifying the training set also completed the "train model" step of the learning method evaluation process. All the other sections of this chapter contributed to the "evaluate trained model" step. In the next chapter, the "evaluate trained model" step will be completed as the actual experiments for each research question will be run

and the statistical analysis described in Section 4.3 will be applied to the results in order to answer the research questions.

Chapter 5

Study Evaluation

The last two chapters determined the inputs into the evaluations (Chapter 3) and the experiment design and analysis methods to be used in them (Chapter 4). This chapter focuses on using these elements to answer the research questions presented in Section 1.2.3. In doing so, the chapter completes the final step ("evaluate trained model") of the learning method evaluation process.

Research Question	Research Question Description	Section Addressed
RQ-1	Multiuser dataset procedure	Section 3.2.4
RQ-2	Situational relevance	Section 5.2
RQ-3	Previous experience versus other users	Section 5.3
RQ-4	Combining instance sets	Section 5.4
RQ-5	Instance set selection	Section 5.5
RQ-6	Instance set sparseness	Section 5.6

Table 5.1: The research questions and the sections in which they are addressed.

The answers to the research questions address the utility of using other users' information to aid pervasive application behavior. This is accomplished by measuring the performance of the learning methods and instance sources in the generated multiuser datasets representing pervasive environments. The alternative that exhibits the better "overall accuracy" in the comparisons will be the answer to the research question. Ta-

ble 5.1 briefly describes each of the research questions and lists the sections in which they are addressed.

In order to identify which alternative exhibits the better "overall accuracy" with good conclusion validity, the difference between the average estimation errors of the two alternatives must be statistically significant. In this chapter, the term "statistically significant" means that the p-value of the test is less than or equal to the significance level (α). The p-value is the probability of observing the data assuming the null-hypothesis and other model assumptions (parametric or nonparametric) are true. If the p-value is smaller than the significance level, it is assumed that there is a difference between the alternatives. That is, if the difference is statistically significant, it means that it is safe to reject the null hypothesis. A significance level of .05 was selected as this is a standard value [90]. This requires the data to give evidence against the null hypothesis that is so strong that observing the data when the null hypothesis is true would occur no more than 5% of the time [90].

Section 5.1 performs the analysis necessary to do direct comparisons between two alternatives, including performing the group tests and analyzing the data to determine whether parametric or non-parametric tests should be performed. The sections that follow address each of the five remaining research questions. RQ-2 to RQ-5 are questions dealing with the full environment, i.e., the non-sparse environments. Section 5.2 to Section 5.5 address RQ-2 to RQ-5. The sparse instance sets research question (RQ-6) is addressed in Section 5.6 and is run differently than the full environment research questions. It includes analysis of the performance in sparse environments and with sparse previous experience. The validity of the single-user conversion procedure in light of the results is explored in Section 5.7, and the chapter summary follows in Section 5.8.

5.1 Full Environment Overview

Research Question	Alternative Comparisons
RQ-2: Situational relevance (Section 5.2)	CBR-OTHERS vs RANDOM-OTHERS
	BAYES-OTHERS vs RANDOM-OTHERS
	SVM-OTHERS vs RANDOM-OTHERS
	BOOST-OTHERS vs RANDOM-OTHERS
RQ-3: Previous experience versus other users (Section 5.3)	CBR-OTHERS vs CBR-HISTORY
	BAYES-OTHERS vs BAYES-HISTORY
	SVM-OTHERS vs SVM-HISTORY
	BOOST-OTHERS vs BOOST-HISTORY
RQ-4: Combining instance sets (Section 5.4)	CBR-OTHERS vs CBR-BOTH
	BAYES-OTHERS vs BAYES-BOTH
	SVM-OTHERS vs SVM-BOTH
	BOOST-OTHERS vs BOOST-BOTH
RQ-5: Instance set selection (Section 5.5)	CBR-HISTORY vs CBR-BOTH
	BAYES-HISTORY vs BAYES-BOTH
	SVM-HISTORY vs SVM-BOTH
	BOOST-HISTORY vs BOOST-BOTH

Table 5.2: The comparisons required to answer each research question.

The term "full environment" is how this thesis identifies the tests in which all of the instances in the instance sets are utilized for training the learning models. This is to differentiate it from the tests that evaluate performance in a "sparse environment" where only a subset of the other users or previous experience sets are utilized (see Section 5.6).

The specific comparisons performed are those that address the corresponding research question (see Table 5.2). The comparisons are set before any experiments are run in order to lay out exactly how the question will be answered and to fix the number of comparisons. The latter is important for the non-parametric multiple comparison test as the significance level of that test is dependent on the number of specific, pairwise comparisons that are performed. Minimizing the number of pairwise comparisons

increases the power of the test, and thereby minimizes the threat to conclusion validity of "low statistical power".

There are 105 different possible pairwise comparisons as there are fifteen alternatives. Since the evaluations only require sixteen of those comparisons, the Bonferonni correction for the significance level of .05 is .003125 (i.e., .05/16) instead of the much less powerful .000476 (i.e., .05/105) with all 105 comparisons. Each of the sixteen pairwise comparisons will be done at the significance level of .003125 in order to keep the overall significance level for the full environment evaluation at the desired .05.

The group tests must be run before performing the specific comparisons. This means one-way repeated ANOVA for the parametric case and the Friedman test for the non-parametric case. These tests show if there is a statistically significant difference among the fifteen alternatives. If this is the case, then the null hypothesis (H_0) is rejected and the multiple comparison tests can be run to discover which alternatives are statistically different.

Figure 5.1, Figure 5.5, and Figure 5.7 show the means for the fifteen different alternatives in the full environment. The differences in height between the different alternatives in the bar graph indicate different average estimation error for each alternative. The lower the average estimation error an alternative has, the more that alternative demonstrates a greater overall accuracy. While there are clearly perceptible differences in the magnitude of estimation error in the graphs, the statistical tests need to be run in order to determine if those differences are statistically significant. Alternatively, values that appear to be the same may have differences that are actually statistically significant.

The group tests are compared in the remainder of this section to see if the alternatives exhibit any statistically significant difference. Section 5.1.1 presents the one-way repeated measures ANOVA test for all three datasets and conducts analysis of how well the data meets the model assumptions described in Section 4.3.1, i.e., the normality and equal variance assumptions. The section also presents the results of the non-parametric Friedman tests. Section 5.1.2 includes an overview of the specific comparisons that are performed in order to answer the full environment research questions in the subsequent sections.

While the results for each test are presented, the statistical outputs for the cor-

responding tests are only given for the Nokia dataset. In the interest of conciseness, the output for the analysis of the remaining two datasets will not be listed in this chapter. A full account of the statistical outputs for all the datasets can be examined in Appendix A.

5.1.1 Group Test Results

This subsection will examine the ANOVA and Friedman tests for each dataset including all the tests' assumptions. The summary of the results of the parametric and non-parametric tests are shown in Table 5.3. Both of the tests have a statistic whose significance is given by the p-value.

For the Friedman test, the test statistic is the "S statistic", and it is based on the ranks of each alternative. The significance of the "S statistic" (and the "F statistic" below) is reflected by the p-value. That is, if the p-value is less than the desired significance level of .05, this indicates that it is safe to reject the null hypothesis, i.e., that there is a systematic difference between the alternatives.

For the ANOVA test, the statistic is the "F statistic" and it represents the ratio of variation attributed to the alternative and the variation due to each user within the alternatives [90]. An F statistic approximately equal to 1 indicates that the null hypothesis is true and the null hypothesis is rejected when it is significantly larger than 1.

The summary of the different alternatives' standard deviations is listed in Table 5.4. The '*'s next to an entry indicate an alternative whose standard deviation is more than twice as large as the alternative with the smallest standard deviation. According to the rule of thumb from Moore et al. [90], this indicates that the Locomotion and the Helsinki datasets violate the equal variances assumption and the parametric tests have questionable applicability for the results of these datasets.

	Nokia	Locomotion	Helsinki
Parametric Statistics			
p-value	0.000	0.000	0.000
F statistic	53.11	324.29	1068.66
Non-parametric Statistics			
p-value	0.000	0.000	0.000
Test statistic (S)	349.59	171.35	2758.51

Table 5.3: Group test results for both the parametric and non-parametric tests.

Alternative	Nokia	Locomotion	Helsinki
CBR-OTHERS	0.0570	0.0296*	0.0249
CBR-HISTORY	0.0949	0.0215*	0.0579*
CBR-BOTH	0.0651	0.0216*	0.0255
BAYES-OTHERS	0.1027	0.0248*	0.0389
BAYES-HISTORY	0.0770	0.0302*	0.0413
BAYES-BOTH	0.0696	0.0299*	0.0395
SVM-OTHERS	0.0724	0.0473*	0.0285
SVM-HISTORY	0.0790	0.0107	0.0451
SVM-BOTH	0.0631	0.0137	0.0273
BOOST-OTHERS	0.0662	0.0324*	0.0290
BOOST-HISTORY	0.0782	0.0164	0.0482
BOOST-BOTH	0.0796	0.0164	0.0290
RANDOM-OTHERS	0.0611	0.0247*	0.0343
RANDOM-HISTORY	0.0653	0.0222*	0.0606*
RANDOM-BOTH	0.0606	0.0199	0.0342

Table 5.4: Standard deviations for each alternative. '*'s denote standard deviations that are more than twice that of the smallest standard deviation.

Nokia Dataset

The overview of the means of the different alternatives for the Nokia dataset is shown in Figure 5.1.[1] The statistical output for the ANOVA test is shown in Figure 5.2. The "Alternative" p-value (0.000) is less than the desired significance level of .05. This indicates that it is safe to reject the null hypothesis, i.e., that there is a systematic difference between the alternatives. This indicates that it is safe to run the multiple comparison tests to discover which alternatives perform better than others.

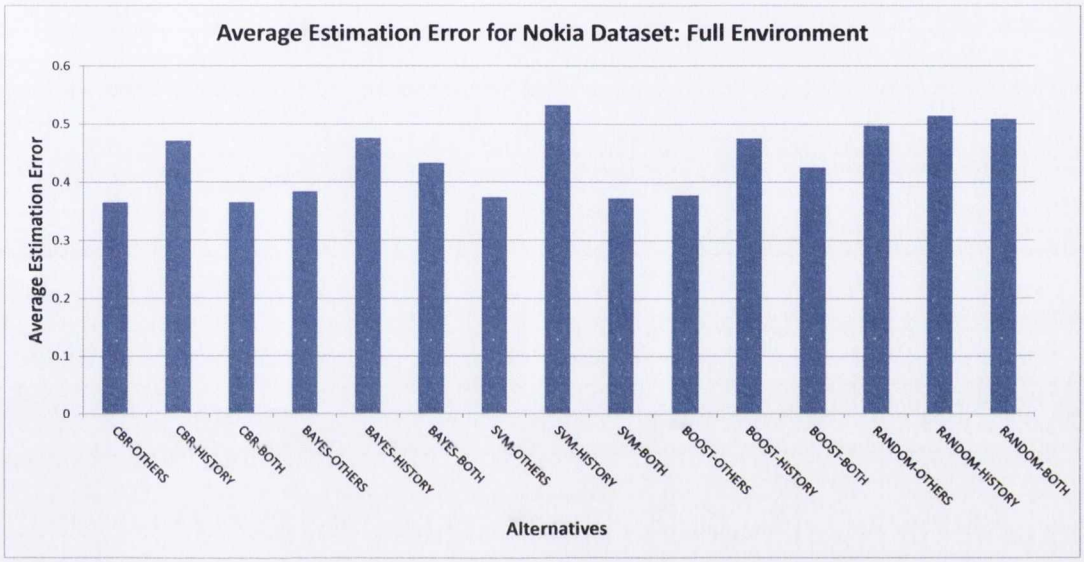

Figure 5.1: Full environment alternative means for the Nokia dataset.

[1]The average estimation error of the alternatives for each individual attribute for the Nokia dataset and the other datasets are given in Appendix C.1.

Analysis of Variance for Average Estimation Error, using Adjusted SS for Tests

Source	DF	Seq SS	Adj SS	Adj MS	F	P
Alternative	14	2.217392	2.217392	0.158385	53.11	0.000
User	41	1.640876	1.640876	0.040021	13.42	0.000
Error	574	1.711921	1.711921	0.002982		
Total	629	5.570189				

S = 0.0546117 R–Sq = 69.27% R–Sq(adj) = 66.32%

Figure 5.2: Minitab output for one-way repeated measures ANOVA with the Nokia dataset.

The conclusions of this test are only valid if the assumptions of the model are met, i.e., that the residuals exhibit normality and the variances are equal. The equal variance assumption holds according to the rule of thumb from Moore et al. [90] on equal variances (see Table 5.4). Figure 5.3 gives insight as to whether the normality assumption is valid. The histogram of the residuals shows that the shape is about normal although it skews slightly to the right. This indicates that there are some values that are much larger than would be encountered by a population with a normal distribution. The outliers are also reflected in the normal probability plot of Figure 5.3.

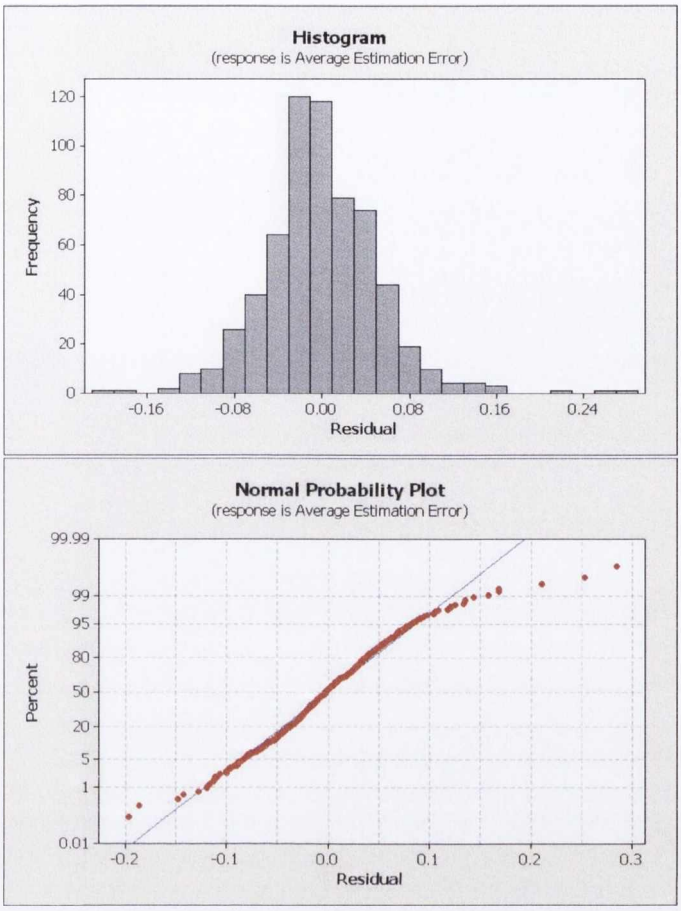

Figure 5.3: Histogram and normal probability plot for the residuals in the Nokia dataset.

The output of the non-parametric Friedman test is given in Figure 5.4. Instead of means of the data, the results use medians. The test statistic is S and it is shown to have a p-value (0.000) that is less than the desired significance level of .05. This indicates that the null hypothesis can be rejected and that the average estimation errors are systematically larger for some alternatives than for others. The remainder of the output shows the estimated median for each alternative which is the total median for all alternatives (i.e., the grand median) plus the effect of the alternative.

Friedman Test: Average Estimation Error versus Alternative
blocked by User

S = 349.59 DF = 14 P = 0.000
S = 349.61 DF = 14 P = 0.000 (adjusted for ties)

Alternative	N	Est	Median	Sum of Ranks
01–CBR–OTHERS	42		0.3578	158.0
02–CBR–HISTORY	42		0.4701	418.0
03–CBR–BOTH	42		0.3662	174.0
04–BAYES–OTHERS	42		0.3663	227.0
05–BAYES–HISTORY	42		0.4758	416.5
06–BAYES–BOTH	42		0.4298	299.5
07–SVM–OTHERS	42		0.3626	171.0
08–SVM–HISTORY	42		0.5413	545.0
09–SVM–BOTH	42		0.3691	185.0
10–BOOST–OTHERS	42		0.3763	205.0
11–BOOST–HISTORY	42		0.4711	427.0
12–BOOST–BOTH	42		0.4225	296.0
13–RANDOM–OTHERS	42		0.4884	476.0
14–RANDOM–HISTORY	42		0.5103	537.0
15–RANDOM–BOTH	42		0.5042	505.0

Grand median = 0.4341

Figure 5.4: Minitab output for the Friedman test with the Nokia dataset.

The conclusions are that the Nokia dataset meets the assumptions for ANOVA and that the ANOVA test is valid. This indicates that there is a difference between the alternatives and that the parametric multiple comparison tests should be performed. Although there are some outliers, the basic shape of the histogram indicates a normal

distribution. In addition, the largest outliers do not show a specific trend for belonging to a specific alternative or user. The fact that there are more than 40 observations also indicates that the data can be considered normal even if there is strong skewness [90]. The Friedman test also rejected the null hypothesis so the non-parametric multiple comparison tests are also applicable. However, less weight will be assigned to the non-parametric test results as there is a more powerful test available, i.e., the parametric Tukey test.

Locomotion Dataset

Figure 5.5 shows the means of the average estimation error for each alternative with the Locomotion dataset. Again, the p-value (0.000) is less than the desired significance level of .05. This indicates that it is safe to reject the null hypothesis and that there is a systematic difference between the alternatives. If the assumptions of the parametric model are verified, this indicates that it is safe to run the parametric multiple comparison test to discover which alternatives perform better than others.

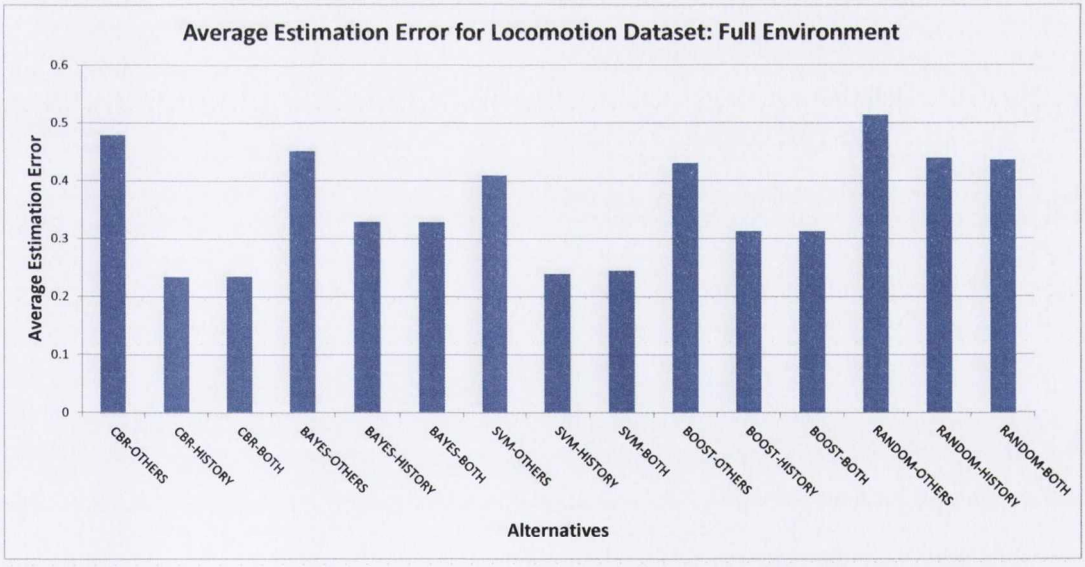

Figure 5.5: Full environment alternative means for the Locomotion dataset.

However, the assumptions for ANOVA are not met for the Locomotion dataset. As already indicated, Table 5.4 shows that the standard deviations of ten of the fifteen alternatives are more than twice the smallest standard deviation in the dataset. This is

due to the SVM-HISTORY alternative having such a low standard deviation. In fact, if the SVM tests are removed, the equal variance assumption holds.

In addition, most of the OTHERS sets have a very high standard deviation relatively. Other than the BAYES tests, the four methods with the OTHERS instance sets by themselves have the highest standard deviation in the dataset. This is because almost all the attributes have to do with tracking a user's motion and almost none are shared environment context. Different users have varying degrees of similarity of motion patterns to other users and this causes the wide spread in average estimation error when using the OTHERS set. The similarity in walking motion for a user is also the main reason the HISTORY set vastly outperforms the OTHERS set.

In addition, Figure 5.6 shows that the distribution is almost normal. This is key, because any deviation from a normal distribution is especially important for the Locomotion dataset as it is a small sample, i.e., there are less than 40 observations per alternative. However, due to the failure of the equal variance assumption, the conclusion is that the Locomotion dataset violates the assumptions of the ANOVA model and that the parametric test should not be relied upon.

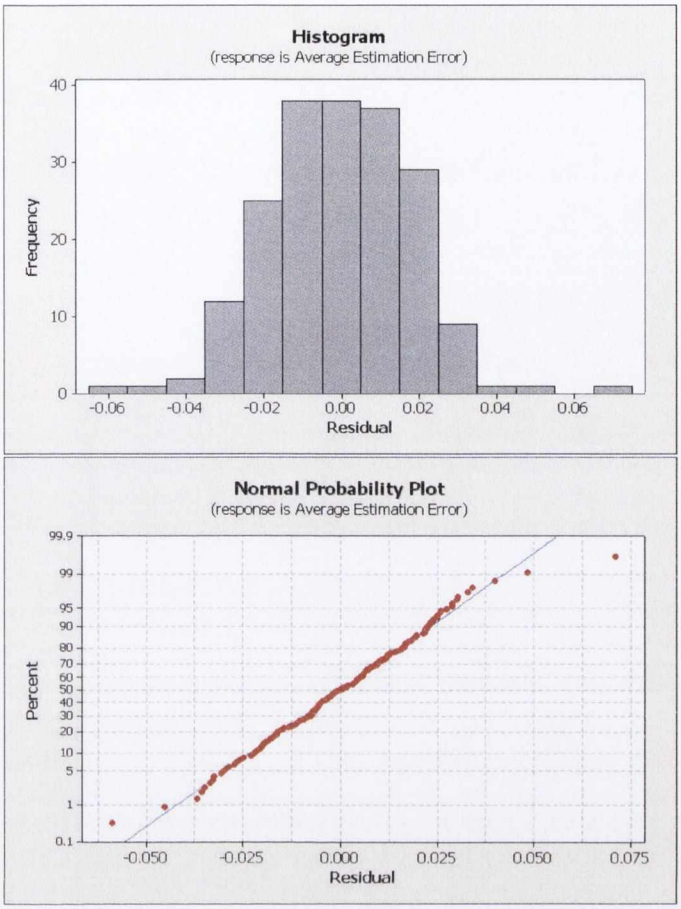

Figure 5.6: Histogram and normal probability plot for the residuals in the Locomotion dataset.

The output of the non-parametric Friedman test has a p-value (0.000) that is less than the desired significance level of .05. This indicates that the null hypothesis can be rejected and that the average estimation errors are systematically larger for some alternatives than for others. The non-parametric multiple comparison tests can also be performed given this result.

Helsinki Dataset

Figure 5.7 shows the means of the average estimation error for each alternative with the Helsinki dataset. As with the other datasets, the p-value (0.000) is less than the desired significance level of .05, and this indicates that it is safe to reject the null hypothesis. That is, there is a systematic difference between the alternatives. If the assumptions of the parametric model are verified, this indicates that it is safe to run

the parametric multiple comparison tests to discover which alternatives perform better than others.

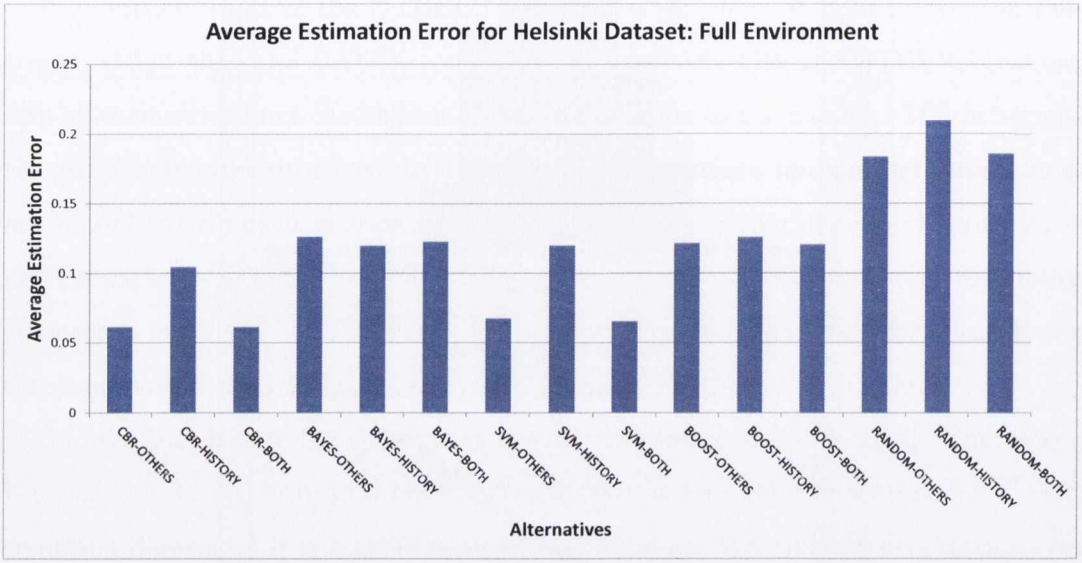

Figure 5.7: Full environment alternative means for the Helsinki dataset.

However, the assumptions for ANOVA are not met for the Helsinki dataset. As already indicated, Table 5.4 shows that the standard deviations of the RANDOM-HISTORY and CBR-HISTORY alternatives are more than twice the smallest standard deviation in the dataset. In fact, the five tests with the HISTORY instance sets by themselves have the highest standard deviation.

In addition, Figure 5.8 shows that the distribution is greatly skewed towards the right. The thirteen largest outliers are mostly from the RANDOM-HISTORY alternative and all involve the HISTORY instance source. This is not unexpected, as Table 5.4 indicates a large standard deviation which is a statistic that outliers can greatly affect. Removing these outliers brings the standard deviations to within an acceptable range to assume equal variance. The normal probability plot still contains some anomalous readings, however, toward the tails of the plot. Removing the thirteen outliers seems to affect the data enough so that it meets the assumptions required of ANOVA, especially as deviations from normality are not as important when the sample size is over 40 (in this case, it is 240). However, removing the outliers and particularly the outliers of such a great number from the same instance source would not be a true reflection of the environment. The conclusion is that the Helsinki dataset violates the assumptions

and that the parametric test should not be relied upon.

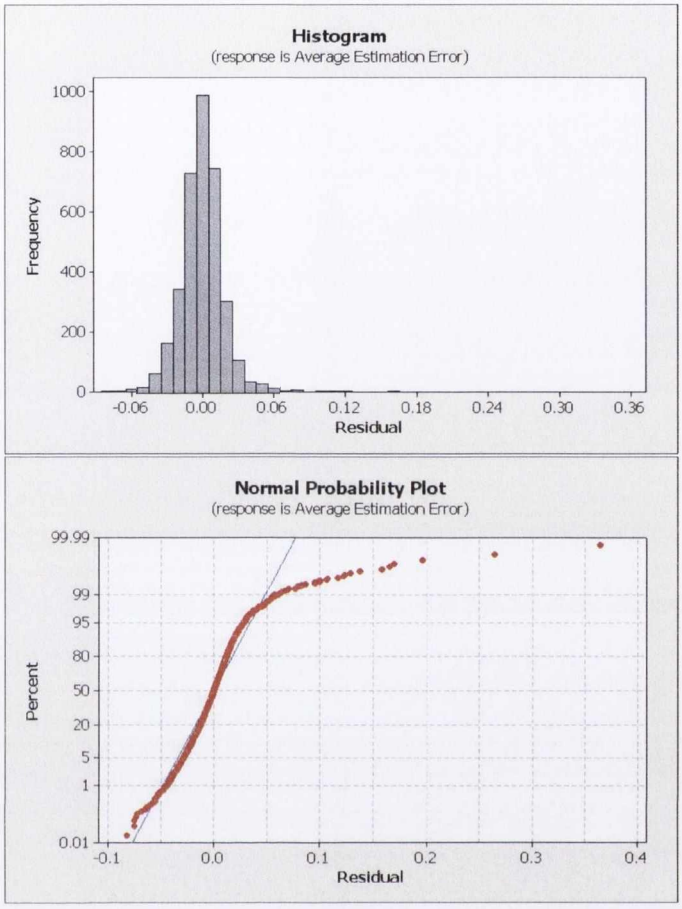

Figure 5.8: Histogram and normal probability plot for the residuals in the Helsinki dataset.

After adding the outliers back into the dataset, the non-parametric Friedman test has a p-value (0.000) which is less than the desired significance level (.05). This indicates that the null hypothesis can be rejected and that the average estimation errors are systematically larger for some alternatives than for others. The non-parametric multiple comparison tests can be performed in light of this result.

5.1.2 Multiple Comparisons Overview

As the ANOVA and Friedman tests all showed a significant difference between at least a subset of the alternatives, the multiple comparison tests can be performed. The specific research questions will be addressed by the results of these tests.

Both the parametric and non-parametric tests will be performed for each dataset as suggested in the work of Juristo and Moreno [61]. However, the equal variance assumptions did not hold for the Locomotion and Helsinki datasets, so the parametric tests will exert much less influence than the corresponding non-parametric ones for these datasets. The analysis of the Nokia dataset uses the parametric Tukey test as the data did not violate the parametric test's assumptions.

The Minitab grouping output for the parametric Tukey test of the Nokia dataset is shown in Figure 5.9. Any alternatives that are significantly different belong to different groupings, i.e., the letters to the right of the output. The non-parametric test is the Wilcoxon signed-rank test, and the Minitab output for this test is shown Figure 5.10. Since this is a pairwise test and there are sixteen comparisons, the Bonferonni correction for the significance level is used. The significance level is .003125 and this is reflected in the output as "99.7% Achieved Confidence". If the confidence intervals overlap in this test then the test reports no statistically significant difference between the alternatives being compared.

Grouping Information Using Tukey Method and 95.0% Confidence

Alternative	N	Mean	Grouping
08–SVM–HISTORY	42	0.53	A
14–RANDOM–HISTORY	42	0.51	A B
15–RANDOM–BOTH	42	0.51	A B C
13–RANDOM–OTHERS	42	0.50	A B C
05–BAYES–HISTORY	42	0.48	B C
11–BOOST–HISTORY	42	0.48	B C
02–CBR–HISTORY	42	0.47	C D
06–BAYES–BOTH	42	0.43	D E
12–BOOST–BOTH	42	0.43	E
04–BAYES–OTHERS	42	0.38	F
10–BOOST–OTHERS	42	0.38	F
07–SVM–OTHERS	42	0.37	F
01–CBR–OTHERS	42	0.37	F
09–SVM–BOTH	42	0.37	F
03–CBR–BOTH	42	0.37	F

Means that do not share a letter are significantly different.

Figure 5.9: Tukey test output for the Nokia dataset.

Wilcoxon Signed Rank CI

	Estimated Median	Achieved Confidence	Confidence Interval Lower	Upper
Avg Est Err_01–CBR–OTHERS	0.3641	99.7	0.3360	0.3947
Avg Est Err_02–CBR–HISTORY	0.4763	99.7	0.4311	0.5178
Avg Est Err_03–CBR–BOTH	0.3699	99.7	0.3362	0.3987
Avg Est Err_04–BAYES–OTHERS	0.381	99.7	0.328	0.437
Avg Est Err_05–BAYES–HISTORY	0.4831	99.7	0.4477	0.5163
Avg Est Err_06–BAYES–BOTH	0.4385	99.7	0.4007	0.4704
Avg Est Err_07–SVM–OTHERS	0.3704	99.7	0.3324	0.4093
Avg Est Err_08–SVM–HISTORY	0.5430	99.7	0.5043	0.5705
Avg Est Err_09–SVM–BOTH	0.3726	99.7	0.3385	0.4051
Avg Est Err_10–BOOST–OTHERS	0.3777	99.7	0.3423	0.4109
Avg Est Err_11–BOOST–HISTORY	0.4761	99.7	0.4341	0.5170
Avg Est Err_12–BOOST–BOTH	0.4241	99.7	0.3811	0.4666
Avg Est Err_13–RANDOM–OTHERS	0.4947	99.7	0.4636	0.5273
Avg Est Err_14–RANDOM–HISTORY	0.5200	99.7	0.4898	0.5465
Avg Est Err_15–RANDOM–BOTH	0.5140	99.7	0.4844	0.5394

Figure 5.10: Minitab output for the Wilcoxon signed-rank test with the Nokia dataset.

A summary of the comparisons needed to answer the research questions and the results of the comparisons for both parametric and non-parametric multiple comparison tests are given in Table 5.5. If the difference between two alternatives is not significantly different than zero, a dash is inserted into the table. If there is a statistically significant difference, a greater than or less than symbol is used. For example, as the CBR-OTHERS alternative has less average estimation error than the RANDOM-OTHERS alternative, the symbol "<" is used for their comparison. This is also interpreted as the CBR-OTHERS alternative exhibits more overall accuracy than RANDOM-OTHERS. If there is a difference between the results of the two tests, both are given with the parametric test result listed first. If there is no difference between the parametric and

the non-parametric tests then only the single result is displayed.

Specific Comparisons	Nokia	Locomotion	Helsinki
RQ-2: Comparing OTHERS			
CBR vs. RANDOM	<	<,-	<
BAYES vs. RANDOM	<	<	<
SVM vs. RANDOM	<	<	<
BOOST vs. RANDOM	<	<	<
RQ-3: OTHERS vs. HISTORY			
CBR	<	>	<
BAYES	<	>	-
SVM	<	>	<
BOOST	<	>	-
RQ-4: OTHERS vs. BOTH			
CBR	-	>	-
BAYES	<,-	>	-
SVM	-	>	-
BOOST	<,-	>	-
RQ-5: BOTH vs. HISTORY			
CBR	<	-	<
BAYES	<,-	-	-
SVM	<	-	<
BOOST	<	-	-

Table 5.5: Multiple comparison overview. If the parametric and non-parametric tests disagree, both results are given with the parametric test result listed first.

The overview in Table 5.5 shows that there are four differences between the results of the parametric and the non-parametric tests. These differences always represent a significant difference recognized in the parametric test and no significant difference with the non-parametric test. This reflects the less powerful nature of the non-parametric tests. The following four sections use the information in this table to address their corresponding research questions. The parametric tests in the Nokia dataset are fa-

vored for three of these four differences between statistical tests as the dataset was determined to meet the parametric test assumptions. The remaining difference (CBR-OTHERS versus CBR-HISTORY for the Locomotion dataset) uses the non-parametric tests. It is also informative to note for the conclusions of the next few sections that SVM and CBR have the same results in the table (except for RQ-2 for the Locomotion dataset) and that BOOST and BAYES have the same results (except for RQ-5 for the Nokia dataset).

Table 5.5 shows that the differences in most tests are statistically significant. However, a statistically significant difference does not mean that the difference is meaningful. For example, a one percent improvement in prediction error for the "user activity" attribute in the Nokia dataset for two alternatives might be statistically significant, but it is not necessarily useful to the user's application or even worth any of the extra work that the improvement provides.

The definition of "meaningful difference" is attribute- and application-specific, but the improvements in the evaluations of this thesis can be put into some context for both nominal and numerical attributes. The difference of the average estimation error in the first comparison (CBR-OTHERS versus RANDOM-OTHERS) for the Nokia dataset is a little bit larger than .1. An improvement of .1 for a nominal attribute like user activity would be a decrease of 10% for prediction error. For a continuous attribute like temperature, the average difference in error is about .136 units or about .68 degrees Celsius after converting back from the standardization in Gelman [47] (where the temperature groupings are mostly 5 degrees and the units have a standard deviation of .681). Depending on the application, these differences could be quite meaningful.

5.2 RQ-2: Situational Relevance

> Can machine learning methods using situational relevance to acquire context information from other users perform with more overall accuracy than existing context selection methods that do not employ situational relevance?

The second research question determines the utility of using learning methods with

situational relevance to select from other users versus the current existing selection methods which do not. As noted in Chapter 2, existing context selection methods in pervasive applications [57, 111, 27] do not select their values using any relevance beyond timeliness. Given that the other users' instances in the OTHERS set all have the same timestamp, this is equivalent to random selection.

The proximity-based methods [68, 58] of Section 2.2 are a slight improvement on context selection methods, but only for context information that is relevant based on proximity. Much of this information requires location information as the ability to communicate does not always indicate proximity, e.g., communication over the Internet. This requirement is not met as two of the datasets do not contain any location information and the third (i.e., the Nokia dataset) requires extra knowledge regarding the proximity of cell towers and area codes.

The use of learning methods with situational relevance extends the limited relevance of context selection and proximity-based methods by including similarity considerations for all of the recorded attributes in a situation. The differences in the evaluation between randomly selecting from the set of other users (i.e., the current approach of existing context selection methods) and using learning methods with more extensive situational relevance to select other users' information is statistically significant for both the parametric and non-parametric tests. The only exception is the non-parametric test for the Locomotion dataset. All other approaches exhibit more overall accuracy than random selection across all datasets (see Table 5.6).

Comparisons	Nokia	Locomotion	Helsinki
CBR-OTHERS vs. RANDOM-OTHERS	$<$	$<$,-	$<$
BAYES-OTHERS vs. RANDOM-OTHERS	$<$	$<$	$<$
SVM-OTHERS vs. RANDOM-OTHERS	$<$	$<$	$<$
BOOST-OTHERS vs. RANDOM-OTHERS	$<$	$<$	$<$

Table 5.6: Comparisons for RQ-2.

5.3 RQ-3: Previous Experience Versus Other Users' Information

Does context acquisition using machine learning methods trained only from the information of other users perform with more overall accuracy than the existing approach of training them from previous experience?

Comparisons	Nokia	Locomotion	Helsinki
CBR-OTHERS vs. CBR-HISTORY	<	>	<
BAYES-OTHERS vs. BAYES-HISTORY	<	>	-
SVM-OTHERS vs. SVM-HISTORY	<	>	<
BOOST-OTHERS vs. BOOST-HISTORY	<	>	-

Table 5.7: Comparisons for RQ-3.

As seen in Section 5.2, learning methods that use situational relevance improve the overall accuracy of context acquisition from other users over existing context selection methods. The utility of using other users' information is explored by the rest of the research questions. RQ-3 asks whether using learning methods trained on the set of other users' information by itself is better than the existing approach of using learning methods trained on previous experience. The hypothesis is that training with the set of other users' information (OTHERS) will show an improvement over the previous experience (HISTORY) set because the former set more accurately reflects the current situation. That is, the current situation could be an unfamiliar situation or concept drift could have been experienced since the previous experience was recorded. This improvement is expected in spite of the fact that it is assumed that the information describes other users in the environment and not the application's user.

The answer to RQ-3 is a "no" although it is not as definitive as it was for the answer to RQ-2. The best overall accuracy between the HISTORY and OTHERS instance sets

is not consistent across the different datasets. This means that the hypothesis that the OTHERS instance set is *always* more accurate to train on than HISTORY does not have a great deal of evidence to support it.

For the CBR and SVM approaches, the OTHERS set for the Nokia and Helsinki datasets is more accurate, while for the Locomotion dataset it is less accurate. For the Locomotion dataset, it is important to note that the HISTORY and BOTH sets learn on a maximum instance size of 1000. This does not seem to affect the results as these sets are already much better than OTHERS, even without these extra instances.

For the naive Bayes classifier and the boosting algorithm, the difference between the OTHERS and HISTORY instance sets is the same as the CBR and SVM approaches for the Nokia and Locomotion datasets. For the Helsinki dataset, the difference for these two approaches is not significant. The conclusion is that there is sometimes a difference between OTHERS and HISTORY instance sets but that the better instance set on which to train is dataset-specific. The consequences of these results to instance set selection will be discussed in Section 5.5.

As a final note, to explore whether this difference is an artifact of the difference in size for the two sets, it is interesting to compare the two sets when they are of equal size. This analysis is done in Section 5.6 with the result being that instance set size does not greatly bias one set over the other when run in a full environment.

5.4 RQ-4: Combining Instance Sources

> Is there an improvement in overall accuracy when utilizing the previous experience of the user in addition to the context information of other users?

The evaluation of RQ-3 in the previous section showed that there is often a statistically significant difference between the set of other users and the previous experience. This section answers RQ-4 which asks whether training on the union of the two sets improves on the performance of using other users' information by itself. The related topics of whether including other users' information improves the overall accuracy of the previous experience as well as a discussion of the instance set selection knowledge requirements of pervasive applications are addressed by the evaluation of RQ-5 in Section 5.5.

Comparisons	Nokia	Locomotion	Helsinki
CBR-OTHERS vs. CBR-BOTH	-	>	-
BAYES-OTHERS vs. BAYES-BOTH	<,-	>	-
SVM-OTHERS vs. SVM-BOTH	-	>	-
BOOST-OTHERS vs. BOOST-BOTH	<-	>	-

Table 5.8: Comparisons for RQ-4.

Table 5.8 shows that the answer to RQ-4 is that overall there seems to be no statistically significant improvement when adding the HISTORY set to the OTHERS set. The Locomotion dataset is an exception to this, however. For the Locomotion dataset, there is an improvement in overall accuracy for all alternatives when adding the HISTORY set. This is expected as the overall accuracy of HISTORY was shown to be significantly better than OTHERS in Section 5.7.

Besides the Locomotion dataset, there is no improvement in overall accuracy when adding previous experience to the set of other users' information. In the parametric BAYES and BOOST comparisons for the Nokia dataset, introduction of the HISTORY set actually makes the overall accuracy worse. This would seem to be a product of introducing an unfavorable instance set (HISTORY) into the more favorable instance set (OTHERS). Interestingly, the CBR and SVM approaches do not exhibit this decrease in accuracy upon introduction of an unfavorable source, and this observation will be explored further in the next section.

5.5 RQ-5: Instance Set Selection

What is the approach for instance set selection in pervasive environments that exhibits the most overall accuracy and how knowledge autonomous is this approach?

Comparisons	Nokia	Locomotion	Helsinki
CBR-BOTH vs. CBR-HISTORY	<	-	<
BAYES-BOTH vs. BAYES-HISTORY	<,-	-	-
SVM-BOTH vs. SVM-HISTORY	<	-	<
BOOST-BOTH vs. BOOST-HISTORY	<	-	-

Table 5.9: Comparisons for RQ-5.

In the last two sections there were two comparisons of the instance sets within each learning method exhibiting situational relevance, i.e., OTHERS versus HISTORY for RQ-3 and OTHERS versus BOTH for RQ-4. The evaluation of RQ-5 examines the final comparison within each learning method, i.e., the comparison between the HISTORY set and the BOTH set. The analysis will help determine any overall accuracy gained by introducing the set of other users' information into the previous experience set. In addition, since all three instance set comparisons are done for all the learning methods, the instance set selection knowledge needed for them can be assessed, its effect on knowledge autonomy considered, and the final full environment research question (RQ-5) answered.

The answer to whether the OTHERS set helps the overall accuracy of the HISTORY set is that it improves the overall accuracy or has no effect at all for both learning methods over all datasets. Specifically, all of the comparisons involving the Nokia dataset and the CBR and SVM comparisons involving the Helsinki dataset show a statistically significant improvement in overall accuracy. The remainder of comparisons shows no statistically significant differences. This is equivalent to the results of RQ-4's evaluation in that the set with more overall accuracy between HISTORY and OTHERS at the least does not decrease overall accuracy when added to the set with less overall accuracy.

In fact, this latter point can be summarized in order to characterize the knowledge

requirements for instance set selection for the four selected learning methods. For the CBR and SVM learning methods, the results of the comparisons in RQ-4 and RQ-5 lead to the conclusion that adding the set with less overall accuracy to the set with more overall accuracy does not produce a significant difference in overall accuracy. For these two learning methods, the HISTORY set for the Locomotion dataset and the OTHERS set for the Nokia and Helsinki datasets show no statistical difference from the BOTH set. The conclusion is that, no dataset-specific instance set selection knowledge is necessary as long as retrieval and processing concerns are not important, because the BOTH set will not be significantly different than the best set.

For the BAYES and BOOST learning methods, the conclusion is somewhat different and also does not exhibit as high conclusion validity as the one for CBR and SVM. Looking at the trends on the graphs of overall means for each dataset, it appears that the set with the less overall accuracy negatively affects the overall accuracy of the more accurate set when they are combined. For example, Figure 5.1 and 5.7 have a BOTH overall accuracy that appears to be between HISTORY and OTHERS for BAYES and BOOST. Only the Nokia dataset shows that this difference is significant. The size of the OTHERS set in the Locomotion dataset is hypothesized to be too small relative to the HISTORY set to cause any significant negative change in overall accuracy. This indicates that instance set selection knowledge is important for the BAYES and BOOST learning methods, as there is sometimes a statistically significant difference between the set with the most overall accuracy and the BOTH set. That is, to identify the set with the best overall accuracy for the BAYES and BOOST learning methods, knowledge of whether HISTORY or OTHERS is the set with the most overall accuracy is needed.

While the BAYES and BOOST conclusions are not as well supported by the statistical analysis of the datasets as the CBR and SVM conclusions, all of these trends make sense in light of how their respective learning methods behave. It is hypothesized that this behavior is the case for the CBR approach as it chooses the most similar instance regardless of what other instances are put into the set. This means if the most similar instance is always in one set, it will always choose that instance from that set regardless of any other added instances. Similarly, the SVM approach deals with similarity of instances as well [7]. Any instances that could affect the value but are still

within the margin are ignored. They are both also stable algorithms, meaning that a small change in the training set does not produce a large change in the model. For the discretized BAYES and BOOST approaches, any new value affects the probability and learner, respectively, associated with that bin for that attribute. Introducing instances that exhibit less accuracy would affect the overall accuracy. BOOST also uses an unstable algorithm as a base learner, i.e., decision trees. This means a small change in the training set causes a large change in the model. These similarities between learning methods are the main reasons that the results of the CBR and SVM alternatives are similar to each other and the results of the BOOST and BAYES alternatives follow a similar pattern. These trends will be reinforced in the evaluations of the next section.

5.6 RQ-6: Instance Set Sparseness

> What is the approach with the best overall accuracy for context acquisition involving sparse instance sets, i.e., when the previous experience of the user is small or nonexistent or when the environment contains few or zero other users?

The evaluation of the final research question (RQ-6) serves two purposes: to explore which instance sets exhibit the most overall accuracy when confronting sparse instance sets and to further support the conclusions in the evaluations of RQ-2 to RQ-5. The former is achieved by running experiments with smaller subsets than the full environment for both the previous experience and other users sets. The latter is achieved by ensuring that the evaluations in sparse environments are consistent with the conclusions of Section 5.2 to Section 5.5. Specifically, the subsets of previous experience and users in this section's evaluations allow the comparison of different instance sets with an equal number of instances. This removes any bias in the full environment evaluations because of disparities in instance set size. The gradual introduction of instances from one set into another also allows the examination of the effects on the learning methods' overall accuracy when instances from a less or more accurate set are introduced (see the instance set selection conclusions in Section 5.5).

The experiments for sparse instance sets are done by examining charts that inject an increasing number of users (for sparse environment evaluations) or instances of

previous experience (for sparse previous experience evaluations). For example, Figure 5.11 charts the trends in overall accuracy as the number of other users increases from no other users (0) to the full amount of other users (41). Overall accuracy for HISTORY for all methods is shown by the BOTH line at the far left of that line, i.e., when the number of other users in the BOTH set is zero. The opposite is true in the sparse previous experience charts, i.e., the first data point of the BOTH line represents the OTHERS set when the number of HISTORY instances is zero.

Besides taking a subset of each instance source, sparse tests are run exactly as they are for the full environment tests with two exceptions. First, the results of Appendix C.2 show that there is no difference when the start times are randomized and when they are not. In light of these results, the start time randomization has been removed from the sparse tests. This allows the tests to be analyzed with the variations from the different subsets isolated from any other random fluctuations due to random start time. Second, while most of the lines of each alternative are relatively smooth as more instances are added to an alternative, there could still be variation due to the membership of the specific random sample. To combat this, any subset with less than fifty instances are resampled five times in order to counteract any random sampling bias. After examining the lines with and without resampling, most of the alternatives follow roughly the same trajectory. This leads to the conclusion that any non-smooth fluctuations seem to be a product of the learning method for a particular number of sources rather than a product of random sampling bias, particularly when there are just a few instances.

Two other notes about the charts will aid in their interpretation. First, when there are no instance sets for an alternative, the average of the total range for that attribute in the dataset is used because an attribute's range is already information assumed to be available for the learning methods (see Section 3.4). This causes the average estimation error values to skyrocket but requires no extra knowledge. The values are equal for the datasets across all methods and tests, i.e., an average estimation error of .689 for the Nokia dataset, 7.01 for the Locomotion dataset, and 4.57 for the Helsinki dataset. These values are clipped from the graph window in order to more clearly present the differences between alternatives which include instances for all alternatives. Second, when an instance set has a large number of instances, two graphs are presented. One

presents sequential subsets from zero up to a small number of instances, e.g., the Helsinki dataset's initial sparse environment in Figure 5.13, and the other shows the overview of the full range, e.g., the Helsinki dataset overview in Figure 5.14.

The results of this section will be mostly comments about trends as the instance sets become less sparse and will use very little statistical analysis. As most of the interesting observations come from the manner in which one set of the full environment gradually becomes another set of the full environment, e.g., the HISTORY set becomes the BOTH set of the full environment through the gradual introduction of instances from the OTHERS set, the necessary statistical analysis was already completed for the evaluations in the previous sections.

5.6.1 Sparse Environments

Figure 5.11 shows the Nokia dataset's performance in sparse environments. When there are no other users, utilizing the HISTORY (i.e., BOTH when there are zero other users) is a much better option. After that, the OTHERS sets improve dramatically. BAYES-OTHERS and BOOST-OTHERS roughly equal their BOTH equivalents when there are four other users and then the OTHERS sets outperform the BOTH sets as more than four other users are introduced. CBR-OTHERS and SVM-OTHERS eventually converge with their BOTH equivalents, but the BOTH sets are more accurate in very sparse environments. For all learning methods, the HISTORY sets are outperformed fairly quickly (all within four instances added from other users), and the difference becomes statistically significant as the full environment is reached (as shown in the analysis for RQ-3's evaluation).

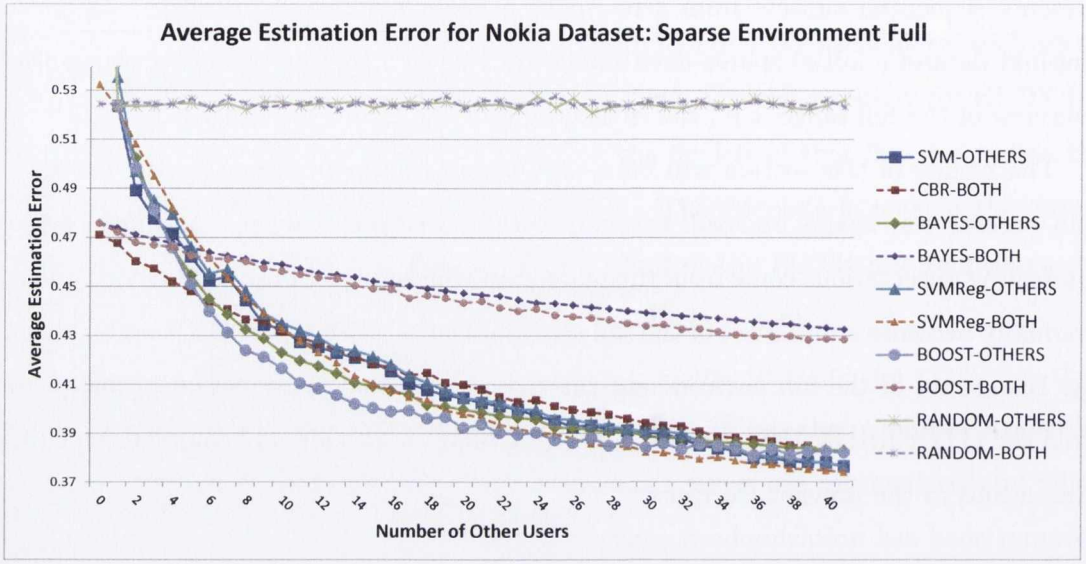

Figure 5.11: Sparse environment alternative means for the Nokia dataset.

Environment sparseness for the Locomotion dataset is shown in Figure 5.12. There is brief improvement in the OTHERS sets as more users are introduced, but it is clear that HISTORY and BOTH are never surpassed. This is demonstrated in the full environment tests in RQ-3 and RQ-4.

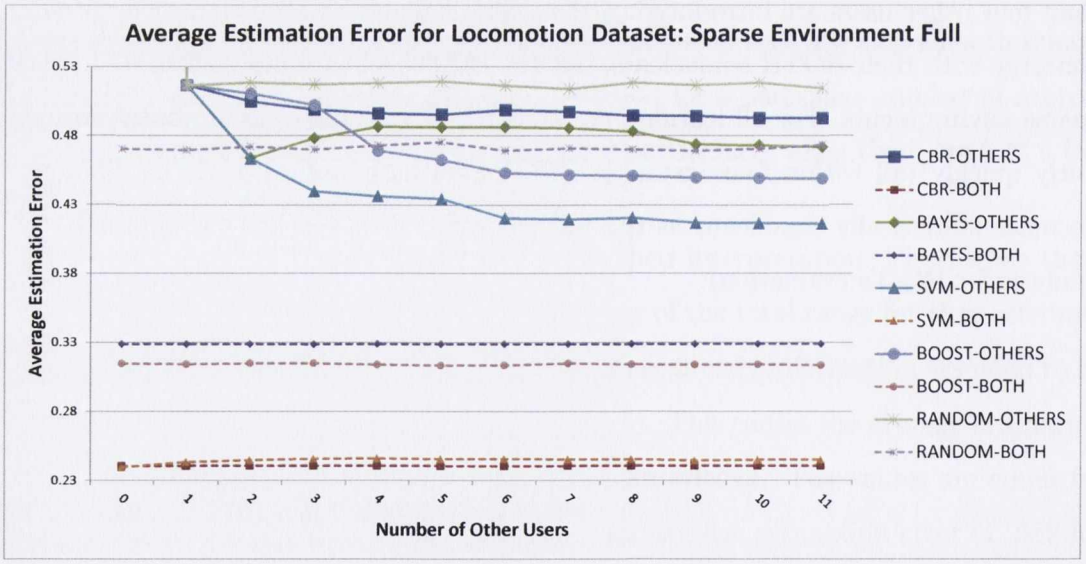

Figure 5.12: Sparse environment alternative means for the Locomotion dataset.

For the Helsinki dataset, Figure 5.13 shows the sparseness for 0 to 11 other users and Figure 5.14 presents the overview for all 239 other users. The CBR and SVM

approaches perform similarly to those in the Nokia dataset. The BAYES and BOOST approaches differ from the trends of the Nokia dataset in that while OTHERS decreases in error, the error of the BOTH sets, after an initial decrease, trends upwards as more other users are added.

It is also interesting to note the alternatives when there are 23 other users as this is approximately equal to the number of previous experience instances (21) in the non-sparse tests. The error means are smaller for the OTHERS than for the HISTORY sets for CBR, SVM, and BOOST. This trend is opposite for BAYES. Although the differences are small in all cases (i.e., less than .015), this seems to indicate that OTHERS is the better set for CBR, SVM, and BOOST and HISTORY is the better set for BAYES given equal set sizes.

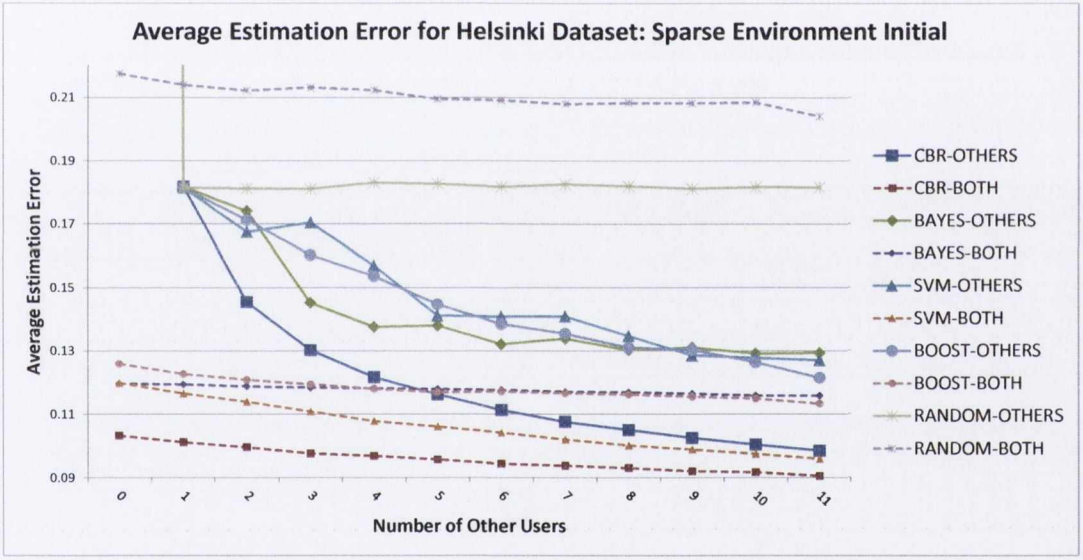

Figure 5.13: Initial sparse environment alternative means for the Helsinki dataset.

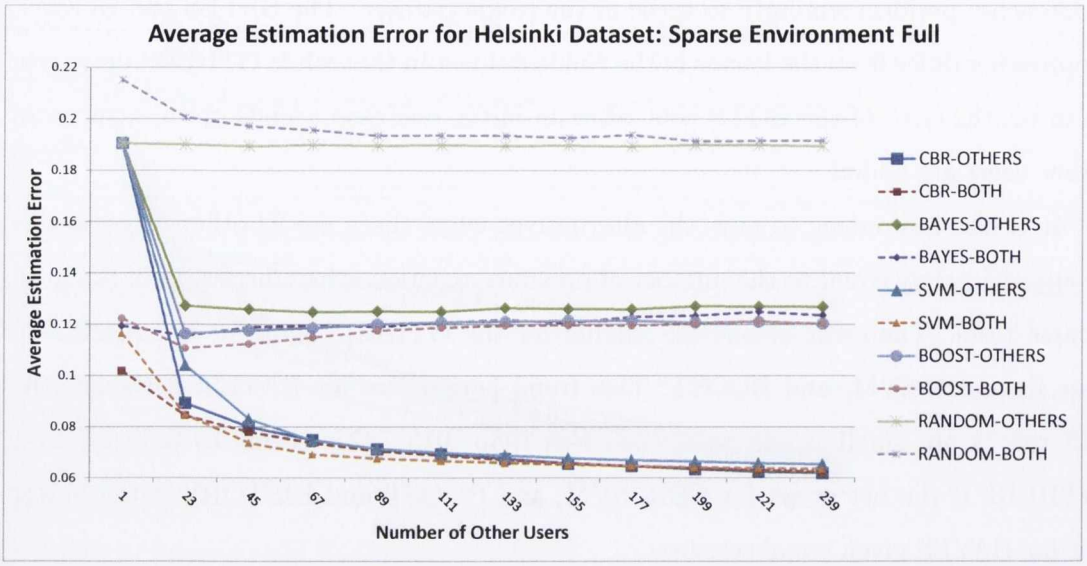

Figure 5.14: Overview of sparse environment alternative means for the Helsinki dataset.

5.6.2 Sparse Previous Experience

The charts in this subsection are very similar to the ones in the preceding subsection, except that previous experience instances are introduced into the full environment OTHERS set instead of other users being introduced into the full environment HISTORY set. This raises a problem for the CBR approaches, as the set that the weights are trained on can potentially be empty. In practice, a viable alternative would be to learn the weights from other users' information. However, in order to avoid any weight learning biases and to focus only on changes due to previous experience instance set size, weight learning with Stahl is not performed and all attributes are given equal weights instead. Tests run in Appendix B show that the differences between using Stahl and using equal weights is minimal when compared to the differences between alternatives.

Overall accuracy with sparse previous experience for the Nokia dataset is shown in Figure 5.15 for the initial instances and Figure 5.16 for the overview. The graphs show trends consistent with the conclusions already drawn, with the exception of the SVM learning method (discussed in the subsequent paragraph). BOTH remains the best approach for all four learning methods and the overall accuracy of the BOTH set

for BAYES, BOOST, and SVM becomes worse as more of the HISTORY instances are added.

SVM becoming less accurate as more HISTORY instances are added is a product of the alternative's poor performance for the numeric attributes that show little variety in the history training set, i.e., temperature, pressure, and user activity. In the sparse history tests, subsets of these attributes often consist of a single uniform value and in this case that uniform value is returned as the output. The average error for the uniform value output (i.e., without invoking SVM) for these three attribute types is less than when SVM is invoked, i.e., when there is more than one value of the output type. Since a single uniform value is more likely to be encountered in smaller subsets, the average error of smaller subsets is less than that for larger subsets. In fact, when the SVM approach is replaced by one that takes the mode of the values in the instance set, the overall average error is less and does not increase as more history instances are added.

When there are 41 previous history instances, the size of the HISTORY and OTHERS sets are equal. In this case, the trend is that OTHERS is substantially more accurate than HISTORY. This would seem to indicate that given equal set sizes OTHERS is favored over HISTORY for both approaches in the Nokia dataset.

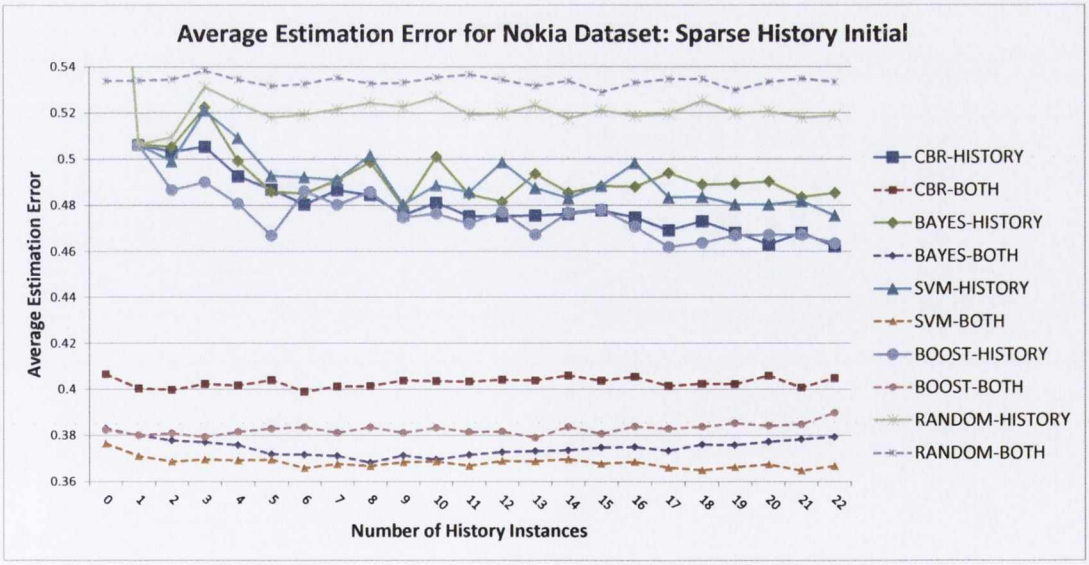

Figure 5.15: Initial sparse history alternative means for the Nokia dataset.

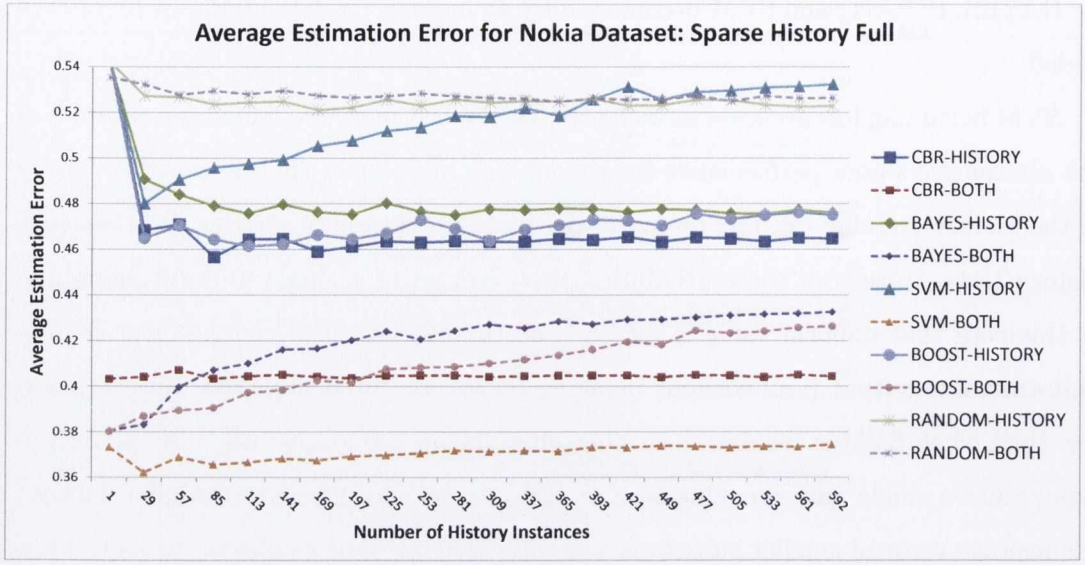

Figure 5.16: Overview sparse history alternative means for the Nokia dataset.

Figure 5.17 and Figure 5.18 show the sparse previous experience results for the Locomotion dataset. The initial HISTORY sets quickly catch up with BOTH and surpass OTHERS when two instances are introduced for SVM-HISTORY and by one instance for the rest of the learning methods. This demonstrates that it is not just instance set size that favors the HISTORY set over the OTHERS set, i.e., two previous experience instances are more accurate than information from twelve other users. As in the full environment tests, SVM and BOOST have a maximum instance count of 1000, and any point beyond 1000 instances for these learning methods is the value for 1000 instances (see Section 3.4).

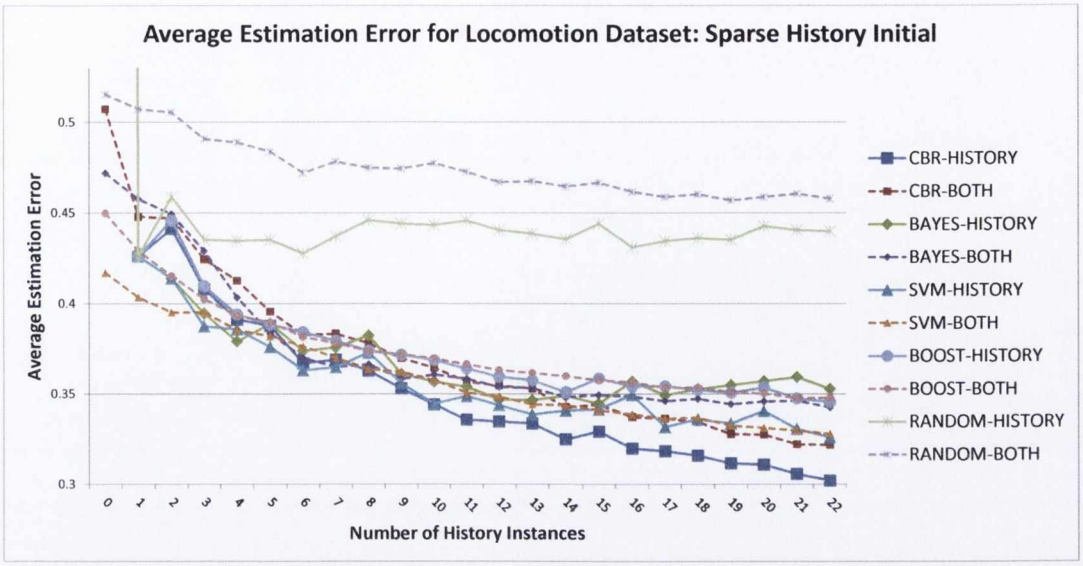

Figure 5.17: Initial sparse history alternative means for the Locomotion dataset.

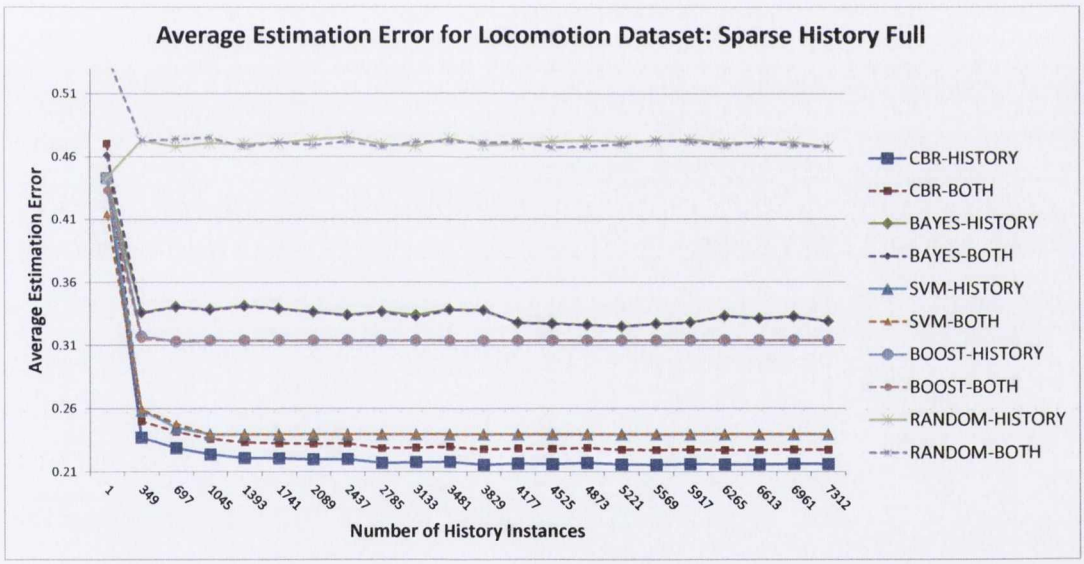

Figure 5.18: Overview sparse history alternative means for the Locomotion dataset.

The final chart of Figure 5.19 depicts the overview of sparse previous experience for the Helsinki dataset. The HISTORY set shows dramatic improvement as more instances of previous experience are added while the BOTH set shows little change.

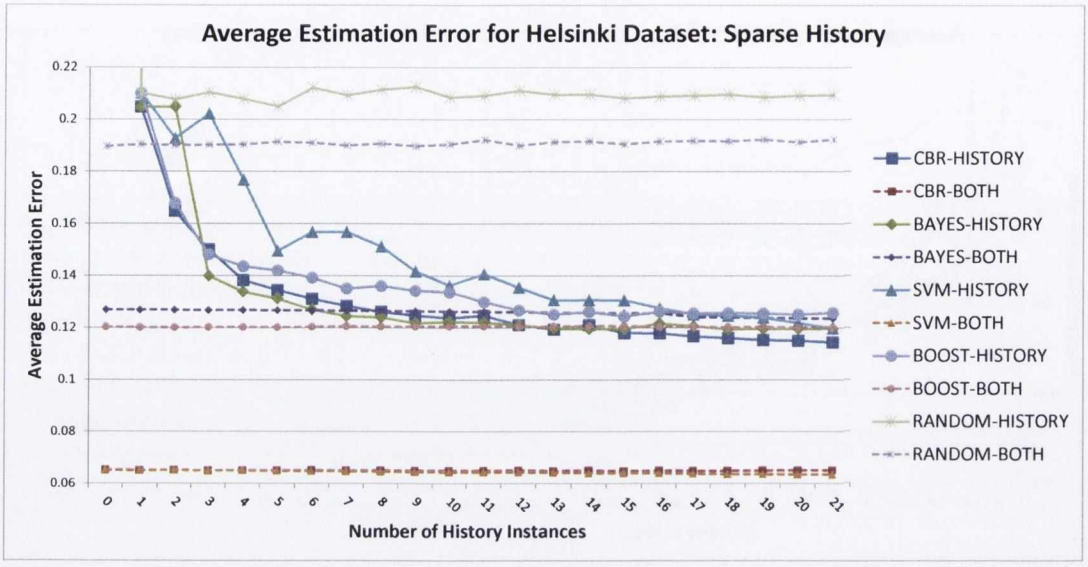

Figure 5.19: Sparse history alternative means for the Helsinki dataset.

5.6.3 Sparse Instance Set Summary

For all learning methods, the answer to the research question for the initially sparse OTHERS and HISTORY sets is to favor the BOTH set (especially when the sparse set is empty). For the CBR and SVM approaches, this is shown to be a good strategy up to the full instance set which reinforces the instance set selection conclusions about the CBR and SVM approaches for the RQ-5 results in Section 5.5.

For the BAYES and BOOST approaches, the answer to the research question is to favor the BOTH set initially, and then to choose the source set that performs better with the full instance set once the sparse set contains several instances. As the number of instances in the sparse set increases, the set with the better overall accuracy for the BAYES and BOOST approaches trends toward the HISTORY set in the Locomotion dataset (although the difference is not significant between HISTORY and BOTH) and the OTHERS set for the Nokia dataset. For the Helsinki dataset the OTHERS set is better for BOOST and the HISTORY set is better for BAYES (although neither difference is significant). Although the difference is not significant in the Helsinki dataset, in both the Nokia and Helsinki datasets it appears that the introduction of the worse set harms the accuracy of the union set once several instances have been introduced into the sparse set. The Locomotion dataset does not appear to have

this trend, most likely because of the relatively small size of the OTHERS set when compared to the HISTORY set. These conclusions reinforce those about the BAYES and BOOST approaches in RQ-5.

5.7 Dataset Validity Analysis

In Section 3.2.4.1, three threats to the validity of the single-user conversion procedure were identified and their severity was hypothesized for the research questions in this thesis. The threats are shifting the time of the runs (Threat 1), users performing the same scenarios (Threat 2), and treating the runs of the same user as runs of a different user (Threat 3). This section discusses the severity of the threats in light of the results of the evaluations of this thesis and the experiments carried out in the appendix (Section 5.7.1). Next, conclusions are drawn about the appropriateness of applying the procedure to the three datasets of this thesis and to datasets in general (Section 5.7.2). Finally, the section ends with a summary of the validity implications for each research question and a discussion of the validity of the alternatives to carrying out the study of this thesis without using the single-user conversion procedure (Section 5.7.3).

5.7.1 Validity for the Research Questions

This section analyzes the validity of the results of the research questions in light of the threats of the single-user conversion procedure. The focus of this section is on the OTHERS vs. HISTORY research question (RQ-3), as this is the research question that best epitomizes the bias of the OTHERS set. The validity of the RQ-3, RQ-4, RQ-5, and RQ-6 results are all affected by the same biases while RQ-2 has the same bias for both alternatives. The Locomotion dataset will not be discussed until the end of this section as its results are not greatly affected by the shifting the time of the runs bias (Threat 1) or the same user bias (Threat 3).

The majority of attributes in the Nokia and Helsinki datasets are shared-environment attributes. When the non-shared environment attributes are removed in the experiments in Appendix C.5, the average estimation error is far from zero. All the runs take place in the same environment, so this should not be the case. This shows that there is an artificial disparity between the runs for the shared environment at-

tributes and that the bias against the OTHERS set by shifting the time of the runs (Threat 1) is significant. This was viewed as a worst-case for the OTHERS set, and might be significant enough to counteract the potential bias towards the OTHERS set from the other two threats. Despite this worst-case, the four learning methods of the Nokia and Helsinki datasets still show that the set of other users is more accurate or shows no statistical difference than the previous experience on average. Without this bias, the shared environment attributes would be more similar and the accuracy with the set of other users (OTHERS) would improve even more over previous experience (HISTORY) as there is no bias in the HISTORY set.

The results in Appendix C.5 also show that the non-shared-environment attributes have little effect on the conclusions drawn for the main thesis evaluation. When these attributes are removed from the evaluations, the conclusions drawn for the Nokia dataset remain unchanged. For the Helsinki dataset, the OTHERS vs. HISTORY test for BAYES and BOOST go from no statistical difference to favoring HISTORY over OTHERS. It is posited in Appendix C.5 that this is the result of the UserAction attributes being closely tied to the specific task being performed at that specific time in the scenario, rather than those attributes being personal to a specific user. For the Helsinki dataset, the most convincing argument that the attributes are not personal as much as they are tied to the specific scenario is contained within the results of the clustering tests (Appendix C.6). For the Nokia dataset, which has only one user, the results of the clustering tests are still promising because removing the most similar runs does not change any of this chapter's conclusions. For the Helsinki dataset, which has two indiscernible users, if there is any bias from treating the runs of the same user as runs of different users, then this would manifest itself as a significant difference between the two different users performing the same scenario. If this difference between runs of the two users exists, it was hypothesized that any clusters formed within a single scenario would be most likely to have membership of a single user as the runs would be divided into clusters based on the differences between the two different users. However, there was not a significant difference in results between the clustering test that removes runs from the same cluster and the full environment test. This leads to the conclusion that either there is not a significant difference between runs of different users (and the runs were not clustered by user) or that there was a significant difference and the clustering

removed runs of the same user from the tests. Either way, based on the fact that there is not a significant difference between the results of the full environment tests and the clustering tests, any bias from using runs from the same user is seen to be minimal relative to the difference between the alternatives.

In Section 3.2.4.1, it was hypothesized that the tasks and attributes that make up the selected datasets and the single snapshot nature of the evaluations decrease the bias of the set of other users due to treating runs from the same user as runs from different users. Particularly, the snapshot nature of the evaluations meant that the identities of the users were not remembered from timestamp to timestamp. At any particular time a current user from any of the datasets would find it common on a walk to work or around an office building that there were other nearby users experiencing the same shared environment context information that the current user is experiencing. In addition, the above discussion of the results of removing the most similar runs (Appendix C.6) and removing the non-shared-environment attributes (Appendix C.5), further minimize the bias to the OTHERS set from having runs of the same user.

However, there is still a potential bias from having runs of the same scenario in the OTHERS set (Threat 2) for all three datasets. This thesis deals with this bias towards the OTHERS set in three ways in Section 3.2.4.1. First, randomization was introduced into the single-user conversion procedure to ensure that the scenarios did not line up directly in time. Second, even when the runs do all start at the same time and a run is compared to its most similar run, the experiments in Appendix C.3 show that the average difference between runs is significant. Third, the inclusion of runs from different scenarios was hypothesized to make the collection more realistic.

For the Nokia dataset, the addition of runs that perform other scenarios had no significant effect on accuracy (see Appendix C.4). For the Locomotion dataset, a similar pattern was observed except that it was the runs of other users that had no significant effect on accuracy. For the Helsinki dataset, the CBR learning method showed the same result but the remaining learning methods did not. In addition to the results of RQ-5, where the BOTH set was seen to have no significant difference from the better set, it is concluded that CBR (and SVM most of the time) will always predict a value that is close to the one of the best set and the best scenario.

At least for the CBR alternative, for all three datasets it can be concluded that

having only runs of the same scenario (Threat 2) does not affect validity if the same number of runs of the same scenario is likely to be a subset of a larger, more realistic population. Regardless of which other scenarios are included or the presence of other less accurate runs (e.g., the less accurate of the HISTORY or OTHERS sets), CBR will find an equivalent to the best scenario and run available. This fact affects the generality of the results as a set with runs of the same scenario in synchronization is expected to return the same result as that set augmented with many different scenarios and runs that do not synchronize well with the current user. As a final observation, this helps explain why the randomization of the start time has little effect on accuracy, i.e., it only takes one similar run to approximately be in synchronicity to improve the accuracy.

5.7.2 Applicability of the Single-User Conversion Procedure

The overall conclusion to the discussion in the last section is that the single-user conversion procedure is applicable to the datasets used and the research questions being asked in this thesis. The procedure seems more appropriate for scenarios involving groups and attributes that are less personal and sense shared-environment phenomena. This makes the OTHERS set a worst-case scenario and any alternatives that favor that set do so despite this bias. It also minimizes the bias from treating runs from the same user as runs from different users. This is difficult without severely biasing the results when static or highly personal attributes exist. For example, the Locomotion dataset consists of highly personalized attributes (precisely tracking a user's gait), and the application of the procedure would not have been possible without there being discernible users that could be removed to eliminate this bias.

Additionally, using the runs of the same user would not have been as valid if the thesis did not employ a single timestamp, snapshot evaluation. If the instances from another user were considered in a window made up of many consecutive instances, user-specific patterns would be more likely to be detected and the same user runs would dominate the most influential instances. Despite the minimization of the same user runs bias given by the snapshot nature of the evaluations and the shared-environment sensors, datasets with discernible users should be favored as this maximizes the validity of the single-user conversion procedure.

5.7.3 Validity Summary

The validity of the single-user conversion procedure was discussed in light of the results of the thesis evaluations and appendix experiments. RQ-1 depends on the validity of the single-user conversion procedure itself, and the applicability of the procedure was discussed in Section 5.7.2. RQ-2 has the same OTHERS biases for both RANDOM-OTHERS and learning method-OTHERS alternatives. The biases of RQ-3 to RQ-6 are all minimized according to the discussion in Section 5.7.1 as each research question has an element of the comparison between OTHERS and HISTORY. Specifically, the conclusions of RQ-4 and RQ-5 are true whether a bias exists or not. The conclusions in RQ-6 are all true to some extent, as the very sparse sets are much worse than the BOTH set relative to any potential bias.

Given these per-research question arguments and the arguments of the previous two subsections, the validity of the single-user conversion procedure seems to be acceptable for these datasets and the conclusions drawn by these research questions. While the results achieved by using the single-user conversion procedure to produce multiuser, simultaneous datasets are not as ideal as they would be if these types of datasets were available, some important conclusions were reached. Ultimately, some of the results are not affected by the biases and would not have been observed without the evaluation with the generated datasets, e.g., RQ-2 favoring learning methods over existing methods and RQ-5 showing the similarities of the learning methods across all three datasets when the HISTORY and OTHERS sets are combined. These conclusions would not have been possible using existing simulation methods with anywhere near the validity presented in this thesis. This is a result of simulations not realistically modeling the relationships between different types of context attributes, the uncertainty inherent in the information, and the manner in which that information changes over time.

In an attempt to counteract the OTHERS bias, an additional experiment in Appendix C.7 ran the tests with random noise added to the attributes of the OTHERS set. While adding noise to actual data was seen to be more valid than also simulating the underlying distribution, the experiments resulted in a dataset that still had the three disadvantages of simulations listed above. For this reason and others discussed in the appendix, this experiment was given minimal consideration.

5.8 Chapter Summary

The evaluations for the utility of other users were presented in this chapter. The statistical methods that were most appropriate for each dataset were selected, i.e., parametric tests for the Nokia dataset and non-parametric tests for the Locomotion and Helsinki datasets. The research questions were then addressed in the remaining sections, and the validity of the single-user conversion procedure was discussed in light of the results in this chapter and in the appendices.

The next chapter summarizes the conclusions of this chapter and ties them together for a higher-level view of what the results mean for the utility of other users' information in pervasive applications. It also presents the next steps of the work in this thesis.

Chapter 6

Conclusions and Future Work

The overall contribution of this thesis is a study that demonstrates the utility of other users' information for acquiring accurate context information in pervasive applications. This was accomplished by addressing the two problems with existing studies given in the motivation (Section 1.2). First, a valid study was produced by generating multiuser datasets exhibiting good external validity with the single-user conversion procedure and by designing the experiments that maximized all four types of validity listed in Section 1.1.3. Second, evaluations with learning methods that are appropriate for the characteristics of pervasive environments were carried out. Most importantly, the learning methods exhibited the situational relevance that existing context selection methods lack while functioning in sparse environments and with high knowledge autonomy.

This chapter concludes the thesis with a discussion of the achievements of this work (Section 6.1) and a proposal for the future directions (Section 6.2) in light of the conclusions of the last chapter.

6.1 Achievements

The summary of the achievements of this thesis is as follows:

A-1 A procedure that aids evaluation validity by generating multiuser datasets: There are currently no existing multiuser datasets on which the evaluations of this thesis can be run (Section 3.2.3). This thesis has presented the single-user conversion procedure (Section 3.2.4) to convert single-user, multi-run datasets into multiuser datasets. The conversion procedure results in datasets that ad-

dress the threats to validity that simulations of multiuser datasets present (see Section 1.1.3). The generated multiuser datasets have limited threats to validity for the purposes of the evaluations of this thesis (see Section 3.2.4.1, Section 5.7, and Appendix C). However, validity of the procedure is much improved if there are multiple, discernible users in the dataset. (Addressing RQ-1)

A-2 Knowledge of the utility of learning methods with situational relevance: This thesis demonstrated that the use of learning methods with situational relevance to select other users' information exhibits greater overall accuracy than existing selection methods that do not consider situational relevance (Section 5.2). This result is statistically significant and consistent across all three datasets with the exception of the CBR-OTHERS alternative in the Locomotion dataset which was better but not significantly different. (Addressing RQ-2)

A-3 Knowledge of the utility of other users' information by itself: This thesis has shown that neither the set of other users nor the previous experience of the application's user exhibit more overall accuracy than each other on a consistent basis (Section 5.7). That is, the set with more overall accuracy between these two is dataset-specific. This conclusion is true even when the two sets have equal sizes, i.e., the better performing set is not always the one with the greater size (Section 5.6). (Addressing RQ-3)

A-4 Knowledge of the utility of augmenting the set of other users' information: This thesis has shown that adding the previous experience set to the other users set results in no statistically significant improvement in overall accuracy when compared to the overall accuracy of the more accurate set by itself (Section 5.8). For the CBR and SVM approaches, adding instances from the less accurate set to the more accurate set showed no statistically significant differences in overall accuracy. For the naive Bayes and MultiBoosting approaches, there is a tendency to perform worse when instances from the set with less overall accuracy are added to the set with more overall accuracy (though the differences here are not always significant). In the three datasets tested, combining the two instance sets might increase the number of instances available, but it does not improve on the overall accuracy of the best individual set by itself. (Addressing

RQ-4)

A-5 Determination of the best approach for instance set selection: As was shown in A-4, there is no individual instance set that consistently demonstrates more overall accuracy across all datasets. For the CBR and SVM approaches, using the union of the previous experience and other users sets is not significantly different from the individual instance set with more overall accuracy. For these three datasets, the union set can always be chosen for the CBR and SVM approaches and there is no dataset-specific instance set selection knowledge required. For the naive Bayes and MultiBoosting approaches, there is an instance set selection knowledge requirement that is dataset-dependent as the union set does not always exhibit more overall accuracy than the better instance set. (Addressing RQ-5)

A-6 Determination of the best approach for sparse instance sets: When one set has very few instances, the union of the previous experience and other users sets tends to outperform the sparse set by itself. For the CBR and SVM approaches, the union set appears to always be the instance set with the most overall accuracy for full and sparse environments. For the naive Bayes and MultiBoosting approaches, the union set is only the instance set with the most overall accuracy for the initial sparseness and then the instance set exhibiting more overall accuracy for full environments should be selected. (Addressing RQ-6)

The utility of other users' information for acquiring accurate context information in pervasive applications has been demonstrated in a number of ways. Although other users' information is not always more accurate than the previous experience set, it is often quite useful. When confronted with limited previous experience, the inclusion of other users' information initially tends to outperform alternatives using previous experience by itself. In the full environment evaluations of this thesis, the other users set performed better than or equal to the previous experience set in two of the three datasets. The third dataset (the Locomotion dataset) contained context information that was very specific to the user being tested (i.e., how the user walked) and had very little shared environment context. The other datasets that did contain shared

environment context information performed much better with other users' information. This is the case even though the tests show a worst case scenario for shared environment context (see Section 3.2.4.1), which is a result of the single-user conversion procedure described in Section 3.2.4.

These favorable results for other users' information should generalize to actual multiuser environments with shared environment context. Conversely, previous experience would be favored in environments that contain little shared environment context attributes and focus on very user-specific qualities that also might not change very much over time, i.e., encountering unfamiliar situations would be rare. For example, a user's manner of walking is very specific to that user, rarely changes over time, and is not a quality shared by others in the user's vicinity, i.e., people rarely spend time with others because of similarities in the manner in which they walk.

For the stable learning methods of CBR and SVM, the utilization of other users' information in addition to the previous experience set has the added benefits of at least being no worse than the more accurate instance source set in both full and sparse instance sets. It also does not require a priori knowledge of which instance source is the most accurate. These benefits prevent the instance source set with worse overall accuracy from being selected and increase knowledge autonomy of the application at the cost of retrieving and processing the extra information. Another benefit of using CBR (and potentially SVM) to external validity of the evaluations' results is that the accuracy of the learning method appears to be robust to any negative transfer of knowledge from instances with less relevant situations, scenarios, and users. This suggests that as long as the collection of runs in the dataset would likely exist as a subset of the ones in an actual environment, the error should at least not increase as a greater range of runs are encountered in a real environment (see the last paragraph of Section 5.7.1).

In summary, other users' information is often a good source to aid the acquisition of accurate context information for pervasive applications. Consequently, other users' information often leads to more accurate application behavior as more accurate context information allows the behavior to improve.

6.2 Future Work

The future directions for the work of this thesis mostly concern improving the generality of the conclusions. The first two sections (Section 6.2.1 and Section 6.2.2) broaden the evaluation scope from the restricted one of this thesis (defined in Section 3.1). The weight learning results from the experiments in Appendix B prompt the work of the third section (Section 6.2.3). Finally, the last section proposes increasing the external validity by performing evaluations in actual multiuser environments (Section 6.2.4).

6.2.1 Sharing Information and Privacy

Users of pervasive applications have many reasons to avoid sharing information about themselves with other users. Privacy is a major concern for such users and is a major factor in the acceptance of pervasive applications [117]. In addition to privacy, the sensing, derivation, and communication of other users' information are a drain on the limited resources of pervasive applications, e.g., the resources of time, processor, battery, and bandwidth. Some of these resource costs are explicitly considered by the application during context acquisition (see Section 6.2.2).

The work of this thesis depends on other users sharing information about themselves, requiring values for the desired output attribute as well as the associated input attributes for determining situational relevance. Some answers to these problems are to limit concerns about sharing information by how the sharing is implemented and to provide motivation for other users to want to share information about themselves. For example, users on Facebook willingly share information about themselves because they have control over who receives that information and because the sharing provides benefits, e.g., sharing news or preferences with Facebook friends, that outweigh those concerns. As another example, Sheikh et al. [122] suggest an approach where pervasive applications use lower quality context information in order to protect a user's privacy while still allowing the sharing of information about users.

Evaluations should be performed that explore these tradeoffs and determine the feasibility of using the learning methods in this thesis to select other users' information.

6.2.2 Other Context Selection Parameters

The evaluations of this thesis only consider overall accuracy when selecting the best approaches and instance sets. In addition to privacy, other parameters are important to pervasive applications, e.g., trustworthiness, uncertainty, retrieval costs, processing costs, and timeliness. Many of these are listed in the survey of existing context selection methods in Section 2.1.

Parameters like timeliness aim to improve accuracy but some of them are orthogonal or even conflict with accuracy. As an example for the CBR approach, the union set is recommended as a means to avoid instance set selection knowledge requirements while also maintaining good overall accuracy. The improvements in accuracy and knowledge autonomy may not be worth the retrieval and processing costs for a specific pervasive application or user.

Like the evaluations in the previous section, evaluations that explore overall accuracy in addition to the other context selection parameters would increase the generalizability of the work. However, valid evaluations are difficult to perform as most pervasive datasets do not contain information about a user's relative preferences for each parameter.

6.2.3 Learning Weights from Other Users' Information

One of the results from the weight learning experiments in Appendix B was that the instance set that the weights were trained on produced minimal variation in average estimation error when compared to the variation between the experiment alternatives. It is hypothesized that one of the main reasons for this was the lack of static, personal information like age or role in the datasets.

The lack of this information meant that each run could be treated as a different user, thereby allowing the single-user conversion procedure to be used. However, it also meant that training weights on the previous experience of a user had sufficient variation to learn the importance of the different attributes. If the attributes were static, this variation would not exist. For example, learning the weights from a user's previous experience would give no importance to a user's age. This is because age rarely changes for a user so the importance of different ages for estimating an output

attribute cannot be learned. If age is highly predictive for activity, then the acquisition of an activity from the set of other users with different ages will not be as accurate.

Evaluations that use datasets with static attributes as well as dynamic attributes are necessary to explore the utility of learning weights from other users' information. A challenge to performing these evaluations is that no datasets currently exist that allow them to be run.

6.2.4 Evaluations in Actual Multiuser, Simultaneous Environments

The single-user conversion procedure allows multiuser datasets to be generated with a higher degree of external validity than simulations. However, to maximize the external validity for the research questions and to perform many of the evaluations suggested in this Future Work section, experiments in actual multiuser, simultaneous environments should be carried out. The overall accuracy for shared-environment context information should show improvement when acquiring information from other users rather than previous experience, as users in the same environment are more likely to be sensing the same phenomena. Static variables could also be used to measure the utility of learning the weights from other users, as described in the previous section. Finally, the willingness to share information and the relative importance of context selection parameters other than overall accuracy could be assessed if given access to the users under study.

Appendix A

Statistical Outputs

The following figures are the Minitab outputs for the evaluations of Chapter 5 for all the datasets. Each dataset section contains outputs for the one-way repeated ANOVA test, the Tukey test, the Friedman test, and the Wilcoxon signed-rank test. For the Wilcoxon signed-rank test, only a subset of this test's comparisons are used, i.e., the ones for the sixteen comparisons required to answer the research questions (see Table 5.2).

Nokia Dataset

Analysis of Variance for Average Estimation Error, using Adjusted SS for Tests

Source	DF	Seq SS	Adj SS	Adj MS	F	P
Alternative	14	2.217392	2.217392	0.158385	53.11	0.000
User	41	1.640876	1.640876	0.040021	13.42	0.000
Error	574	1.711921	1.711921	0.002982		
Total	629	5.570189				

S = 0.0546117 R-Sq = 69.27% R-Sq(adj) = 66.32%

Figure A.1: Minitab output for ANOVA test with the Nokia dataset.

Grouping Information Using Tukey Method and 95.0% Confidence

Alternative	N	Mean	Grouping
08–SVM–HISTORY	42	0.53	A
14–RANDOM–HISTORY	42	0.51	A B
15–RANDOM–BOTH	42	0.51	A B C
13–RANDOM–OTHERS	42	0.50	A B C
05–BAYES–HISTORY	42	0.48	B C
11–BOOST–HISTORY	42	0.48	B C
02–CBR–HISTORY	42	0.47	C D
06–BAYES–BOTH	42	0.43	D E
12–BOOST–BOTH	42	0.43	E
04–BAYES–OTHERS	42	0.38	F
10–BOOST–OTHERS	42	0.38	F
07–SVM–OTHERS	42	0.37	F
01–CBR–OTHERS	42	0.37	F
09–SVM–BOTH	42	0.37	F
03–CBR–BOTH	42	0.37	F

Means that do not share a letter are significantly different.

Figure A.2: Minitab output with the Tukey test for the Nokia dataset.

Friedman Test: Average Estimation Error versus Alternative blocked by User

S = 349.59 DF = 14 P = 0.000
S = 349.61 DF = 14 P = 0.000 (adjusted for ties)

Alternative	N	Est	Median	Sum of Ranks
01–CBR–OTHERS	42		0.3578	158.0
02–CBR–HISTORY	42		0.4701	418.0
03–CBR–BOTH	42		0.3662	174.0
04–BAYES–OTHERS	42		0.3663	227.0
05–BAYES–HISTORY	42		0.4758	416.5
06–BAYES–BOTH	42		0.4298	299.5
07–SVM–OTHERS	42		0.3626	171.0
08–SVM–HISTORY	42		0.5413	545.0
09–SVM–BOTH	42		0.3691	185.0
10–BOOST–OTHERS	42		0.3763	205.0
11–BOOST–HISTORY	42		0.4711	427.0
12–BOOST–BOTH	42		0.4225	296.0
13–RANDOM–OTHERS	42		0.4884	476.0
14–RANDOM–HISTORY	42		0.5103	537.0
15–RANDOM–BOTH	42		0.5042	505.0

Grand median = 0.4341

Figure A.3: Minitab output for Friedman test with the Nokia dataset.

Wilcoxon Signed Rank CI

	N	Estimated Median	Achieved Confidence	Confidence Interval Lower	Upper
Avg Est Err_01–CBR–OTHERS	42	0.3641	99.7	0.3360	0.3947
Avg Est Err_02–CBR–HISTORY	42	0.4763	99.7	0.4311	0.5178
Avg Est Err_03–CBR–BOTH	42	0.3699	99.7	0.3362	0.3987
Avg Est Err_04–BAYES–OTHERS	42	0.381	99.7	0.328	0.437
Avg Est Err_05–BAYES–HISTORY	42	0.4831	99.7	0.4477	0.5163
Avg Est Err_06–BAYES–BOTH	42	0.4385	99.7	0.4007	0.4704
Avg Est Err_07–SVM–OTHERS	42	0.3704	99.7	0.3324	0.4093
Avg Est Err_08–SVM–HISTORY	42	0.5430	99.7	0.5043	0.5705
Avg Est Err_09–SVM–BOTH	42	0.3726	99.7	0.3385	0.4051
Avg Est Err_10–BOOST–OTHERS	42	0.3777	99.7	0.3423	0.4109
Avg Est Err_11–BOOST–HISTORY	42	0.4761	99.7	0.4341	0.5170
Avg Est Err_12–BOOST–BOTH	42	0.4241	99.7	0.3811	0.4666
Avg Est Err_13–RANDOM–OTHERS	42	0.4947	99.7	0.4636	0.5273
Avg Est Err_14–RANDOM–HISTORY	42	0.5200	99.7	0.4898	0.5465
Avg Est Err_15–RANDOM–BOTH	42	0.5140	99.7	0.4844	0.5394

Figure A.4: Minitab output for Wilcoxon signed-rank test with the Nokia dataset.

Locomotion Dataset

Analysis of Variance for Average Estimation Error, using Adjusted SS for Tests

Source	DF	Seq SS	Adj SS	Adj MS	F	P
Alternative	14	1.711083	1.711083	0.122220	324.29	0.000
User	12	0.054936	0.054936	0.004578	12.15	0.000
Error	168	0.063316	0.063316	0.000377		
Total	194	1.829335				

S = 0.0194135 R–Sq = 96.54% R–Sq(adj) = 96.00%

Figure A.5: Minitab output for ANOVA test with the Locomotion dataset.

Grouping Information Using Tukey Method and 95.0% Confidence

Alternative	N	Mean	Grouping
13–RANDOM–OTHERS	13	1.0	A
01–CBR–OTHERS	13	0.8	B
04–BAYES–OTHERS	13	0.8	C
14–RANDOM–HISTORY	13	0.9	C
15–RANDOM–BOTH	13	0.9	C
10–BOOST–OTHERS	13	0.8	C D
07–SVM–OTHERS	13	0.8	D
05–BAYES–HISTORY	13	0.6	E
06–BAYES–BOTH	13	0.6	E
11–BOOST–HISTORY	13	0.6	E
12–BOOST–BOTH	13	0.6	E
09–SVM–BOTH	13	0.5	F
08–SVM–HISTORY	13	0.5	F
03–CBR–BOTH	13	0.5	F
02–CBR–HISTORY	13	0.5	F

Means that do not share a letter are significantly different.

Figure A.6: Minitab output with the Tukey test for the Locomotion dataset.

Friedman Test: Average Estimation Error versus Alternative blocked by User

S = 171.35 DF = 14 P = 0.000
S = 171.44 DF = 14 P = 0.000 (adjusted for ties)

Alternative	N	Est	Median	Sum of Ranks
01–CBR–OTHERS	13		0.4829	178.0
02–CBR–HISTORY	13		0.2351	25.0
03–CBR–BOTH	13		0.2357	30.0
04–BAYES–OTHERS	13		0.4620	149.0
05–BAYES–HISTORY	13		0.3234	94.0
06–BAYES–BOTH	13		0.3233	90.0
07–SVM–OTHERS	13		0.3964	127.0
08–SVM–HISTORY	13		0.2382	30.0
09–SVM–BOTH	13		0.2429	45.0
10–BOOST–OTHERS	13		0.4206	142.0
11–BOOST–HISTORY	13		0.3099	76.0
12–BOOST–BOTH	13		0.3096	78.0
13–RANDOM–OTHERS	13		0.5087	195.0
14–RANDOM–HISTORY	13		0.4350	155.0
15–RANDOM–BOTH	13		0.4322	146.0

Grand median = 0.3571

Figure A.7: Minitab output for Friedman test with the Locomotion dataset.

Wilcoxon Signed Rank CI

	N	Estimated Median	Achieved Confidence	Confidence Interval Lower	Upper
Avg Est Err_01–CBR–OTHERS	13	0.4784	99.7	0.4431	0.5190
Avg Est Err_02–CBR–HISTORY	13	0.2369	99.7	0.2049	0.2563
Avg Est Err_03–CBR–BOTH	13	0.2372	99.7	0.2056	0.2567
Avg Est Err_04–BAYES–OTHERS	13	0.4545	99.7	0.4160	0.4763
Avg Est Err_05–BAYES–HISTORY	13	0.3279	99.7	0.2914	0.3699
Avg Est Err_06–BAYES–BOTH	13	0.3268	99.7	0.2914	0.3681
Avg Est Err_07–SVM–OTHERS	13	0.405	99.7	0.364	0.477
Avg Est Err_08–SVM–HISTORY	13	0.2399	99.7	0.2260	0.2505
Avg Est Err_09–SVM–BOTH	13	0.2473	99.7	0.2250	0.2568
Avg Est Err_10–BOOST–OTHERS	13	0.4263	99.7	0.3980	0.4772
Avg Est Err_11–BOOST–HISTORY	13	0.3111	99.7	0.2931	0.3357
Avg Est Err_12–BOOST–BOTH	13	0.3113	99.7	0.2934	0.3330
Avg Est Err_13–RANDOM–OTHERS	13	0.5090	99.7	0.4909	0.5471
Avg Est Err_14–RANDOM–HISTORY	13	0.4391	99.7	0.4132	0.4677
Avg Est Err_15–RANDOM–BOTH	13	0.4357	99.7	0.4101	0.4613

Figure A.8: Minitab output for Wilcoxon signed-rank test with the Locomotion dataset.

Helsinki Dataset

Analysis of Variance for Average Estimation Error, using Adjusted SS for Tests

Source	DF	Seq SS	Adj SS	Adj MS	F	P
Alternative	14	7.03400	7.03400	0.50243	1068.66	0.000
User	239	3.92861	3.92861	0.01644	34.96	0.000
Error	3346	1.57311	1.57311	0.00047		
Total	3599	12.53572				

S = 0.0216828 R–Sq = 87.45% R–Sq(adj) = 86.50%

Figure A.9: Minitab output for ANOVA with the Helsinki dataset.

Grouping Information Using Tukey Method and 95.0% Confidence

Alternative	N	Mean	Grouping
08–RANDOM–HISTORY	240	0.4	A
09–RANDOM–BOTH	240	0.4	B
07–RANDOM–OTHERS	240	0.4	B
04–BAYES–OTHERS	240	0.3	C
11–BOOST–HISTORY	240	0.2	C
06–BAYES–BOTH	240	0.2	C
10–BOOST–OTHERS	240	0.3	C
12–BOOST–BOTH	240	0.2	C
08–SVM–HISTORY	240	0.2	C
05–BAYES–HISTORY	240	0.2	C
02–CBR–HISTORY	240	0.2	D
07–SVM–OTHERS	240	0.1	E
09–SVM–BOTH	240	0.1	E
01–CBR–OTHERS	240	0.1	E
03–CBR–BOTH	240	0.1	E

Means that do not share a letter are significantly different.

Figure A.10: Minitab output with the Tukey test for the Helsinki dataset.

Friedman Test: Average Estimation Error versus Alternative blocked by User

S = 2758.51 DF = 14 P = 0.000

Alternative	N	Est Median	Sum of Ranks
01–CBR–OTHERS	240	0.05793	567.0
02–CBR–HISTORY	240	0.09527	1544.0
03–CBR–BOTH	240	0.05759	525.0
04–BAYES–OTHERS	240	0.12152	2289.0
05–BAYES–HISTORY	240	0.11509	2031.0
06–BAYES–BOTH	240	0.11784	2081.0
07–SVM–OTHERS	240	0.06365	807.0
08–SVM–HISTORY	240	0.11108	1956.0
09–SVM–BOTH	240	0.06184	620.0
10–BOOST–OTHERS	240	0.11720	2130.0
11–BOOST–HISTORY	240	0.11799	2155.0
12–BOOST–BOTH	240	0.11613	2069.0
13–RANDOM–OTHERS	240	0.17864	3258.0
14–RANDOM–HISTORY	240	0.19679	3484.0
15–RANDOM–BOTH	240	0.18121	3284.0

Grand median = 0.11399

Figure A.11: Minitab output for Friedman test with the Helsinki dataset.

Wilcoxon Signed Rank CI

	N	Estimated Median	Achieved Confidence	Confidence Interval Lower	Upper
Avg Est Err_01–CBR–OTHERS	240	0.05816	99.7	0.05512	0.06159
Avg Est Err_02–CBR–HISTORY	240	0.0986	99.7	0.0910	0.1068
Avg Est Err_03–CBR–BOTH	240	0.05784	99.7	0.05464	0.06145
Avg Est Err_04–BAYES–OTHERS	240	0.1231	99.7	0.1173	0.1290
Avg Est Err_05–BAYES–HISTORY	240	0.1162	99.7	0.1104	0.1221
Avg Est Err_06–BAYES–BOTH	240	0.1195	99.7	0.1135	0.1257
Avg Est Err_07–SVM–OTHERS	240	0.06375	99.7	0.06032	0.06745
Avg Est Err_08–SVM–HISTORY	240	0.1141	99.7	0.1081	0.1207
Avg Est Err_09–SVM–BOTH	240	0.06163	99.7	0.05849	0.06507
Avg Est Err_10–BOOST–OTHERS	240	0.1180	99.7	0.1142	0.1219
Avg Est Err_11–BOOST–HISTORY	240	0.1201	99.7	0.1129	0.1280
Avg Est Err_12–BOOST–BOTH	240	0.1173	99.7	0.1135	0.1212
Avg Est Err_13–RANDOM–OTHERS	240	0.1800	99.7	0.1756	0.1845
Avg Est Err_14–RANDOM–HISTORY	240	0.2020	99.7	0.1935	0.2114
Avg Est Err_15–RANDOM–BOTH	240	0.1813	99.7	0.1770	0.1860

Figure A.12: Minitab output for Wilcoxon signed-rank test with the Helsinki dataset.

Appendix B

Weight Learning Experiments

Figure B.1, Figure B.2, and Figure B.3 show how the average estimation error fluctuates for each retrieval source when the different instance sources are used to train the weights for the CBR approach. In all three figures, there are three groups of bars that indicate how the average estimation error is affected when the weights are learned from the OTHERS set (WTS_OTHERS), the HISTORY set (WTS_HISTORY), and the BOTH set (WTS_BOTH). For example, the second bar in each chart reflects the average estimation error when the weights are trained from the OTHERS set and used to retrieve from the HISTORY set (i.e., the HISTORY-WTS_OTHERS alternative). The three figures show how little the choice of weight training set affects average estimation error. Each chart shows relatively little variation when training from different weight sources when compared to the variation of retrieving from different sources.

Figure B.4 shows the effect on average estimation error with the Nokia dataset for the different weight training sources when retrieving from the OTHERS set. In addition to the different weight training sets, the figure also shows the performance when all the weights for the attributes are equal (EQUAL), i.e., when there is no training of weights. Similar graphs for all the permutation of retrieval source and dataset show little variation in average estimation error when the weights are set to equal. This allows the experiments of Section 5.6.2 to safely use the equal weights method with the CBR approach. This isolates the effect of sparse previous experience on average estimation error without confounding the results with the manner in which the Stahl algorithm affects the overall accuracy with a sparse training set.

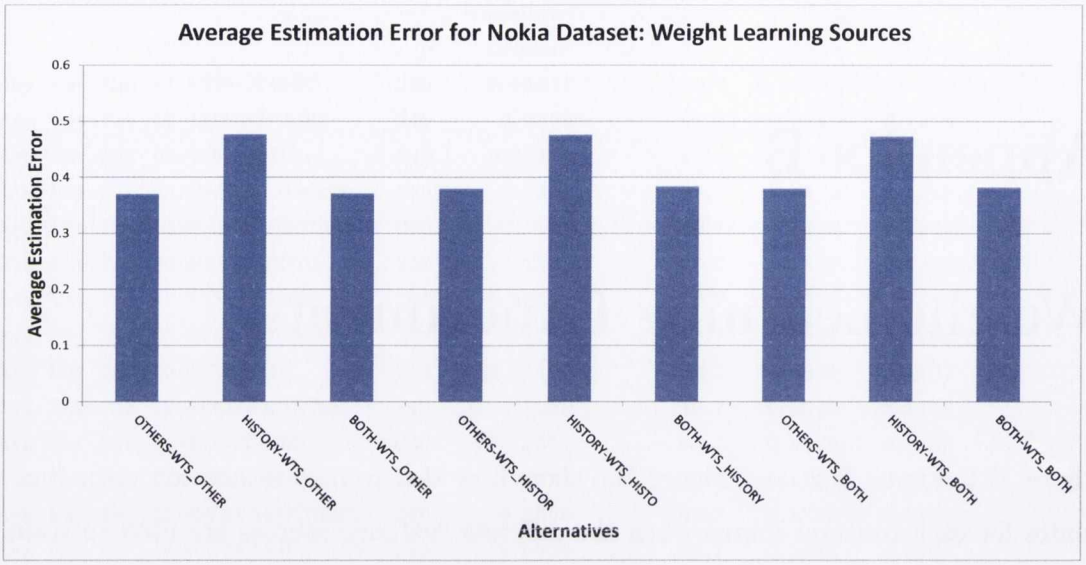

Figure B.1: Full environment CBR weight learning sources for the Nokia dataset.

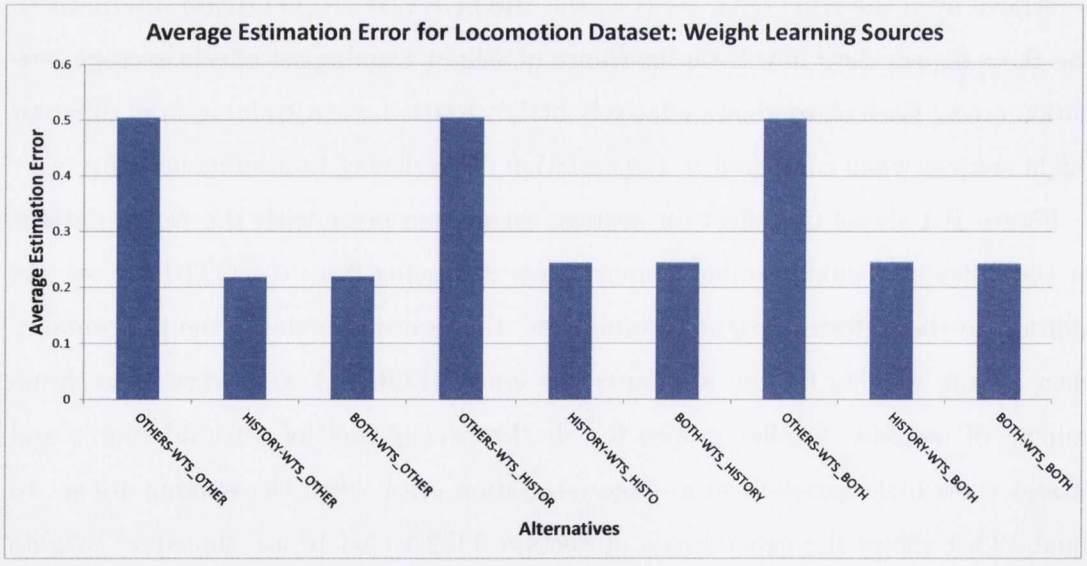

Figure B.2: Full environment CBR weight learning sources for the Locomotion dataset.

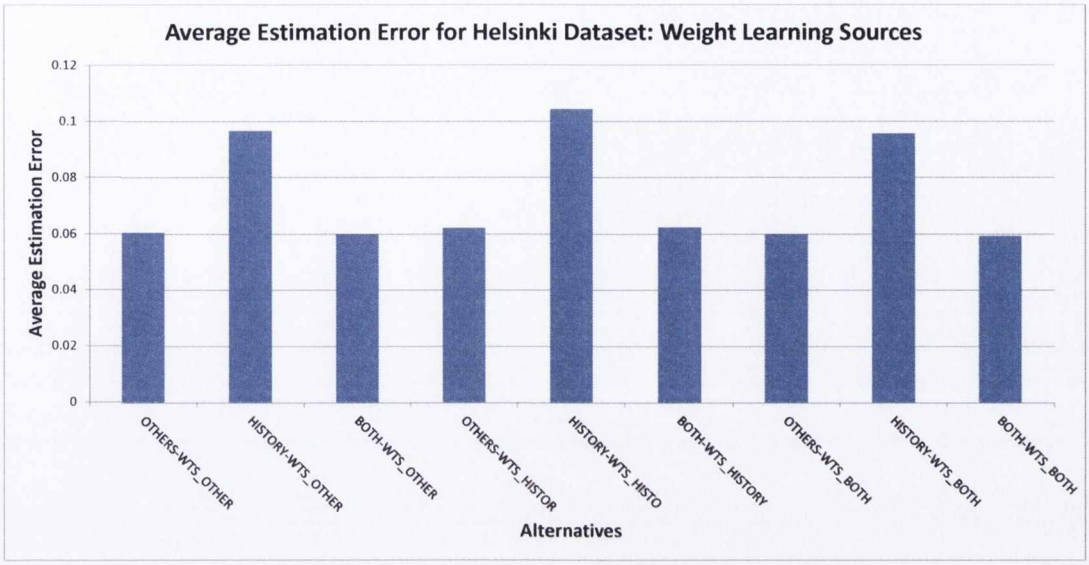

Figure B.3: Full environment CBR weight learning sources for the Helsinki dataset.

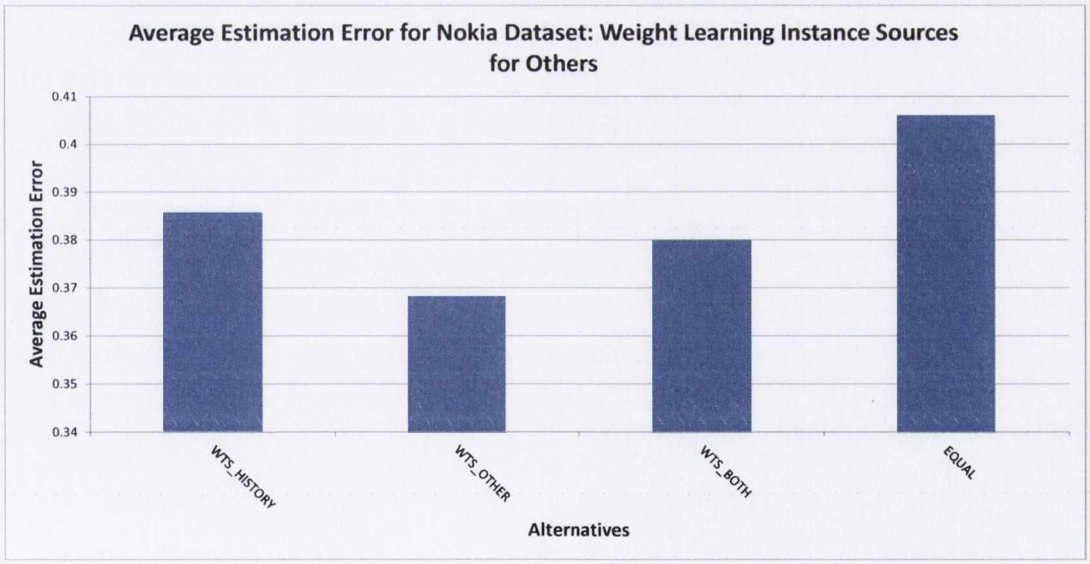

Figure B.4: Full environment CBR weight learning sources selecting from the set of other users for the Nokia dataset.

Appendix C

Single-User Conversion Procedure Validity Experiments

In order to analyze the threats to validity of the single-user conversion procedure described in Section 3.2.4, the experiments of the following sections help to show the degree to which the hypotheses made in Section 3.2.4.1 are valid. The results of these experiments are further discussed within the context of the thesis' evaluation results in Section 5.7.

C.1 Per-Attribute Results for the Full Environment Tests

Figures C.1, C.2, C.3, and C.4 show the average estimation error for each individual attribute for the full environment tests (Section 5.1). The brackets surrounding each bar show approximately a 95% confidence interval on the mean, giving an idea of how much each alternative varies for each attribute.

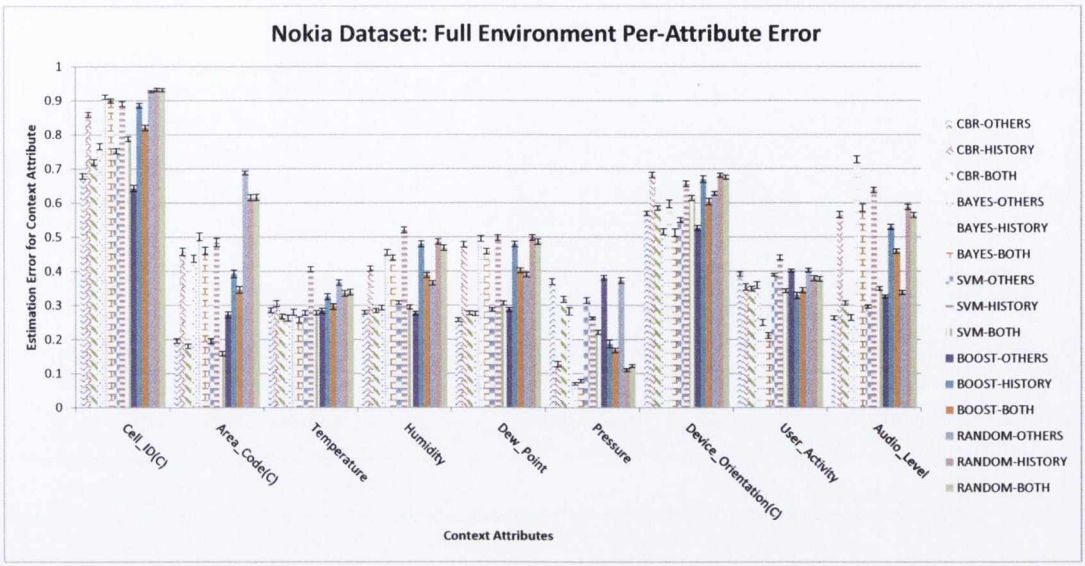

Figure C.1: Full environment per-attribute alternative means for the Nokia dataset.

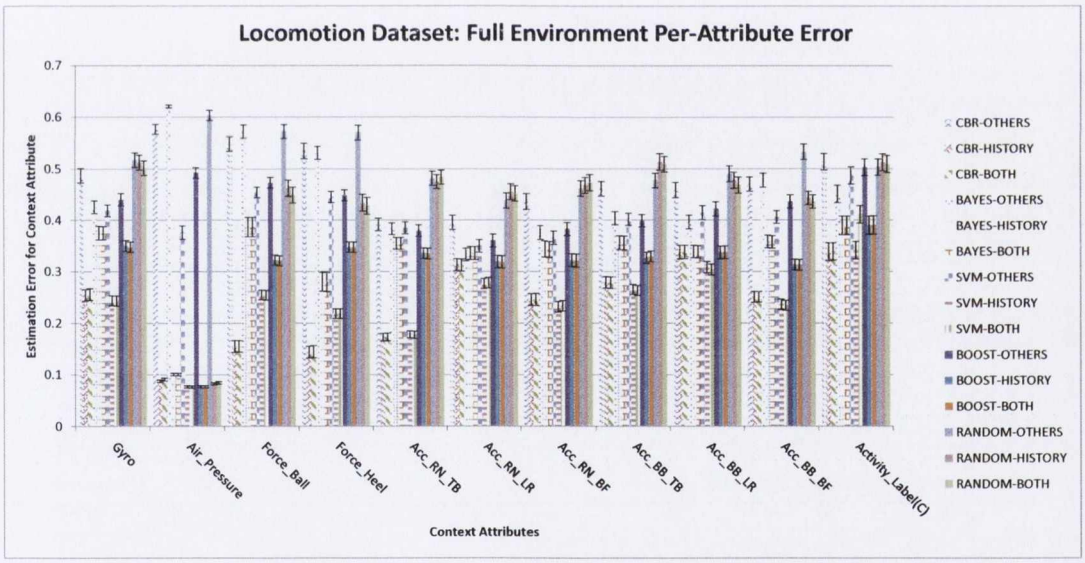

Figure C.2: Full environment per-attribute alternative means for the Locomotion dataset.

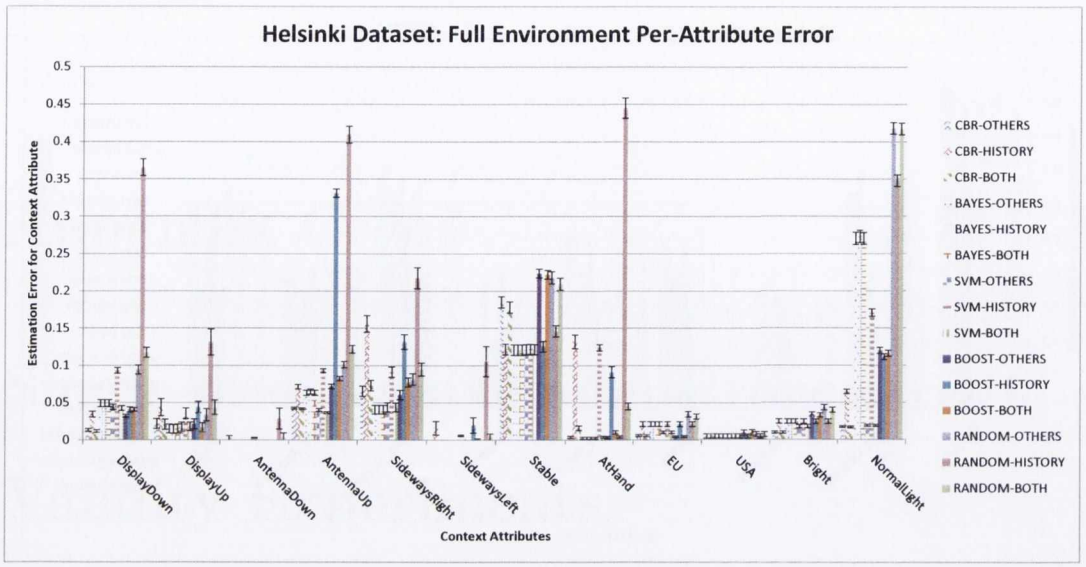

Figure C.3: Full environment per-attribute alternative means for the Helsinki dataset (Part 1).

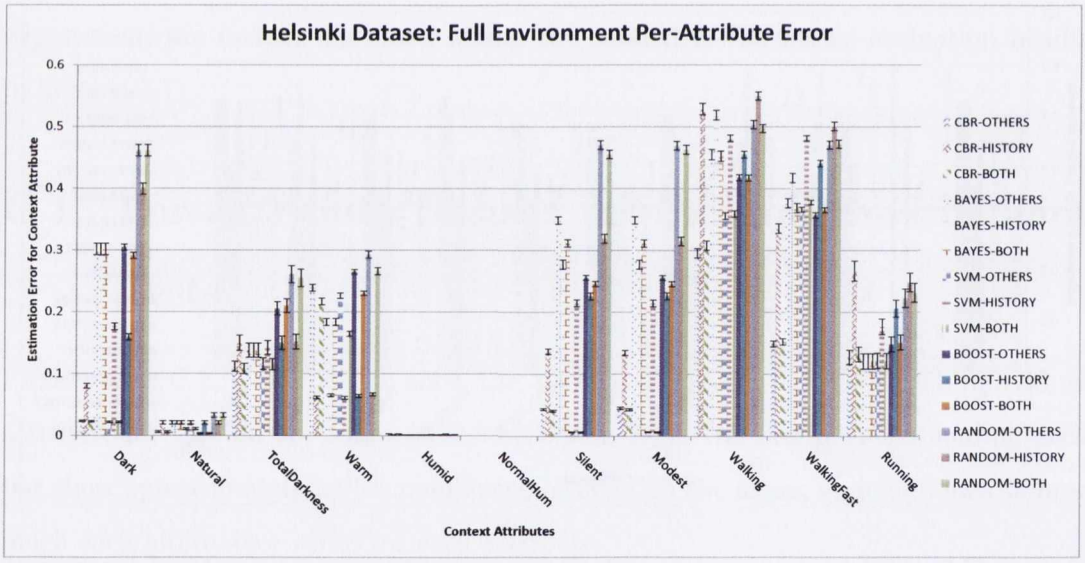

Figure C.4: Full environment per-attribute alternative means for the Helsinki dataset (Part 2).

C.2 Graphs of Non-Randomized Start Time

The randomized start time for the single-user conversion procedure (see Section 3.2.4)

was introduced to lessen the bias in favor of other users introduced by starting the runs performing the same scenario at the same time. Figure C.5, Figure C.6, and Figure C.7 show the results of running the full environment tests without randomizing the start times. Comparisons with the corresponding graphs in Section 5.1 that do have randomized start time show very little difference. This leads to the conclusion that randomizing the start time is not necessary to decrease the bias.

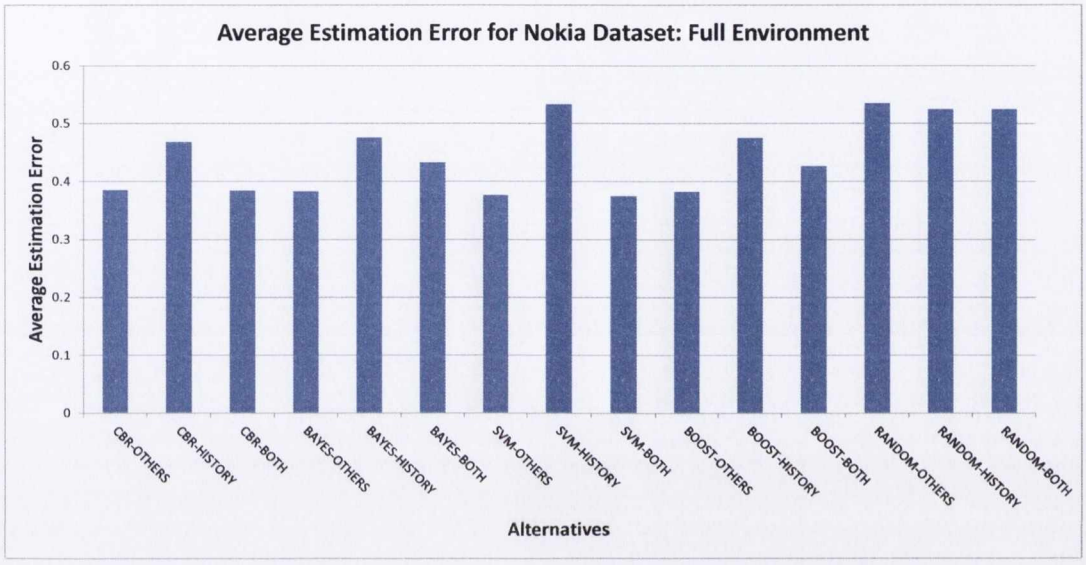

Figure C.5: Full environment without randomized start time for the Nokia dataset.

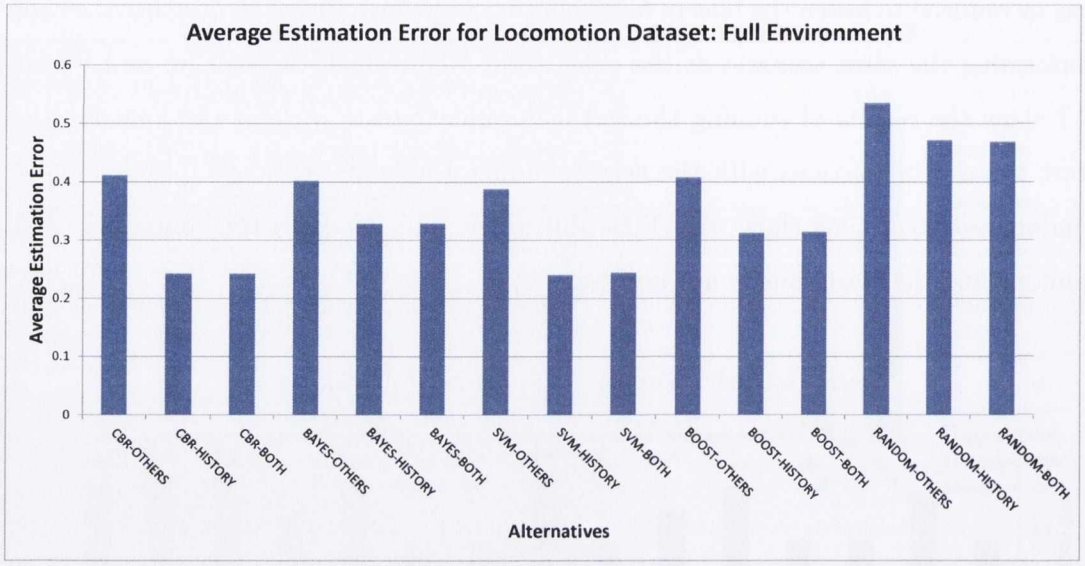

Figure C.6: Full environment without randomized start time for the Locomotion dataset.

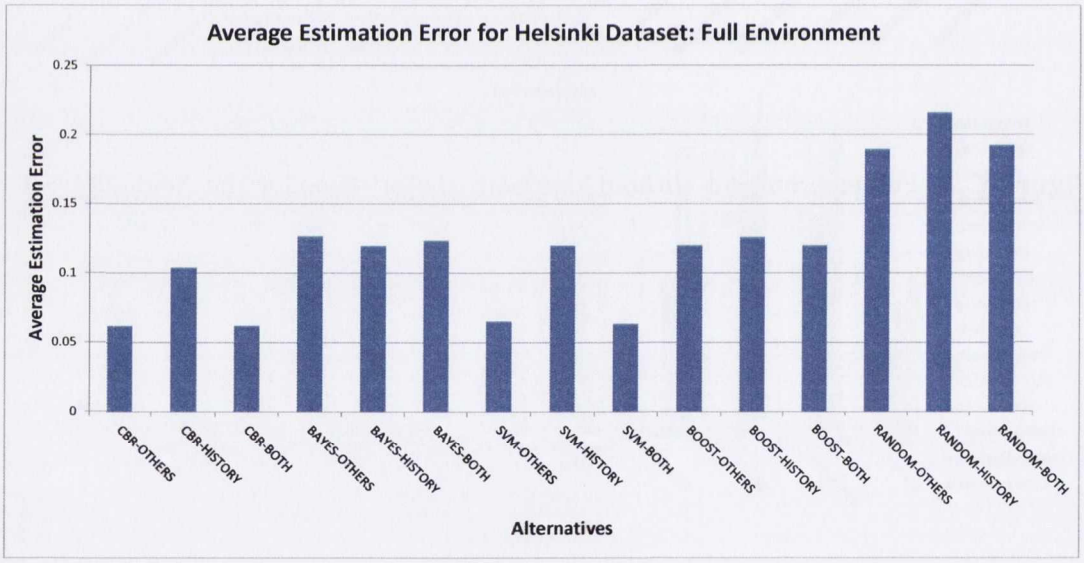

Figure C.7: Full environment without randomized start time for the Helsinki dataset.

C.3 Average Attribute Difference Between Runs

This section examines the threat to validity of using runs performing the same scenario. The concern is that the runs of the same scenario will be almost identical, thereby giv-

ing a favorable bias in accuracy from the set of other users. For the three datasets, Figure C.8, Figure C.9, and Figure C.10 show how different the runs' attributes are from each other. Specifically, the graphs show the average difference between runs for all attributes for different alternatives. The alternatives represent four different subsets of the set of other users and one alternative representing the previous experience of the user for comparison. There are five alternatives shown in the figures:

- *The IN_SCENARIO alternative*: All the runs that perform the same scenario as the test run. For the Locomotion dataset, this is all the runs that are from the same user.

- *The OUT_SCENARIO alternative*: All the runs that perform different scenarios to the test run. For the Locomotion dataset, this is all the runs that are from a different user.

- *The TOTAL alternative*: The combination of the OUT_SCENARIO and IN_SCENARIO, i.e., all runs of all users.

- *The MOST_SIMILAR alternative*: A set of size one consisting of the run that is most similar to the run of the current user. The similarity of two runs is identified by calculating the aggregate similarity of attribute values at each time instance and summing those similarity values for all time instances. The run with the highest value for this overall similarity sum is selected as the most similar. The similarity at each time instance is computed using the global similarity measure described in Section 3.4.2 with equal weights for each attribute's local similarity.

- *The HISTORY alternative*: The previous experience of the user represented by the initial instances of a run, i.e., the HISTORY set described in Section 4.1.1. The HISTORY set remains constant throughout the run for each user.

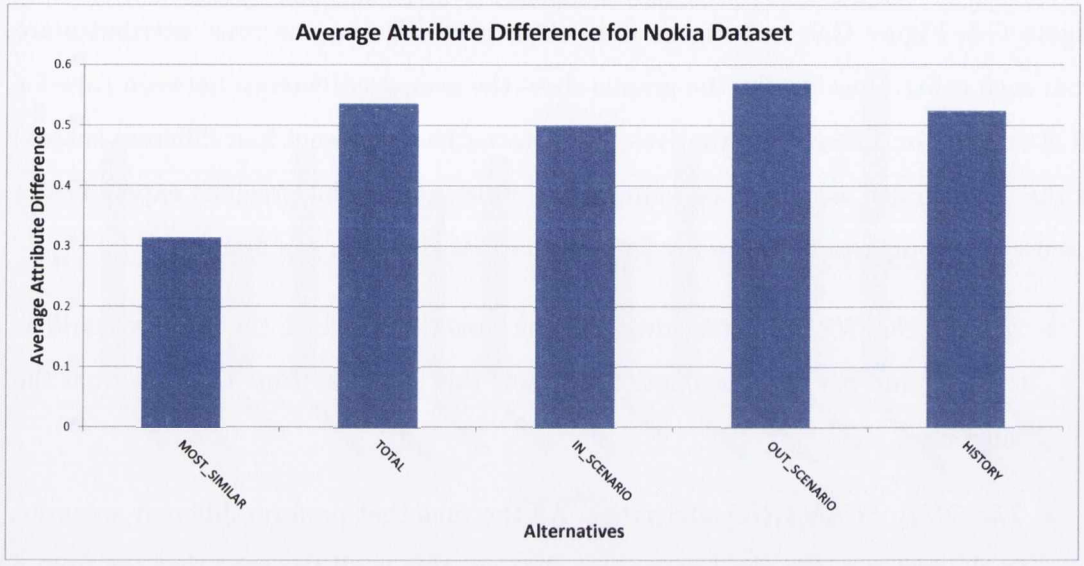

Figure C.8: The average attribute difference between runs for the Nokia dataset.

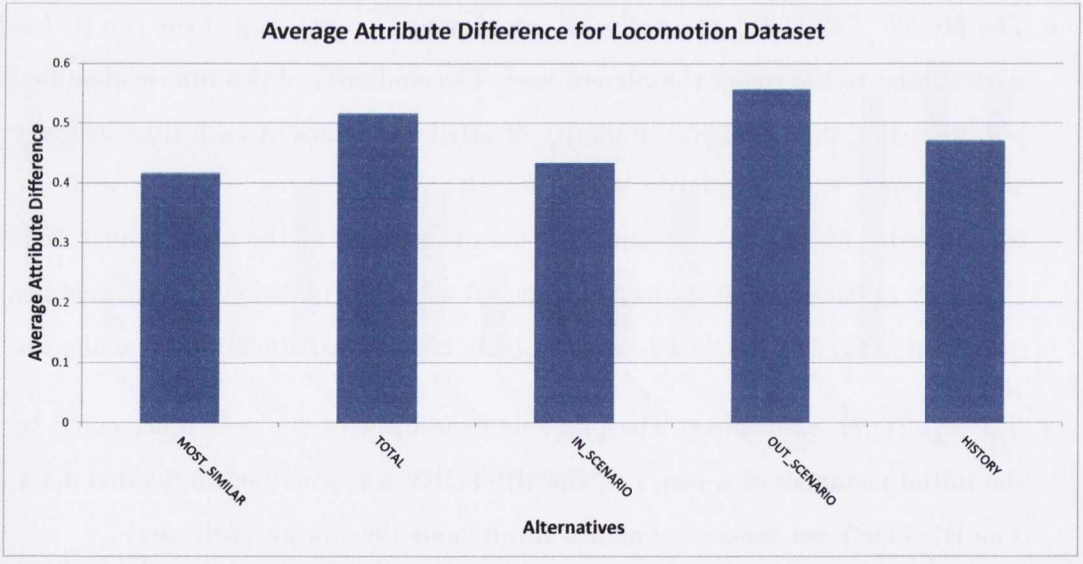

Figure C.9: The average attribute difference between runs for the Locomotion dataset.

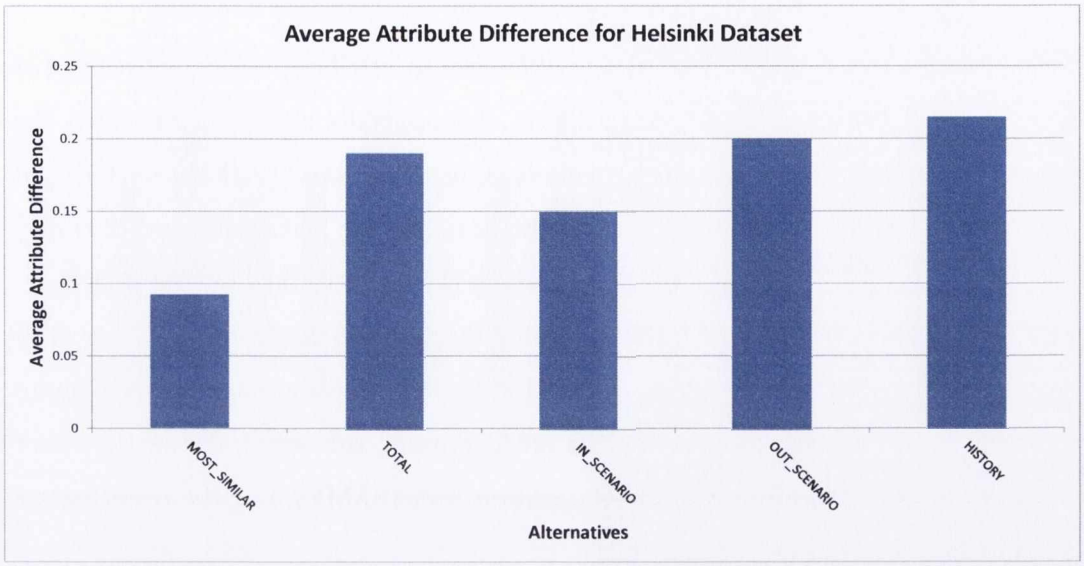

Figure C.10: The average attribute difference between runs for the Helsinki dataset.

All three of the figures use the dataset partitioning and use the same calculations for the average estimation error measure described in Chapter 4. Instead of running a learning method to estimate the value as described in that chapter, the average attribute difference is the difference in magnitude between the test instance's attribute value and the corresponding value of each run in the alternative's training set averaged across the entire test run. In addition, there is no randomized start time for the runs, so any similarity is a worst case scenario that will not be present when the randomized start time is performed for the main evaluation's tests.

As would be expected, the TOTAL alternative shows that the inclusion of all other scenarios and users generally has an average difference of greater magnitude than the IN_SCENARIO alternative but with a smaller average difference than the OUT_SCENARIO alternative. For all three datasets, the average attribute difference does not approach zero for any of the alternatives. This is true even when comparing the average difference of runs from the same scenario (i.e., the IN_SCENARIO alternative) and the average difference between its most similar run (i.e., the MOST_SIMILAR alternative). This means that there is a clear difference between the attributes of runs on average, even when the most similar run is known beforehand.[1]

[1]The MOST_SIMILAR alternative uses information that is not available for the tests in the main

C.4 Effect of Other Scenarios and Users

Where Appendix C.3 showed the average difference in attribute values between runs for a variety of subsets of users and scenarios, this appendix shows the effect on the average estimation error of introducing variety into runs of the OTHERS set for each learning method of the tests used in the evaluation. For the Nokia (Figure C.11) and Helsinki (Figure C.12) datasets, the graphs show the effect of introducing users performing different scenarios (OUT) to runs with the scenario of the current user (IN) resulting in a set with all scenarios (TOTAL). For the Locomotion dataset (Figure C.14), where the scenario is the same for all four users, this graph shows the effect of adding different users (OTHER) to the current user (SAME), i.e., the resulting set including all users (ALL).

For the Nokia and Locomotion datasets, the introduction of different scenario (IN versus TOTAL) and users (SAME versus ALL), respectively, has little effect on the error. For the Helsinki dataset, all the learning methods but CBR have a difference that is significantly greater than zero between the IN and TOTAL alternatives. For the Helsinki dataset, this means that the instances of the OUT alternative negatively affect the error even when instances of the same scenario exist in the TOTAL set. The important conclusion drawn from these graphs is that for the Nokia and Locomotion datasets, variety in user and scenario does not affect the accuracy any differently than if they were the same user and scenario. Particularly for CBR, across all datasets it appears to always select the best instance regardless of the other instances present.

evaluation of this thesis, i.e., it uses the information from other users at all the timestamps and not just the current timestamp to calculate the most similar run. Even with this extra information aiding similarity, there is still a clear difference between the attributes on average.

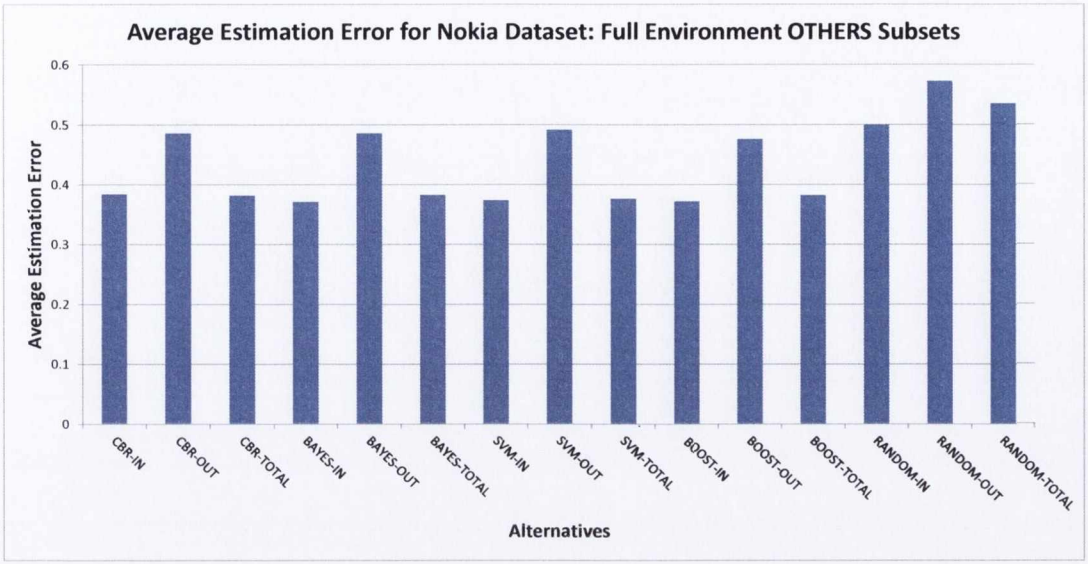

Figure C.11: Full environment scenario subsets for the Nokia dataset.

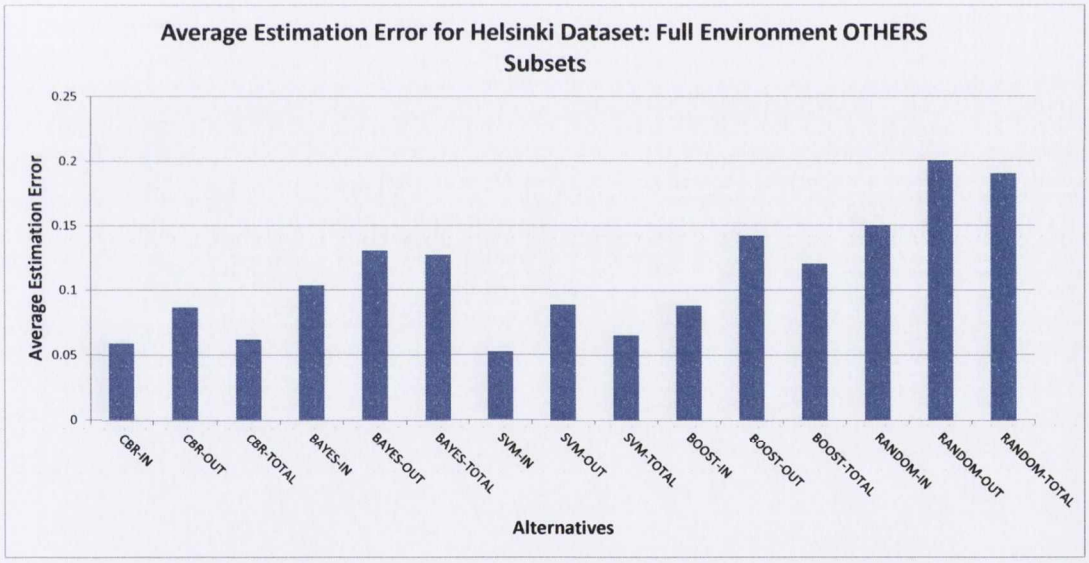

Figure C.12: Full environment scenario subsets for the Helsinki dataset.

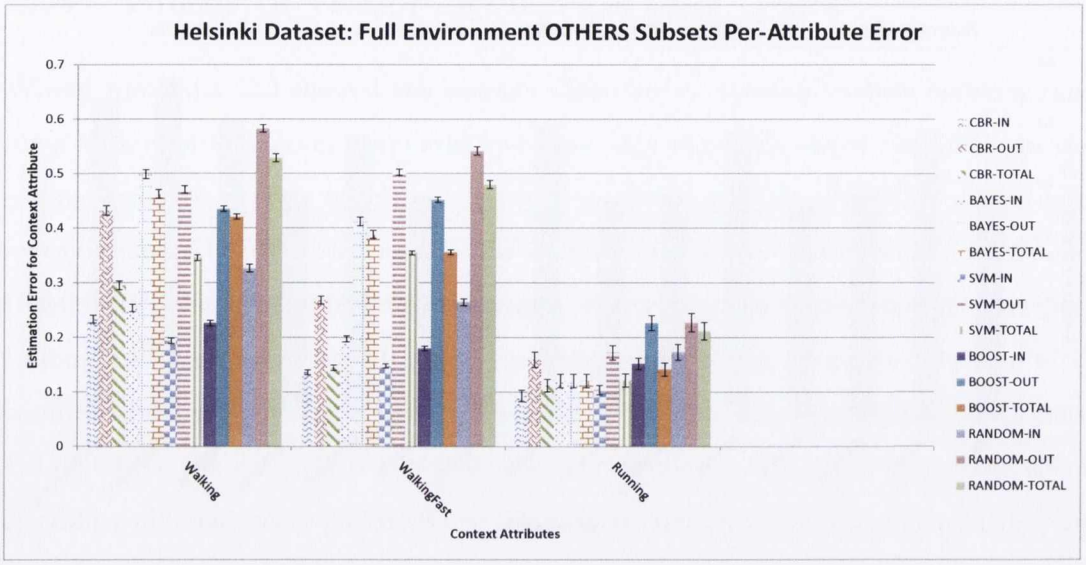

Figure C.13: Per-attribute full environment scenario subsets for the Helsinki dataset.

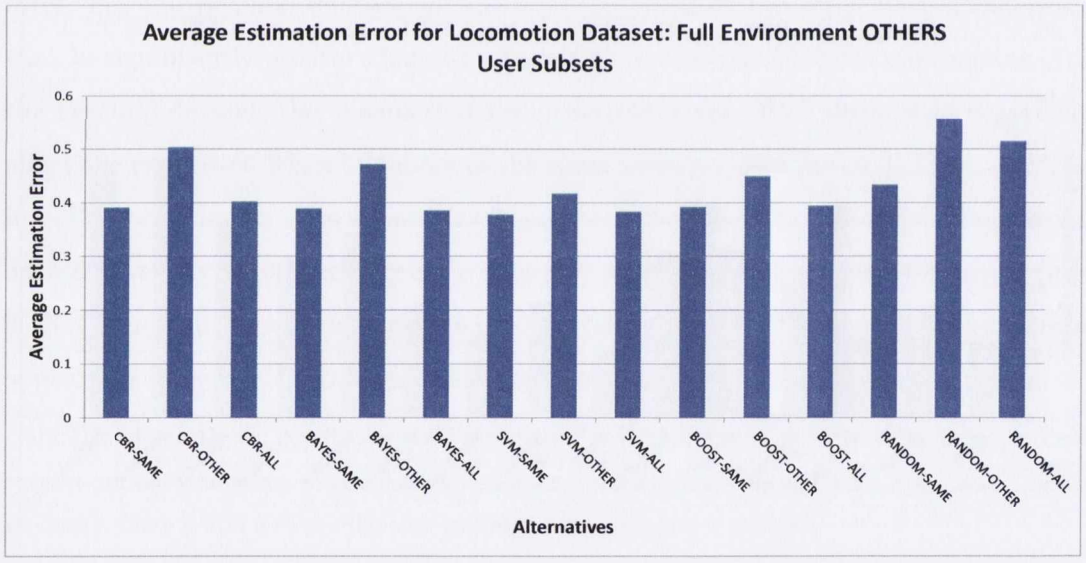

Figure C.14: Full environment user subsets for the Locomotion dataset.

C.5 Shared-Environment Attributes

The third threat to the validity of the single user conversion procedure is that it treats runs of the same user as though they were from different users (see Section 3.2.4.1). For the Nokia and Helsinki datasets, an argument against the severity of this threat

is that all the context attributes are non-static and most are shared-environment attributes that have more to do with where the user is rather than who the user is. In order to empirically explore the effects on accuracy due to the attributes that do sense the shared-environment, the evaluations of the thesis are run again without these non-shared-environment attributes. If the non-shared-environment attributes are shown to have little effect on the evaluation conclusions, this would help the validity of the dataset conversion procedure by showing that the non-shared-environment attributes do not bias the results in favor of the OTHERS alternative even though they contain runs of the same user.

For the Nokia dataset (Figure C.15), the evaluations are run without the Device_Orientation and User_Activity attributes. The statistical tests are the same as those performed in the main thesis evaluation. The standard deviations of the SVM-HISTORY and CBR-HISTORY alternatives are more than twice the smallest standard deviation in the dataset (.0597 for RANDOM-OTHERS). This means that the non-parametric tests were used for this analysis, unlike the evaluation with the full complement of attributes where the parametric tests can be used (Section 5.1.1). The statistical results for the shared-environment attribute tests are the same as the non-parametric results in Table 5.5. This means that three of the comparisons are no longer statistically significant (OTHERS vs. BOTH for BAYES and BOOST, and BOTH vs. HISTORY for BAYES). However, the trends and the conclusions drawn are the same as the evaluation with the full complement of attributes. This means that even with the potentially more user-specific, non-shared-environment attributes removed, the results remain the same.

For the Helsinki dataset (Figure C.16), the evaluations are run without the Device attributes (DisplayDown, DisplayUp, AntennaDown, AntennaUp, SidewaysRight, SidewaysLeft, Stable, and AtHand) and UserAction attributes (Walking, WalkingFast, and Running). While the values of the average estimation error are quite different for all alternatives (resulting from removing almost half of the attributes), all the comparisons are the same except for three that now show a statistical difference where there was none in the comparisons with the full complement of attributes. Namely, HISTORY is now better than OTHERS for BAYES and BOOST, and HISTORY is better than BOTH for BOOST.

Removing only the Device attributes keeps approximately the same trends but minimally increases the error for all of the alternatives, as the estimation error for most of the Device attributes (Figure C.3) is lower than each alternative's average when all attributes are included (Figure 5.7). Removing the UserAction attributes significantly lessens the error for all alternatives and does so for the HISTORY alternatives more than the OTHERS alternatives, relatively. This effect could be predicted by observing the relative difference between the OTHERS and HISTORY alternatives for the Walking and WalkingFast attribute errors in Figure C.4. That is, the estimation error for most of the individual UserAction attributes (Figure C.3) is higher than each alternative's average when all attributes are included, and the OTHERS alternatives are much more accurate than the HISTORY alternatives.

This all means that the removal of the UserAction attributes are the key to the differences between the results for BOOST and BAYES when all attributes are included and when the non-shared-environment attributes are removed. This is due to the fact that these attributes are tied to the specific scenario and the specific task at a point in time in the run, i.e., the speed at which the user is currently walking. This is clear from the difference of the OTHERS and HISTORY alternatives for the Walking and WalkingFast attributes, i.e., the accuracy depends on what the user is currently doing rather than what the user has done in the past. As further evidence, the scenario experiments in Appendix C.4 show that the accuracy of the Walking and WalkingFast attributes are closely tied to the scenario being run, as there is a great difference between in-scenario (IN) and total-scenario (TOTAL) in Figure C.13. This leads to the conclusion that there is strong evidence that the differences in results for the BOOST and BAYES alternatives without the non-shared-environment attributes is due more to the fact that the UserAction attributes are scenario-specific through time rather than that these attributes are somehow biased by treating runs of the same user as runs from different users.

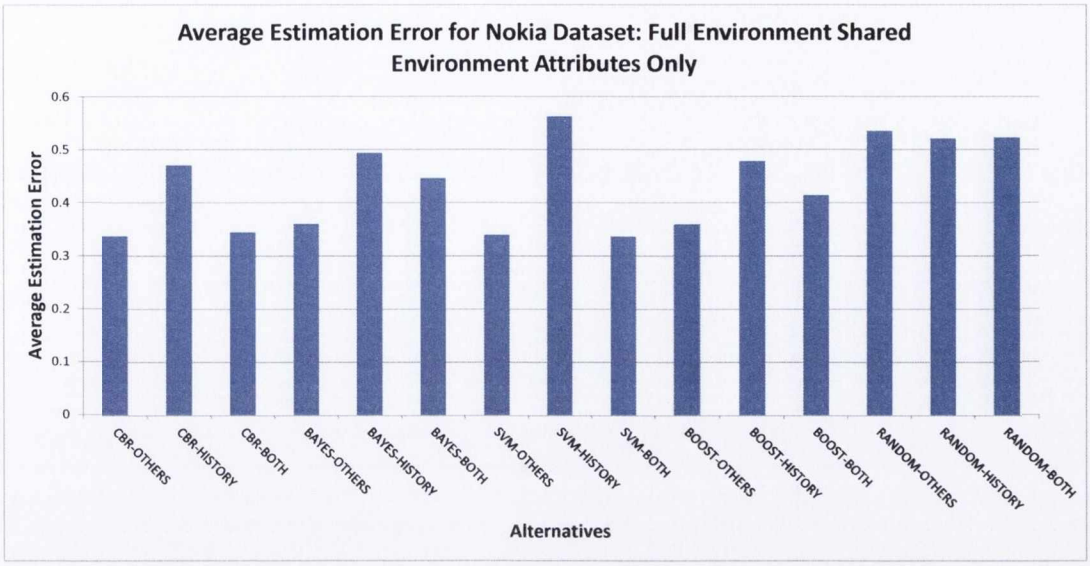

Figure C.15: Full environment with only shared-environment attributes for the Nokia dataset.

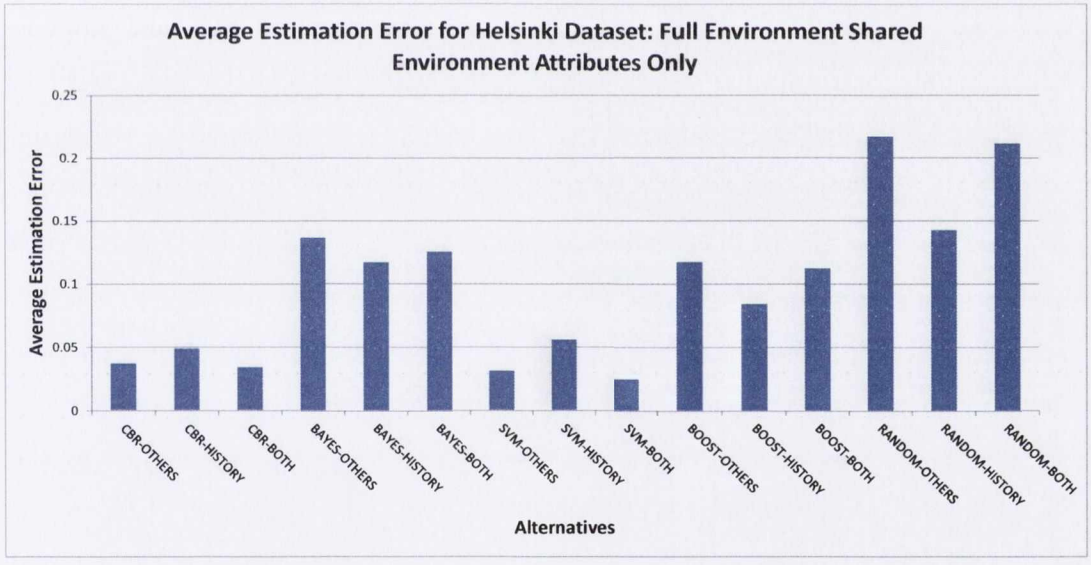

Figure C.16: Full environment with only shared-environment attributes for the Helsinki dataset.

C.6 Removing the Most Similar Runs

As discussed in Section 3.2.4.1, the potential biases that the single-user conversion

procedure introduce are ameliorated by the scenarios and attributes of the selected datasets, i.e., many of the datasets' attributes measure the shared environment and are not user-specific (see Appendix C.5). While this diminishes the bias from treating runs from the same user as runs from different users, there could still be user-specific similarities in attributes between runs of the same user. For example, the specific manner in which a user carries a device could muffle the microphone and could decrease the measured audio level relative to other users' microphones that are not obstructed.

A means to potentially relieve the same user bias even further is to remove the runs that are most similar to the current run from the set of other users. The removal of the most similar runs will necessarily cause the accuracy of the set of other users to decrease, as the learning methods depend on similar relationships between attributes to train themselves. However, if it can be shown that the overall trends of the evaluation remain unchanged, even when the most similar runs are removed, then the effect of the bias can be considered relatively small.

For the Locomotion dataset, the different users are labeled, and they have already been removed from the evaluation of this thesis. For the Nokia (one user) and Helsinki (two indistinguishable users) datasets, the most similar runs are found by clustering the runs of each scenario and removing any within-cluster runs when testing the current user's run. For the Helsinki dataset especially, it is hypothesized that the runs removed from within-cluster will tend to be those from the same user.

The datasets all contain multi-attribute time series data. In Fu's review on time series data mining [44], mining multi-attribute time series data is considered a state-of-the-art time series data mining issue. There are several approaches such as the interaction k-means that clusters based on similarity of models representing dependencies among attributes [107] and one that clusters based on the similarity of principal components derived from the data rather than the actual data [124].

The clustering method chosen for this experiment is a combination of the work of Wang and McGreavy [140] and the X-means clustering method [106]. The X-means clustering method was chosen as it is a performance improvement on K-means clustering that automatically determines the number of clusters. The specific X-means implementation is version 1.0.1 using the Weka machine learning toolkit (Version 3.7.3). All

the options used for the X-means implementation are from the default configuration. The X-means clustering method is not built for time series, so Wang and McGreavy's conversion of the multi-attribute time series data to single instance multi-attribute is used. The conversion unstacks each run represented by a matrix of m attributes by t time steps and places them end-to-end in a single vector of length $m \times t$. This clustering method in which the attributes are only compared to those of the same timestamp was chosen over clustering methods mentioned previously [124, 107] that detect patterns across time because of the single snapshot nature of the evaluations of this thesis (see Section 3.2.4.1). The selected clustering method aggregates the runs whose attributes are more synchronized with their users and scenarios.

The clustering method implementation details are as follows:

- *Per-scenario clustering*: For each scenario, the runs of that scenario are clustered so that the experiments can focus on the in-scenario differences rather than forming clusters based on the different scenarios.

- *History not considered*: When matching the runs, the portion of the run marked for history is not considered. The focus is on the portion of the runs made up of the instances that make up the set of other users (see the Training Sets discussion in Section 4.1.1).

- *No start time randomization*: As shown by comparing the average estimation error graphs for the full environment in Appendix C.2 and Section 5.1, the randomized start time does not affect the accuracy of the results. Eliminating the start time randomization might put some bias into the clustering, but the minimal differences in the results with and without the randomization point to this bias being very small. Experiments without the randomized start time have the added benefits of allowing them to focus on the variations due to clustering and allowing the most similar runs to remain the same throughout the clustering repetitions.

- *Timestamp intervals*: The instances being compared for the clustering method are regularly sampled in the same manner as they are for the thesis evaluations, i.e., the Nokia dataset interval is every 10 time steps and the Helsinki dataset interval is every time step.

- *Clustering repetitions*: The clustering method is run several times with a different seed. For each cluster assignment, the full environment test is run and the estimation error is recorded. The estimation error is then averaged for each cluster assignment and that average is reported in the graphs of this section. The variation of results between clustering repetitions was not seen to be very large, so the number of clustering repetitions was selected to be five.

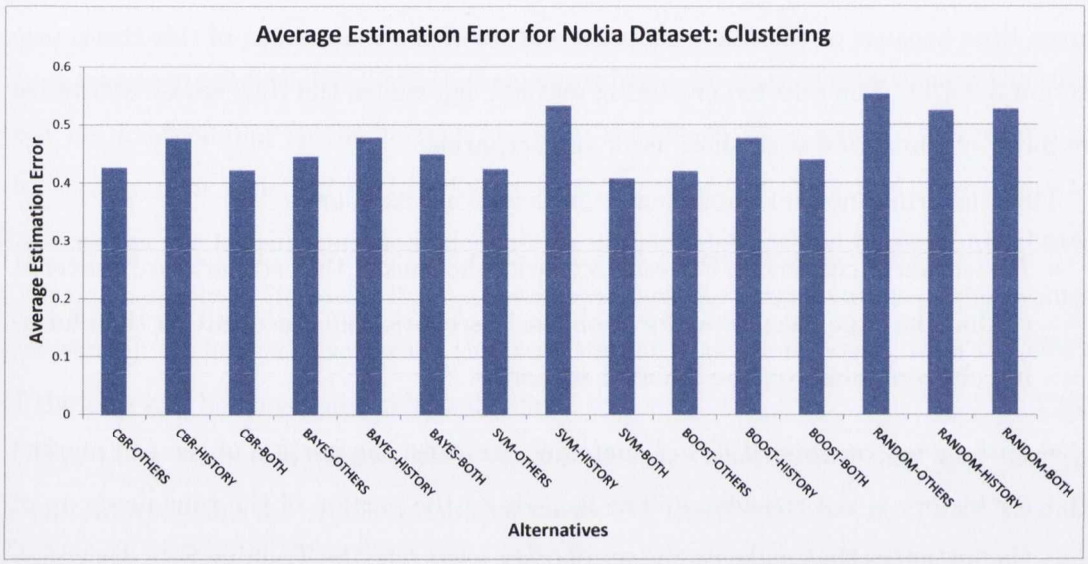

Figure C.17: Full environment clustering for the Nokia dataset.

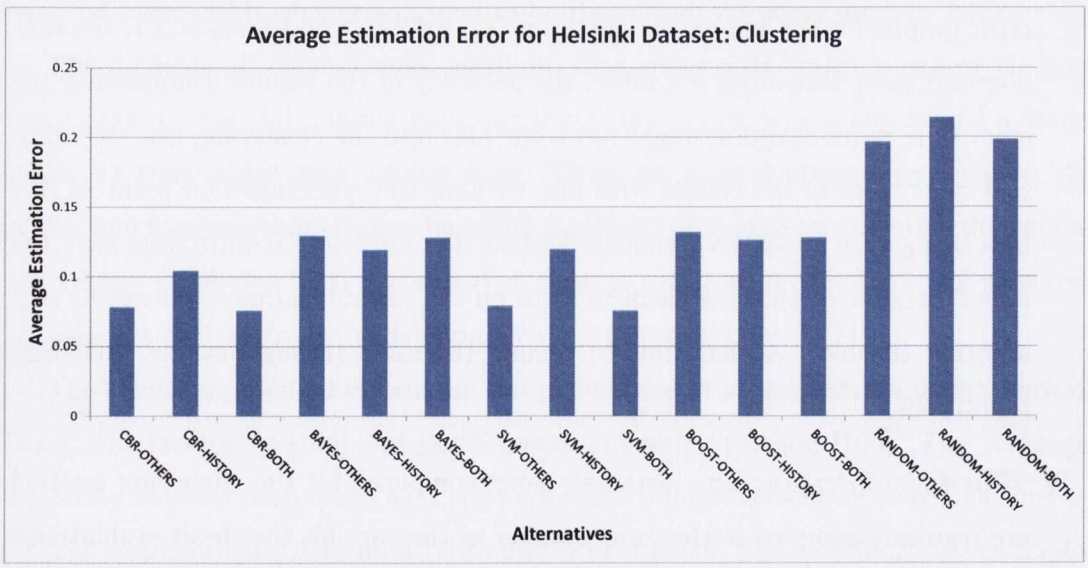

Figure C.18: Full environment clustering for the Helsinki dataset.

Figure C.17 and Figure C.18 show the average results after the clustering was repeated five times. The OTHERS and BOTH runs are slightly worse for both datasets, but the trends are the same as those without clustering (see Section 5.1). The statistical analysis carried out on these experiments is the same as was used for the full environment tests in the main evaluation of the thesis (Section 5.1).

For the Nokia dataset, the statistical results for the clustering experiments are the same as the full environment results in Table 5.5 with four differences: Four of the comparisons are no longer statistically significant (OTHERS vs. BOTH and BOTH vs. HISTORY for both BAYES and BOOST). However, the trends and the conclusions drawn are the same as the evaluation with the similar runs included. This means that even with the most similar runs removed from each scenario, the results and conclusions remain the same.

For the Helsinki dataset, the statistical results for the clustering experiments are exactly the same as the full environment results in Table 5.5. This means that even with the most similar runs removed from each scenario, the results and conclusions remain the same.

The fact that removing the most similar runs does not affect the results leads to a reduction in severity of the bias created by treating the runs of the same user as runs of different users. Further analysis of this experiment is carried out in the Dataset Validity Analysis section (Section 5.7).

C.7 Graphs with Added Noise

This section presents the evaluations with noise added to the OTHERS set in order to measure the effect on accuracy for each learning method. It is hypothesized that adding noise to the OTHERS set is a way to remove some of the advantages given to the OTHERS set that exist because of using runs of the same scenarios and same users. Adding noise to actual data is seen to be a much better option than fabricating the underlying data distributions. It should be noted that the noise also affects the subset of the BOTH set consisting of instances of the OTHERS set, and that the HISTORY set remains unaffected.

For this experiment, noise was added to each attribute's value by sampling from a

Gaussian distribution with the mean being the value of the attribute and the deviation being half of the standard deviation of the value of an attribute over the entire dataset, i.e., half of the number as was used for the Gelman transformation in Section 4.2.2. This calculation was also carried out for nominal values and then the value was rounded to the nearest integer. If the value went beyond the maximum or minimum of the attribute, the value was assigned to the maximum or minimum, respectively. Each test was repeated five times in order remove any random sampling bias.[2]

All of the OTHERS alternatives show at least a small increase in error (Figure C.19 and Figure C.20). Most learning methods follow the same pattern as that described in the analysis of the main evaluation. There are some interesting exceptions, however.

For the Nokia dataset (Figure C.19), all the alternatives show only a modest increase in error relative to the CBR-OTHERS and CBR-BOTH alternatives. This is hypothesized to be the case as the relationship of the predictive attributes is also affected by the noise. This is less important for the discretized methods where the values must only fall inside intervals, whereas CBR attempts to find the most similar for each attribute. CBR-BOTH also appears to be a little bit more accurate than the best instance set, i.e., CBR-HISTORY. This would appear to contradict the conclusion that CBR-BOTH mirrors the best instance set for CBR. The behavior appears to be consistent, however, because while the OTHERS set is worse overall, it can occasionally be better due to the wild variation of the OTHERS set. When the OTHERS set does have the better value, it is selected by the CBR method.

The Helsinki dataset is consistent with the evaluation conclusions and the Nokia results. The OTHERS and BOTH instance sets are worse and CBR is much worse, relatively. The one exception is that SVM-BOTH does not appear to be equivalent to the best instance set for SVM. This was seen to be the case for the Helsinki dataset in the scenario experiments in Appendix C.4 as well. It appears that while SVM behaves similarly to CBR most of the time, there are some occassions where its behavior is closer to BAYES and BOOST.

[2]In addition, the random start time was not implemented here in order to focus on the variation due to the added noise. See Appendix C.2 to see a discussion of why the random start time is not necessary.

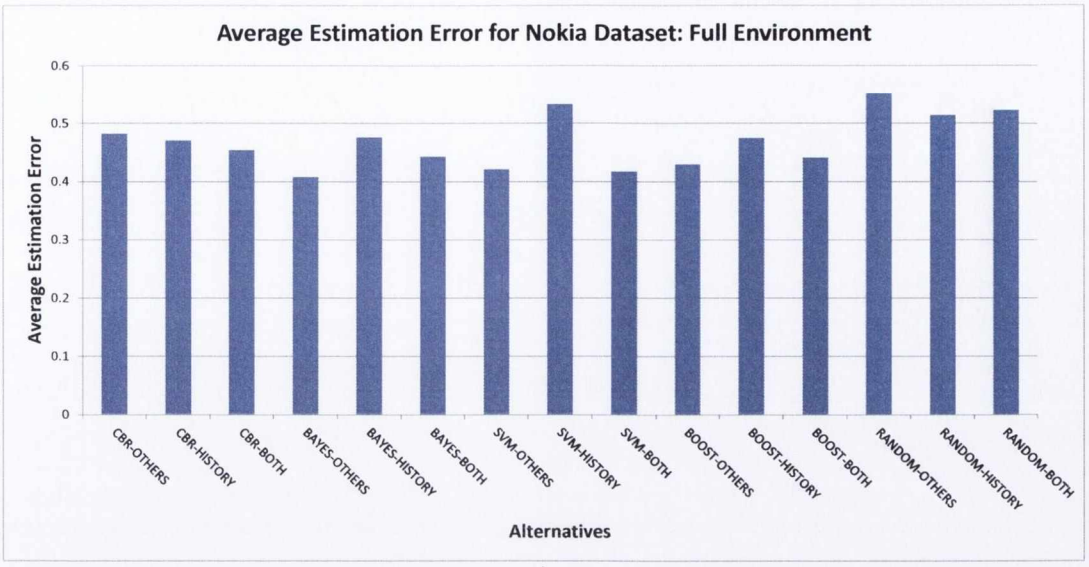

Figure C.19: Full environment with noise added to OTHERS for the Nokia dataset.

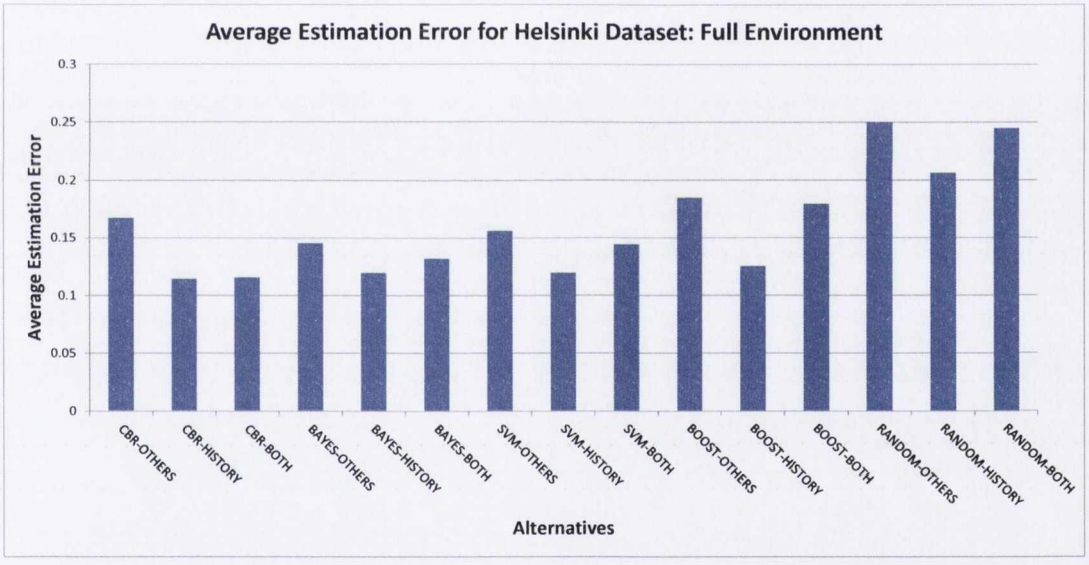

Figure C.20: Full environment with noise added to OTHERS for the Helsinki dataset.

While the observations of the learning methods on the dataset with noise added to the OTHERS set are interesting, there are three major problems with this experiment. The first is that the parameters appear to be arbitrary. Gaussian distribution might not be acceptable for representing the noise for all the attribute types. The parameter of half the standard deviation, while hypothesized to be a worst-case deviation for users performing the same task in the same environment, is also an arbitrary number

and might not reflect actual differences between different users performing the same task.

The second difficulty is that the three problems with simulations that the single-user conversion procedure solved by using runs from actual pervasive environments now exist in the dataset. The impetus behind this experiment is that adding noise to actual data is a much better option than fabricating the underlying data distributions as well. However, adding random noise removes the realistic relationships between attributes, the true manner in which attributes change over time, and the realistic modeling of sensor uncertainty (see Section 1.1.3). This is especially apparent in the relatively very poor results of the CBR learning method that relies on the relationships between attributes to be regular.

The third and biggest problem with the experiment is that adding noise is essentially only adding an arbitrary penalty to instance sets that include the OTHERS set. As seen by the second problem, it does this while making the individual runs less realistic. In fact, a substantial penalty already exists in the single-user conversion procedure without having to add in any noise. This is due to the shifting the time of the runs threat where the shared-environment attributes are seen to be a worst-case scenario for the OTHERS set. A good estimate of the severity of this penalty is shown in the shared-environment attribute experiments in Appendix C.5.

Due to these three problems, the importance of this experiment is weighted less than the others that keep the dataset realistic while attempting to judge how much bias the OTHERS set exhibits due to the single-user transformation procedure.

Bibliography

[1] Agnar Aamodt. Knowledge-intensive case-based reasoning in CREEK. In P. Funk and P. A. Gonzàlez-Calero, editors, *Proc. of the European Conference on Case-Based Reasoning, ECCBR'04*, volume 3155 of *Lecture Notes in Artificial Intelligence*, pages 1–15. Springer, 2004.

[2] Agnar Aamodt and Enric Plaza. Case-based reasoning: Foundational issues, methodological variations, and system approaches. *Artificial Intelligence Communications 7*, 1:39–52, 1994.

[3] Russell Lincoln Ackoff. From data to wisdom. *Journal of Applied Systems Analysis*, 16:3–9, 1989.

[4] Gediminas Adomavicius and Alexander Tuzhilin. Toward the next generation of recommender systems: A survey of the state-of-the-art and possible extensions. *IEEE Trans. on Knowl. and Data Eng.*, 17:734–749, June 2005.

[5] Gediminas Adomavicius and Alexander Tuzhilin. Context-aware recommender systems. In Francesco Ricci, Lior Rokach, Bracha Shapira, and Paul B. Kantor, editors, *Recommender Systems Handbook*, pages 217–253. Springer, 2011.

[6] Fahd Albinali, Nigel Davies, and Adrian Friday. Structural learning of activities from sparse datasets. In *PERCOM '07: Proceedings of the Fifth IEEE International Conference on Pervasive Computing and Communications*, pages 221–228, Washington, DC, USA, 2007. IEEE Computer Society.

[7] Ethem Alpaydin. *Introduction to machine learning.* Adaptive Computation and Machine Learning. MIT Press, 2010.

[8] Arthur Asuncion and David J. Newman. UCI machine learning repository, 2007.

[9] Marko Balabanović and Yoav Shoham. Fab: content-based, collaborative recommendation. *Commun. ACM*, 40:66–72, March 1997.

[10] Guruduth Banavar and Abraham Bernstein. Software infrastructure and design challenges for ubiquitous computing applications. *Commun. ACM*, 45(12):92–96, 2002.

[11] Agathe Battestini and Adrian Flanagan. Modelling and simulating context data in a mobile environment. In *Proceedings of the Workshop on Context Awareness for Proactive Systems (CAPS 2005)*, 2005.

[12] Leo Breiman. Bagging predictors. *Mach. Learn.*, 24:123–140, August 1996.

[13] Leo Breiman, J. H. Friedman, R. A. Olshen, and C. J. Stone. *Classification and Regression Trees.* Statistics/Probability Series. Wadsworth Publishing Company, Belmont, California, U.S.A., 1984.

[14] Lionel Briand, Khaled E. Emam, and Sandro Morasca. On the application of measurement theory in software engineering. *Empirical Software Engineering*, 1(1):61–88, January 1996.

[15] Gavin Brown, Jeremey Wyatt, Rachel Harris, and Xin Yao. Diversity creation methods: A survey and categorisation. *Journal of Information Fusion*, 6(1):5–20, March 2005.

[16] Thomas Buchholz and Michael Schiffers. Quality of context: What it is and why we need it. In *In Proceedings of the 10th Workshop of the OpenView University Association: OVUA03*, 2003.

[17] Robin Burke. Hybrid recommender systems: Survey and experiments. *User Modeling and User-Adapted Interaction*, 12:331–370, November 2002.

[18] Robin Burke. Hybrid web recommender systems. In Peter Brusilovsky, Alfred Kobsa, and Wolfgang Nejdl, editors, *The adaptive web*, pages 377–408. Springer-Verlag, Berlin, Heidelberg, 2007.

[19] Hee Byun and Keith Cheverst. Utilizing context history to provide dynamic adaptations. *Applied Artificial Intelligence*, 18(6):533–548, 2004.

[20] Andrew T. Campbell, Shane B. Eisenman, Nicholas D. Lane, Emiliano Miluzzo, Ronald A. Peterson, Hong Lu, Xiao Zheng, Mirco Musolesi, Kristóf Fodor, and Gahng-Seop Ahn. The rise of people-centric sensing. *IEEE Internet Computing*, 12(4):12–21, 2008.

[21] Bin Cao, Sinno Jialin Pan, Yu Zhang, Dit-Yan Yeung, and Qiang Yang. Adaptive transfer learning. In Maria Fox and David Poole, editors, *Proceedings of the Twenty-Fourth AAAI Conference on Artificial Intelligence, AAAI 2010, Atlanta, Georgia, USA, July 11-15, 2010.* AAAI Press, 2010.

[22] Rich Caruana. Multitask learning. *Machine Learning*, 28(1):41–75, 1997.

[23] Jason Catlett. On changing continuous attributes into ordered discrete attributes. In *EWSL-91: Proceedings of the European working session on learning on Machine learning*, pages 164–178, New York, NY, USA, 1991. Springer-Verlag New York, Inc.

[24] Gert Cauwenberghs and Tomaso Poggio. Incremental and Decremental Support Vector Machine Learning. In *Advances in Neural Information Processing Systems*, volume 13, pages 409–415. Bradford Books, 2000.

[25] Chih-Chung Chang and Chih-Jen Lin. Libsvm: A library for support vector machines. *ACM Trans. Intell. Syst. Technol.*, 2:27:1–27:27, May 2011.

[26] Maria Chantzara and Miltiades E. Anagnostou. Designing the context matching engine for evaluating and selecting context information sources. In Thomas Roth-Berghofer, Stefan Schulz, and David B. Leake, editors, *Modeling and Retrieval of Context*, volume 3946 of *Lecture Notes in Computer Science*, pages 101–117. Springer, 2005.

[27] Maria Chantzara, Miltiades E. Anagnostou, and Efstathios D. Sykas. Designing a quality-aware discovery mechanism for acquiring context information. In *AINA (1)*, pages 211–216, 2006.

[28] Weiwei Cheng and Eyke Hüllermeier. Learning similarity functions from qualitative feedback. In *ECCBR '08: Proceedings of the 9th European conference*

on Advances in Case-Based Reasoning, pages 120–134, Berlin, Heidelberg, 2008. Springer-Verlag.

[29] Hamed Chok and Le Gruenwald. Spatio-temporal association rule mining framework for real-time sensor network applications. In *Proceeding of the 18th ACM conference on Information and knowledge management*, CIKM '09, pages 1761–1764, New York, NY, USA, 2009. ACM.

[30] W. J. Conover. *Practical Nonparametric Statistics*. John Wiley & Sons, December 1999.

[31] Thomas D. Cook and Donald T. Campbell. *Quasi-Experimentation: Design and Analysis Issues for Field Settings*. Houghton Mifflin Company, 1979.

[32] Wenyuan Dai, Qiang Yang, Gui-Rong Xue, and Yong Yu. Boosting for transfer learning. In Zoubin Ghahramani, editor, *Machine Learning, Proceedings of the Twenty-Fourth International Conference (ICML 2007), Corvallis, Oregon, USA, June 20-24, 2007*, volume 227 of *ACM International Conference Proceeding Series*, pages 193–200. ACM, 2007.

[33] Ramon López de Mántaras, David McSherry, Derek G. Bridge, David B. Leake, Barry Smyth, Susan Craw, Boi Faltings, Mary Lou Maher, Michael T. Cox, Kenneth D. Forbus, Mark T. Keane, Agnar Aamodt, and Ian D. Watson. Retrieval, reuse, revision and retention in case-based reasoning. *Knowledge Eng. Review*, 20(3):215–240, 2005.

[34] Anind K. Dey. Understanding and using context. *Personal Ubiquitous Comput.*, 5(1):4–7, 2001.

[35] Anind K. Dey, Daniel Salber, and Gregory D. Abowd. A conceptual framework and a toolkit for supporting the rapid prototyping of context-aware applications. *Human-Computer Interaction (HCI) Journal*, 16 (2-4):97–16, 2001.

[36] Thomas G. Dietterich. Ensemble methods in machine learning. In *Proceedings of the First International Workshop on Multiple Classifier Systems*, MCS '00, pages 1–15, London, UK, 2000. Springer-Verlag.

[37] Richard O. Duda, Peter E. Hart, and David G. Stork. *Pattern Classification (2nd Edition)*. Wiley-Interscience, 2nd edition, November 2000.

[38] Robert P. W. Duin and David M. J. Tax. Experiments with classifier combining rules. In *Proceedings of the First International Workshop on Multiple Classifier Systems*, MCS '00, pages 16–29, London, UK, 2000. Springer-Verlag.

[39] Nathan Eagle and Alex (Sandy) Pentland. Reality mining: sensing complex social systems. *Personal Ubiquitous Comput.*, 10(4):255–268, 2006.

[40] Jesús Favela, Monica Tentori, Luís A. Castro, Víctor M. González, Elisa B. Moran, and Ana I. Martínez-Garcia. Activity recognition for context-aware hospital applications: Issues and opportunities for the deployment of pervasive networks. *MONET*, 12(2-3):155–171, 2007.

[41] John A. Flanagan, David Murphy, and Jussi Kaasinen. A nokia context recording database with synchronized user interaction. In *Benchmarks and a Database for Context Recognition: Workshop Proceedings Pervasive 2004*, Linz/Vienna, Austria, April 18-23 2004.

[42] Eibe Frank, Leonard E. Trigg, Geoffrey Holmes, and Ian H. Witten. Naive bayes for regression (technical note). *Machine Learning*, 41(1):5–25, 2000.

[43] Yoav Freund and Robert E. Schapire. Experiments with a new boosting algorithm. In *ICML*, pages 148–156, 1996.

[44] Tak-chung Fu. A review on time series data mining. *Eng. Appl. Artif. Intell.*, 24:164–181, February 2011.

[45] Johannes Fürnkranz and Eyke Hüllermeier. Preference learning: An introduction. In Johannes Fürnkranz and Eyke Hüllermeier, editors, *Preference Learning*, pages 1–17. Springer-Verlag, 2010.

[46] Pedro J. Garcia-Laencina, Jose-Luis Sancho-Gomez, and Anibal R. Figueiras-Vidal. Pattern classification with missing data: a review. *Neural Comput. Appl.*, 19:263–282, March 2010.

[47] Andrew Gelman. Scaling regression inputs by dividing by two standard deviations. *Statistics in Medicine*, 27(15):2865–2873, July 2008.

[48] Marco de Gemmis, Leo Iaquinta, Pasquale Lops, Cataldo Musto, Fedelucio Narducci, and Giovanni Semeraro. Learning preference models in recommender systems. In Johannes Fürnkranz and Eyke Hüllermeier, editors, *Preference Learning*, pages 387–407. Springer-Verlag, 2010.

[49] David Goldberg, David Nichols, Brian M. Oki, and Douglas Terry. Using collaborative filtering to weave an information tapestry. *Commun. ACM*, 35:61–70, December 1992.

[50] Le Gruenwald, Hanqing Yang, Md. Shiblee Sadik, and Rahul Shukla. Using data mining to handle missing data in multi-hop sensor network applications. In *Proceedings of the Ninth ACM International Workshop on Data Engineering for Wireless and Mobile Access*, MobiDE '10, pages 9–16, New York, NY, USA, 2010. ACM.

[51] Egon G. Guba and Yvonna S. Lincoln. *Effective evaluation.* Jossey-Bass Publishers, San Francisco :, 1st ed. edition, 1981.

[52] Karen Henricksen and Jadwiga Indulska. Modelling and using imperfect context information. In *PERCOMW '04: Proceedings of the Second IEEE Annual Conference on Pervasive Computing and Communications Workshops*, page 33, Washington, DC, USA, 2004. IEEE Computer Society.

[53] Karen Henricksen and Jadwiga Indulska. Personalising context-aware applications. In *IN OTM WORKSHOP ON CONTEXTAWARE MOBILE SYSTEMS*, pages 122–131. Springer-Verlag, 2005.

[54] Chun-Nan Hsu, Hung-Ju Huang, and Tzu-Tsung Wong. Why discretization works for naive bayesian classifiers. In *ICML '00: Proceedings of the Seventeenth International Conference on Machine Learning*, pages 399–406, San Francisco, CA, USA, 2000. Morgan Kaufmann Publishers Inc.

[55] Markus C. Huebscher and Julie A. McCann. Adaptive middleware for context-aware applications in smart-homes. In *MPAC '04: Proceedings of the 2nd work-*

shop on Middleware for pervasive and ad-hoc computing, pages 111–116, New York, NY, USA, 2004. ACM.

[56] Markus C. Huebscher and Julie A. McCann. Simulation model for self-adaptive applications in pervasive computing. In *DEXA '04: Proceedings of the Database and Expert Systems Applications, 15th International Workshop*, pages 694–698, Washington, DC, USA, 2004. IEEE Computer Society.

[57] Markus C. Huebscher and Julie A. McCann. An adaptive middleware framework for context-aware applications. *Personal and Ubiquitous Computing*, 10(1):12–20, 2006.

[58] Pertti Huuskonen, Jani Mäntyjärvi, and Ville Könönen. Collaborative context recognition for mobile devices. In Hideyuki Nakashima, Hamid Aghajan, and Juan Carlos Augusto, editors, *Handbook of Ambient Intelligence and Smart Environments*, pages 257–280. Springer US, 2010.

[59] Stephen S. Intille, Kent Larson, Emmanuel Munguia Tapia, Jennifer Beaudin, Pallavi Kaushik, Jason Nawyn, and Randy Rockinson. Using a live-in laboratory for ubiquitous computing research. In Kenneth P. Fishkin, Bernt Schiele, Paddy Nixon, and Aaron J. Quigley, editors, *Pervasive*, volume 3968 of *Lecture Notes in Computer Science*, pages 349–365. Springer, 2006.

[60] Holger Junker, Paul Lukowicz, and Gerhard Tröster. User activity related data sets for context recognition. In *Benchmarks and a Database for Context Recognition: Workshop Proceedings Pervasive 2004*, Linz/Vienna, Austria, April 18-23 2004.

[61] Natalia Juristo and Ana M. Moreno. *Basics of Software Engineering Experimentation.* Kluwer Academic Publishers, 2001.

[62] A. Baki Kocaballi and Altan Kocyigit. Granular best match algorithm for context-aware computing systems. In *PERSER '06: Proceedings of the 2006 ACS/IEEE International Conference on Pervasive Services*, pages 143–149, Washington, DC, USA, 2006. IEEE Computer Society.

[63] Anders Kofod-Petersen and Agnar Aamodt. Contextualised ambient intelligence through case-based reasoning. In T. R. Roth-Berghofer, Mehmet H. Göker, and H. Altay Güvenir, editors, *Proceedings of the Eighth European Conference on Case-Based Reasoning (ECCBR 2006)*, volume 4106 of *Lecture Notes in Computer Science*, pages 211–225, Ölüdeniz, Turkey, September 2006. Springer Verlag.

[64] Anders Kofod-Petersen and Agnar Aamodt. Case-based reasoning for situation-aware ambient intelligence: A hospital ward evaluation study. In *ICCBR '09: Proceedings of the 8th International Conference on Case-Based Reasoning*, pages 450–464, Berlin, Heidelberg, 2009. Springer-Verlag.

[65] J.A. Konstan, John Riedl, Al Borchers, and Jonathan Herlocker. Recommender systems: A grouplens perspective. In *Recommender Systems. Papers from 1998 Workshop. Technical Report WS-98-08*, pages 60–64. AAAI Press, 1998.

[66] Panu Korpipaa, Jani Mantyjarvi, Juha Kela, Heikki Keranen, and Esko-Juhani Malm. Managing context information in mobile devices. *IEEE Pervasive Computing*, 2(3):42–51, 2003.

[67] Kristof Van Laerhoven and Hans-Werner Gellersen. Spine versus porcupine: A study in distributed wearable activity recognition. In *ISWC '04: Proceedings of the Eighth International Symposium on Wearable Computers*, pages 142–149, Washington, DC, USA, 2004. IEEE Computer Society.

[68] Nicholas D. Lane, Hong Lu, Shane B. Eisenman, and Andrew T. Campbell. Cooperative techniques supporting sensor-based people-centric inferencing. In *Pervasive '08: Proceedings of the 6th International Conference on Pervasive Computing*, pages 75–92, Berlin, Heidelberg, 2008. Springer-Verlag.

[69] Averill Law. *Simulation Modeling and Analysis (McGraw-Hill Series in Industrial Engineering and Management)*. McGraw-Hill Science/Engineering/Math, 2006.

[70] Lucian Leahu, Phoebe Sengers, and Michael Mateas. Interactionist ai and the promise of ubicomp, or, how to put your box in the world without putting the

world in your box. In *UbiComp '08: Proceedings of the 10th international conference on Ubiquitous computing*, pages 134–143, New York, NY, USA, 2008. ACM.

[71] David B. Leake and Raja Sooriamurthi. When two case bases are better than one: Exploiting multiple case bases. In *ICCBR '01: Proceedings of the 4th International Conference on Case-Based Reasoning*, pages 321–335, London, UK, 2001. Springer-Verlag.

[72] David B. Leake and Raja Sooriamurthi. Automatically selecting strategies for multi-case-base reasoning. In Susan Craw and Alun D. Preece, editors, *Advances in Case-Based Reasoning, 6th European Conference, ECCBR 2002 Aberdeen, Scotland, UK, September 4-7, 2002, Proceedings*, volume 2416 of *Lecture Notes in Computer Science*, pages 204–233. Springer, 2002.

[73] David B. Leake and Raja Sooriamurthi. Managing multiple case bases: Dimensions and issues. In *Proceedings of the Fifteenth International Florida Artificial Intelligence Research Society Conference*, pages 106–110. AAAI Press, 2002.

[74] David B. Leake and Raja Sooriamurthi. Dispatching cases versus merging case-bases: When MCBR matters. In Ingrid Russell and Susan M. Haller, editors, *FLAIRS Conference*, pages 129–133. AAAI Press, 2003.

[75] David B. Leake and Raja Sooriamurthi. Case dispatching versus case-base merging: when mcbr matters. *International Journal on Artificial Intelligence Tools*, 13(1):237–254, 2004.

[76] Marie-Jeanne Lesot, Maria Rifqi, and H. Benhadda. Similarity measures for binary and numerical data: a survey. *Int. J. Knowl. Eng. Soft Data Paradigm.*, 1(1):63–84, 2008.

[77] Greg Linden, Brent Smith, and Jeremy York. Amazon.com recommendations: Item-to-item collaborative filtering. *IEEE Internet Computing*, 7:76–80, 2003.

[78] Tinghuai Ma, Yong-Deak Kim, Qiang Ma, Meili Tang, and Weican Zhou. Context-aware implementation based on CBR for smart home. In *IEEE In-*

ternational Conference on Wireless And Mobile Computing, Networking And Communications, volume 4, pages 112 – 115, 2005.

[79] Takuya Maekawa, Yutaka Yanagisawa, Yasue Kishino, Katsuhiko Ishiguro, Koji Kamei, Yasushi Sakurai, and Takeshi Okadome. Object-based activity recognition with heterogeneous sensors on wrist. In Patrik Floreen, Antonio Krueger, and Mirjana Spasojevic, editors, *Pervasive Computing*, volume 6030 of *Lecture Notes in Computer Science*, pages 246–264. Springer Berlin / Heidelberg, 2010.

[80] P. C. Mahalanobis. On the generalised distance in statistics. In *Proceedings National Institute of Science, India*, volume 2, pages 49–55, April 1936.

[81] Jani Mäntyjärvi, Johan Himberg, and Pertti Huuskonen. Collaborative context recognition for handheld devices. In *PERCOM '03: Proceedings of the First IEEE International Conference on Pervasive Computing and Communications*, page 161, Washington, DC, USA, 2003. IEEE Computer Society.

[82] Lorraine McGinty and Barry Smyth. Personalized route planning: A case-based approach. In *EWCBR '00: Proceedings of the 5th European Workshop on Advances in Case-Based Reasoning*, pages 431–442, London, UK, 2000. Springer-Verlag.

[83] Lorraine McGinty and Barry Smyth. Collaborative case-based reasoning: Applications in personalised route planning. In David W. Aha and Ian Watson, editors, *4th International Conference on Case-Based Reasoning, ICCBR 2001*, volume 2080 of *Lecture Notes in Computer Science*, pages 362–376, 2001.

[84] Lorraine McGinty and Barry Smyth. Shared experiences in personalized route planning. In *Proceedings of the Fifteenth International Florida Artificial Intelligence Research Society Conference*, pages 111–115. AAAI Press, 2002.

[85] Sharan B. Merriam. *Qualitative Research and Case Study Applications in Education: Revised and Expanded from I Case Study Research in Education/I (Jossey Bass Education Series)*. Jossey-Bass, September 1997.

[86] Emiliano Miluzzo, Nicholas D. Lane, Kristóf Fodor, Ronald Peterson, Hong Lu, Mirco Musolesi, Shane B. Eisenman, Xiao Zheng, and Andrew T. Campbel.

Sensing meets mobile social networks: the design, implementation and evaluation of the cenceme application. In *SenSys '08: Proceedings of the 6th ACM conference on Embedded network sensor systems*, pages 337–350, New York, NY, USA, 2008. ACM.

[87] Tom M. Mitchell. *Machine Learning.* McGraw-Hill, New York, 1997.

[88] J Mäntyjärvi, P Huuskonen, and J Himberg. Collaborative context determination to support mobile terminal applications. *IEEE Wireless Communications*, 9(5):39–45, 2002.

[89] Jani Mäntyjärvi, Johan Himberg, Petri Kangas, Urpo Tuomela, and Pertti Huuskonen. Sensor signal data set for exploring context recognition of mobile devices. In *Benchmarks and a Database for Context Recognition: Workshop Proceedings Pervasive 2004*, Linz/Vienna, Austria, April 18-23 2004.

[90] David S. Moore, George P. McCabe, and Bruce Craig. *Introduction to the Practice of Statistics.* Freeman, W.H. & Company, 2007.

[91] Martin Mühlenbrock, Oliver Brdiczka, Dave Snowdon, and Jean-Luc Meunier. Learning to detect user activity and availability from a variety of sensor data. In *PERCOM '04: Proceedings of the Second IEEE International Conference on Pervasive Computing and Communications (PerCom'04)*, page 13, Washington, DC, USA, 2004. IEEE Computer Society.

[92] Hongbo Ni, Xingshe Zhou, Daqing Zhang, Kejian Miao, and Yaqi Fu. Towards a task supporting system with CBR approach in smart home. In *ICOST '09: Proceedings of the 7th International Conference on Smart Homes and Health Telematics*, pages 141–149, Berlin, Heidelberg, 2009. Springer-Verlag.

[93] David M. Nichols. Implicit rating and filtering. In *In Proceedings of the Fifth DELOS Workshop on Filtering and Collaborative Filtering*, pages 31–36, 1998.

[94] Neil O'Connor. *KAFCA: Knowledge Autonomy for Reactive Context-aware Applications.* PhD thesis, Trinity College Dublin, 2010.

[95] Kenta Oku, Shinsuke Nakajima, Jun Miyazaki, and Shunsuke Uemura. Context-aware svm for context-dependent information recommendation. In *Proceedings of the 7th International Conference on Mobile Data Management*, MDM '06, pages 109–, Washington, DC, USA, 2006. IEEE Computer Society.

[96] Santi Ontañón and Enric Plaza. An argumentation-based framework for deliberation in multi-agent systems. In *ArgMAS'07: Proceedings of the 4th international conference on Argumentation in multi-agent systems*, pages 178–196. Springer-Verlag, 2008.

[97] Santiago Ontañón and Enric Plaza. Justification-based multiagent learning. Technical report, Artificial Intelligence Research Institute, 2003.

[98] Santiago Ontañón and Enric Plaza. Learning, information exchange, and joint-deliberation through argumentation in multi-agent systems. In Robert Meersman, Zahir Tari, and Pilar Herrero, editors, *On the Move to Meaningful Internet Systems: OTM 2008 Workshops*, volume 5333 of *Lecture Notes in Computer Science*, pages 150–159. Springer, 2008.

[99] Nikunj C. Oza. Online bagging and boosting. In *Systems, Man and Cybernetics, 2005 IEEE International Conference on*, volume 3, pages 2340 – 2345 Vol. 3, oct. 2005.

[100] Sinno Jialin Pan and Qiang Yang. A survey on transfer learning. *IEEE Transactions on Knowledge and Data Engineering*, 22:1345–1359, 2010.

[101] Spiros Papadimitriou, Jimeng Sun, and Christos Faloutsos. Streaming pattern discovery in multiple time-series. In *Proceedings of the 31st international conference on Very large data bases*, VLDB '05, pages 697–708. VLDB Endowment, 2005.

[102] Pravin Pawar, Aart van Halteren, and Kamran Sheikh. Enabling context-aware computing for the nomadic mobile user: A service oriented and quality driven approach. In *Proceedings of IEEE Wireless Communications and Networking Conference, 2007.WCNC 2007.*, pages 2531–2536. IEEE Communication Society, March 2007.

[103] Michael Pazzani and Daniel Billsus. Learning and revising user profiles: The identification of interesting web sites. *Machine Learning*, 27:313–331, 1997.

[104] Michael J. Pazzani. A framework for collaborative, content-based and demographic filtering. *Artif. Intell. Rev.*, 13:393–408, December 1999.

[105] Michael J. Pazzani and Daniel Billsus. The adaptive web. chapter Content-based recommendation systems, pages 325–341. Springer-Verlag, Berlin, Heidelberg, 2007.

[106] Dan Pelleg and Andrew Moore. X-means: Extending k-means with efficient estimation of the number of clusters. In *Proceedings of the Seventeenth International Conference on Machine Learning*, pages 727–734, San Francisco, 2000. Morgan Kaufmann.

[107] Claudia Plant, Afra M. Wohlschläger, and Andrew Zherdin. Interaction-based clustering of multivariate time series. In *Proceedings of the 2009 Ninth IEEE International Conference on Data Mining*, ICDM '09, pages 914–919, Washington, DC, USA, 2009. IEEE Computer Society.

[108] Enric Plaza, Josep Lluis Arcos, and Francisco Martin. *Cooperative Case-Based Reasoning*, volume 1221, pages 180–201. Spriger-Verlag, 1997.

[109] Enric Plaza and Santiago Ontañón. Ensemble case-based reasoning: Collaboration policies for multiagent cooperative CBR. In David W. Aha and Ian Watson, editors, *Case-Based Reasoning Research and Development, 4th International Conference on Case-Based Reasoning, ICCBR 2001*, volume 2080 of *Lecture Notes in Computer Science*, pages 437–451, Vancouver, BC, Canada, 2001. Springer.

[110] M. V. Nagendra Prasad, Victor R Lesser, Susan Lander, and Susan L. Retrieval and reasoning in distributed case bases. In *Journal of Visual Communication and Image Representation, Special Issue on Digital Libraries*, pages 74–87, 1995.

[111] Davy Preuveneers and Yolande Berbers. Architectural backpropagation support for managing ambiguous context in smart environments. In *4th International Conference on Universal Access in Human-Computer Interaction*, volume 4555

of *Lecture Notes in Computer Science (LNCS)*, pages 178–187. Springer-Verlag, July 2007.

[112] Parisa Rashidi and Diane J. Cook. Activity knowledge transfer in smart environments. *Pervasive and Mobile Computing*, 7(3):331 – 343, 2011.

[113] Vinny Reynolds, Vinny Cahill, and Aline Senart. Requirements for an ubiquitous computing simulation and emulation environment. In *InterSense '06: Proceedings of the first international conference on Integrated internet ad hoc and sensor networks*, page 1, New York, NY, USA, 2006. ACM.

[114] Francesco Ricci, Lior Rokach, and Bracha Shapira. Introduction to recommender systems handbook. In Francesco Ricci, Lior Rokach, Bracha Shapira, and Paul B. Kantor, editors, *Recommender Systems Handbook*, pages 1–35. 2011.

[115] Pedro Pereira Rodrigues and João Gama. Online prediction of streaming sensor data. June 2006.

[116] Michael T. Rosenstein, Zvika Marx, Leslie Pack Kaelbling, and Thomas G. Dietterich. To transfer or not to transfer. In *In NIPS'05 Workshop, Inductive Transfer: 10 Years Later*, 2005.

[117] Norman M. Sadeh, Fabien L. Gandon, and Oh Byung Kwon. Ambient intelligence: The mycampus experience. Technical report, Carnegie Mellon University, 2005.

[118] J. Ben Schafer, Dan Frankowski, Jon Herlocker, and Shilad Sen. The adaptive web. chapter Collaborative filtering recommender systems, pages 291–324. Springer-Verlag, Berlin, Heidelberg, 2007.

[119] Bill Schilit, Norman Adams, and Roy Want. Context-aware computing applications. In *IEEE Workshop on Mobile Computing Systems and Applications*, Santa Cruz, CA, US, 1994.

[120] Bernhard Schölkopf, Alex J. Smola, Robert C. Williamson, and Peter L. Bartlett. New support vector algorithms. *Neural Comput.*, 12:1207–1245, May 2000.

[121] Yanir Seroussi, Fabian Bohnert, and Ingrid Zukerman. Personalised rating prediction for new users using latent factor models. In *Proceedings of the 22nd ACM conference on Hypertext and hypermedia*, HT '11, pages 47–56, New York, NY, USA, 2011. ACM.

[122] Kamran Sheikh, Maarten Wegdam, and Marten van Sinderen. Quality-of-context and its use for protecting privacy in context aware systems. *JSW*, 3(3):83–93, 2008.

[123] Bing Shi, Xianping Tao, and Jian Lu. Rewards-based negotiation for providing context information. In *MPAC '06: Proceedings of the 4th international workshop on Middleware for Pervasive and Ad-Hoc Computing (MPAC 2006)*, page 8, New York, NY, USA, 2006. ACM.

[124] Ashish Singhal and Dale E. Seborg. Clustering multivariate time-series data. *Journal of Chemometrics*, 19(8):427–438, 2005.

[125] Barry Smyth, Peter Briggs, Maurice Coyle, and Michael P. O'Mahony. A case-based perspective on social web search. In *Proceedings of the 8th International Conference on Case-Based Reasoning: Case-Based Reasoning Research and Development*, ICCBR '09, pages 494–508, Berlin, Heidelberg, 2009. Springer-Verlag.

[126] Barry Smyth, Maurice Coyle, and Peter Briggs. Communities, collaboration, and recommender systems in personalized web search. In Francesco Ricci, Lior Rokach, Bracha Shapira, and Paul B. Kantor, editors, *Recommender Systems Handbook*, pages 579–614. 2011.

[127] Qinbao Song and Martin Shepperd. Missing data imputation techniques. *International Journal of Business Intelligence and Data Mining*, 2(3):261–291, 2007.

[128] Armin Stahl. Learning feature weights from case order feedback. In David W. Aha and Ian Watson, editors, *4th International Conference on Case-Based Reasoning, ICCBR 2001*, volume 2080 of *Lecture Notes in Computer Science*, pages 502–516, 2001.

[129] Armin Stahl and Thomas Gabel. Optimizing similarity assessment in case-based reasoning. In *AAAI*. AAAI Press, 2006.

[130] Peter Stone and Manuela Veloso. Multiagent systems: A survey from a machine learning perspective. *Autonomous Robots*, 8:345–383, 1997.

[131] Thomas Strang and Claudia Linnhoff-Popien. A context modeling survey. In *Workshop on Advanced Context Modelling, Reasoning and Management as part of UbiComp 2004 - The Sixth International Conference on Ubiquitous Computing, Nottingham/England*, 2004.

[132] Xiaoyuan Su and Taghi M. Khoshgoftaar. Collaborative filtering for multi-class data using belief nets algorithms. In *Tools with Artificial Intelligence, 2006. ICTAI '06. 18th IEEE International Conference on*, pages 497 –504, nov. 2006.

[133] Xiaoyuan Su and Taghi M. Khoshgoftaar. A survey of collaborative filtering techniques. *Adv. in Artif. Intell.*, 2009:4:2–4:2, January 2009.

[134] Emmanuel Munguia Tapia, Stephen S. Intille, and Kent Larson. Activity recognition in the home using simple and ubiquitous sensors. In *PERVASIVE, Second International Conference on Pervasive Computing*, pages 158–175, Linz / Vienna, Austria, 2004.

[135] Alexander Tuzhilin. Personalization: The State of the Art and Future Directions. In Gediminas Adomavicius and Alok Gupta, editors, *Business Computing Vol. 3 of Handbooks in Information Systems*, volume 3, chapter 1, pages 3–43. Emerald Group, 2009.

[136] Nithya Vijayakumar and Beth Plale. *Missing Event Prediction in Sensor Data Streams Using Kalman Filters*, chapter 9, pages 149–169. Taylor and Francis/CRC Press, Boca Raton, 2009.

[137] Lucian Vintan, Arpad Gellert, Jan Petzold, and Theo Ungerer. Person movement prediction using neural networks. In *In First Workshop on Modeling and Retrieval of Context*, 2004.

[138] Christian Wagner. Breaking the knowledge acquisition bottleneck through conversational management. *Information Resources Management Journal*, 19:1:70–83, 2006.

[139] S. Wang. Application of self-organising maps for data mining with incomplete data sets. *Neural Computing and Applications*, 12(1):42–48, 2003.

[140] X Z Wang and C McGreavy. Automatic classification for mining process operational data. *Industrial & Engineering Chemistry Research*, 37(6):2215–2222, 1998.

[141] Geoffrey I. Webb. Multiboosting: A technique for combining boosting and wagging. *Machine Learning*, Vol.40(No.2), 2000.

[142] M. Weiser. Ubiquitous computing. *Computer*, 26(10):71–72, 1993.

[143] D.J. Weiss. *Analysis of Variance and Functional Measurement. A Practical Guide.* Oxford University Press, New York, 2006.

[144] Gerhard Widmer and Miroslav Kubat. Learning in the presence of concept drift and hidden contexts. *Machine Learning*, 23(1):69–101, 1996.

[145] Claes Wohlin, Per Runeson, Martin Höst, Magnus C. Ohlsson, Bjöorn Regnell, and Anders Wesslén. *Experimentation in software engineering: an introduction.* Kluwer Academic Publishers, Norwell, MA, USA, 2000.

[146] Xindong Wu, Vipin Kumar, J. Ross Quinlan, Joydeep Ghosh, Qiang Yang, Hiroshi Motoda, Geoffrey J. McLachlan, Angus F. M. Ng, Bing Liu, Philip S. Yu, Zhi-Hua Zhou, Michael Steinbach, David J. Hand, and Dan Steinberg. Top 10 algorithms in data mining. *Knowl. Inf. Syst.*, 14(1):1–37, 2008.

[147] Jun Yang. Toward physical activity diary: motion recognition using simple acceleration features with mobile phones. In *IMCE '09: Proceedings of the 1st international workshop on Interactive multimedia for consumer electronics*, pages 1–10, New York, NY, USA, 2009. ACM.

[148] Ying Yang and Geoffrey I. Webb. Discretization for naive-bayes learning: managing discretization bias and variance. *Mach. Learn.*, 74(1):39–74, 2009.

[149] Ying Yang, Geoffrey I. Webb, and Xindong Wu. Discretization methods. In Oded Maimon and Lior Rokach, editors, *The Data Mining and Knowledge Discovery Handbook*, pages 113–130. Springer, 2005.

[150] Jaegeol Yim. Introducing a decision tree-based indoor positioning technique. *Expert Syst. Appl.*, 34(2):1296–1302, 2008.

[151] Vincent Wenchen Zheng, Derek Hao Hu, and Qiang Yang. Cross-domain activity recognition. In *Proceedings of the 11th international conference on Ubiquitous computing*, Ubicomp '09, pages 61–70, New York, NY, USA, 2009. ACM.

[152] Andreas Zimmermann. Context-awareness in user modelling: Requirements analysis for a case-based reasoning application. In *ICCBR*, pages 718–732, 2003.

[153] Andreas Zimmermann and Andreas Lorenz. Listen: A user-adaptive audio-augmented museum guide. *User Modeling and User-Adapted Interaction*, 18(5):389–416, 2008.