Learning user behaviours in real mobile domains

Andreas Markitanis, Domenico Corapi, Alessandra Russo, and Emil C. Lupu

Department of Computing, Imperial College London
{andreas.markitanis07,d.corapi,a.russo,e.c.lupu}@imperial.ac.uk
http://www.doc.ic.ac.uk

Abstract. With the emergence of ubiquitous computing, innovations in mobile phones are increasingly changing the way users lead their lives. To make mobile devices adaptive and able to autonomously respond to changes in user behaviours, machine learning techniques can be deployed to learn behaviour from empirical data. Learning outcomes should be rule-based enforcement policies that can manage pervasively the devices, and at the same time facilitate user validation when and if required. In this paper we demonstrate the feasibility of non-monotonic ILP in the automated task of extraction of user behaviour rules through data acquisition in the domain of mobile phones. This is a challenging task as real mobile datasets are highly noisy and unevenly distributed. We present two applications, one based on an existing dataset collected as part of the Reality Mining group, and the other generated by a mobile phone application, called *ULearn*, that we have developed to facilitate a realistic evaluation of the accuracy of the learning outcome.

Key words: Inductive Logic Programming, Abductive Logic Programming, Non-monotonic Reasoning, Mobile User Behaviour

1 Introduction

In the past few years, companies such as Apple and Samsung have really managed to develop cutting-edge mobile systems that have revolutionised the way people use mobile phones. Often, the complexity of these systems prevent the user from utilising the device to its maximum. A more pervasive approach requires such systems to be able to continuously adapt to the user's preferences and behaviour requiring as little user intervention as possible. Rules are an effective way of specifying how these systems should adapt in different context, and rule-based enforcement policies that govern system choices are increasingly becoming more popular in pervasive systems. Information about user behaviours can be collected through their actions and interaction with the device and used together with past data and background knowledge about contextual information to compute new rule-based enforcement policies and/or changes in existing ones. Such scenarios suggest the use of Inductive Logic Programming (ILP) [3] as an appropriate learning mechanism. But to our knowledge no existing ILP techniques have so far been applied to large, real data in the mobile phone domain.

In this context, to provide accurate solutions the ILP technique has to make use of heuristics to guide the search in a (potentially large) search space to minimise computation time, cater for noise in the data and for uneven distribution of data. This paper demonstrates that the non-monotonic inductive programming tool, TAL (Top-directed Abductive Learning) [2] can be appropriately customised to learn new mobile user behaviours as well as revise existing rules with approximately 80% level of accuracy. A learning framework is presented where domain knowledge about contextual information and language bias are defined as normal logic programs, and applied to two real datasets. The former uses the mobile dataset collected as part of the Reality Mining group [4] whereas the latter has been collected through a proof-of-concept mobile phone application we have developed, called *ULearn*. The application collects contextual information about user mobility as well as user behaviours in terms of interactions with the device (e.g. accepting a call, rejecting a call, etc.), it allows the users to specify a language bias and computes rule-based enforcement policies. These are presented to the user in the form of English text for validation purposes and the user can refine the learning outcomes by selecting rules to revise and enforcing constraints on the language. The evaluation of the learning outcomes in both applications shows that the learning accuracy changes according to the heuristics and considerably improves with the use of a standard cover loop.

The paper is structured as follows. Section 2 summarises basic background notions used throughout the paper. Section 3 presents our learning framework and introduces main parts of its background knowledge and language bias modelled in the specific domain of phone calls. Section 4 illustrates the application of the framework to two real mobile domain datasets presenting some accuracy results of the learning outcomes. Finally, Section 5 concludes with final remarks and directions for future work.

2 Background

We assume the reader is familiar with basic notions and terminologies of Logic Programming. ILP is regarded as a machine learning technique that is used to enrich a knowledge base with rules that discriminate between positive and negative examples. Specifically, ILP is concerned with the computation of hypotheses H that together with a background knowledge B explain a given set of examples E, namely $B \cup H \models E$ under given semantics. In this paper we consider the case when B and H are normal logic programs, E is a set of ground literals and $\models$ is the entailment relation under the stable model semantics.

The space of possible solutions is inherently large in particular in real domain applications with large datasets, so different levels of constraints can be imposed to restrict the search for hypotheses. A structure on the hypothesis can be employed to impose an instance-specific language bias S. Mode declarations are a common tool to specify a language bias [3]. These define which predicates to use in the head and in each of the body conditions of the rules that form a hypothesis as well as how their arguments are unified or grounded. In the TAL system [2], the mode declarations are mapped into a top theory $\top$ that constrains the search by imposing a generality upper bound on the inductive solution. The system uses an abductive proof procedure instantiated on this top theory together with the background theory. The abductive derivation identifies the heads of the rules (of a hypothesis solution) and the conditions needed to cover positive examples and to exclude negative examples, ensuring consistency. The abductive solution is guaranteed to have a corresponding inductive hypothesis H that is a solution with respect to the examples. For further details on the ILP system, TAL, the reader is referred to [2].

3 Towards an adaptive system using ILP

In this section we briefly describe our learning framework for learning and revising mobile user behaviour rules, with a brief overview of the concepts modelled in the background knowledge and language bias. As shown in Figure 1, our learning framework includes a modelling step where background knowledge and examples are modelled as normal logic programs and a language bias is defined. The TAL learning system is then applied. Rule refinement can then be performed on the learned outcomes based on a subsequent collection of data. The framework also includes cross validation mechanisms for assessing the accuracy of the rules learned.

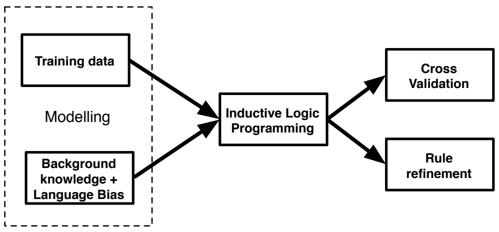
Training data
Modelling
Background knowledge + Language Bias
Inductive Logic Programming
Cross Validation
Rule refinement

Fig. 1: The learning framework

In the application domain of mobile phones, as well as in many context-sensitive applications, the concept of time plays an important role in defining user behaviours, as all user behaviours occur in terms of time. We answer phone calls at a particular point in time and we are at location Y at a time-point X. These are often not instantaneous and have a certain time duration. Events are therefore modelled in our background knowledge using a notion of *timestamp span* defined in terms of a start-time and an end-time. A basic notion of time as $[Day, Month, Year]$ has been defined together with ordering relations over time (e.g. before and after). These have been used to define different notions of *duration span* which allow inference of specific knowledge from our collected data. Examples related to the domain of mobile phones include:

- **Activity span:** defines the period of a user's activity on the phone.
- **Application span:** denotes the period as well as the type of application a user is using. Application types are defined in the background knowledge.
- **Device span:** represents the period of time for which a device is present in the user's vicinity. Devices are also typed and defined in the background knowledge.

- **Cell span:** indicates the period in which the user is at a certain location. All locations traversed by the user are typed and included in the background knowledge.
- **On span:** Shows the period that the phone is switched on.

More abstract notions are defined in the background knowledge in terms of basic notions of time and duration span so to allow the learning of user-behaviour rules that refer to more "high-level" concepts. These include notions of concepts of time like weekend, morning, afternoon and evening, as well as location of user, device proximity, user activity, application usage events, charge event, etc., each at a time point $[D, T]$. These notions are defined in terms of their respective span notions described above and checks between the time and the duration of the span. For example, device proximity at time $[D, T]$. This is defined in the background knowledge in terms of the existence of a device span such that $[D, T]$ is after $[D1, T1]$ but before $[D2, T2]$. Further, location plays a crucial role in defining mobile user behaviours. Sufficient contextual information about user's transition from one location to another can be collected through the mobile device and used to define a predicate that expresses the user being at a location X at time $[D, T]$.

Different language bias can be defined to specify the structure of the user behaviour rules that we might be interested to learn. As proof of concept we have considered the task of learning when a user answers or rejects a phone call. The head declaration for such a task can be as rich as needed in order to compute rules that are dependent on few or many contextual aspects. An example of such head declaration is $modeh(accept(+date, +time, +contact))$ with a corresponding body declaration of the form $modeb(weekend(+date))$, $modeb(evening(+time))$, $modeb(= (+contact, \#contact), \text{[no ground constants]})$, and $modeb(\backslash + (= (+contact, \#contact)), \text{[no ground constants]})$, where the argument [no ground constants] defines the number of ground constants allowed in the search. For lack of space we omit the full definition of our language bias.

4 Real mobile domain applications

We have applied our framework to two different mobile domain datasets. Each learning outcome has been cross validated using 5 folds and we perform ROC[1] analysis on each fold in order to compute the total error estimate. We show below, the solutions in English that have been produced automatically from the output of TAL by means of a translation mechanism that we have implemented.

The first dataset is the Reality Mining [4] dataset. This represents the largest mobile phone experiment attempted in the academic word. It consists of a large amount of data on human behaviour and group interactions collected using one hundred Nokia 6600 smart phones with pre-installed software developed at the University of Helsinki. The information collected includes call and message logs, Bluetooth devices in proximity of approximately five meters, cell tower IDs, application usage and phone status (i.e. active or idle). The dataset has been anonymised and made available to the general public[2]. We have selected a wide

[1] ROC: Receiver Operating Characteristic
[2] The Reality Mining Dataset:http://reality.media.mit.edu/dataset.php

range of users, but ultimately focussed on studying the most problematic cases in terms of user's actions. Due to space limitation we only give an example of the most accurate user behaviour rules that we have been able to compute from this dataset. User #96 has a total number of 142 positive examples and 35 negative examples. Average error estimate turned out to be 19.2%. Listing 1.1 illustrates the best solutions while Table 1 shows some of the performance metrics of the ROC analysis.

```
solution(-106,[(accept(_,_,C):-\+C=200)])
solution(-104,[(accept(A,B,_):-not_nearDevice(A,B,413)),(accept(_,_,H):-\+H=200)])
solution(-104,[(accept(_,_,C):-C= -1),(accept(_,_,G):-\+G=200)])

Accept calls: not from contact 200
Accept calls: when you're not near device 413, OR not from contact 200
Accept calls: when the contact is not in your address book, OR not from contact 200
```

Listing 1.1: Results for User #96

Fold	Accuracy	Error	Precision(PPV)
Fold 1	0.8571	0.1429	0.8529
Fold 2	0.9429	0.0571	0.9429
Fold 3	0.7143	0.2857	0.7143
Fold 4	0.6857	0.3143	0.6857
Fold 5	0.8378	0.1622	0.8378

Table 1: Performance Measure Results for User #96

Accuracy is the proportion of true results (both true positives and true negatives), and the accuracy of the above rule lies between 70% and 95%, a measure which is very promising. Precision is defined as the proportion of the true positives against all positive results (both true positives and false positives). In many cases, precision is equal to accuracy meaning that our results are both accurate and close to each other, showing that in each fold, the user's behaviour does not change much. Overall, the majority of the rules learned have one or two literals in the body and the best solutions include always the negation of a contact as the condition for accepting a call. This illustrates that the users' decision of accepting/rejecting calls is based on who the caller is, a result which is largely intuitive.

For the second dataset, we have developed a comprehensive client-server application, called *ULearn* where data acquisition and user interaction takes place on an Android phone whilst the processing of data and execution of the learning algorithm happens on the server side. The rules, as a result of the training examples, background knowledge and language bias, are displayed to the user for validation. One of the main benefits of ULearn is the user's involvement in the learning process. The user can select any number of integrity constraints to impose restriction on the search space, or select already learned rules for theory revision. We make use of the algorithm presented in [1] so that the existing rules are able to reflect and account for newly-seen instances of examples and background knowledge. We have collected data from two users over a period of approximately 2 months. Here we present the best solutions for both users and also show how the results immediately improve when the cover loop approach is used. The score for each solution represents the accuracy of each rule.

User #2

```
/% Without cover loop %/
solution(-0.7065, [
 (accept(P,Q,R,_,_,_,_,_,Y,_):-\+user_is_active(P,Q),\+R=7517429133,\+Y=1280),
 (accept(A,B,C,_,_,_,_,_,_,_):-\+user_is_active(A,B),\+C=7517429133,timex_after_h(B
      ,10))])

Accept calls:  When you're not active, not from contact 7517429133, not when
 your phone's light level is 1280,  OR when you're not active, not from contact
      7517429133, after 10:00
 o'clock

/% With cover loop %/
solution(-0.7717, [
 (accept(A,B,_,_,_,_,_,_,J,_):-not_at(A,B,1071.8253461),J=225),
 (accept(_,_,Q,_,_,_,_,_,_,_):-Q=1200490800),
 (accept(_,_,C1,_,_,_,_,_,_,_):-C1=447515692890),
 (accept(_,_,_,_,_,_,_,U1,_,_):-U1=0),
 (accept(Y1,Z1,A2,_,_,_,_,_,H2,_):-\+user_is_active(Y1,Z1),\+A2=7517429133,\+H2
      =1280)])

Accept calls: anywhere unless you're at 1071.8253461, when your light level is
 225, OR from contact 1200490800,  OR from contact 447515692890,  OR when
 your screen is off,  OR when you're not active, not from contact 7517429133, not
 when your light level is 1280
```

Listing 1.2: Best solution obtained without Cover Loop

5 Conclusion

The work presented in this paper demonstrates the applicability of the TAL system to real mobile domains for supporting the learning of mobile user behaviours. With appropriate definition of a relevant domain of discourse, language bias and background knowledge our evaluation results indicate that the system performs well when dealing both with large domains and large amount of data. In particular, it has proven to be a powerful non-monotonic ILP system that tolerates noise and scales well with large domains and data because of its use of finite domain constraints that are not available in other systems. Further work includes enriching the background knowledge and language bias further. For example, using light level information we can enrich the inference of context information for instance, whether the user's mobile phone is inside their pocket or handbag. We can further use information about location to predict the user's next location. Last, but not least, we can explore the use of bagging, boosting together with bootstrapping datasets in order to compute potentially richer rules with even higher accuracy.

References

1. D. Corapi, O. Ray, A. Russo, A. Bandara, and E.C. Lupu. Learning Rules from User Behaviour. In *2nd International Workshop on the Induction of Process Models*, September 2008.
2. D. Corapi, A. Russo, and E.C. Lupu. Inductive Logic Programming as Abductive Search. In *Technical Communications of the 26th International Conference on Logic Programming*, 2010.
3. L. DeRaedt and S. Muggleton. Inductive Logic Programming: Theory and Methods. In *Journal of Logic Programming*, pages 19/20:629–680, 1994.
4. N. Eagle, A. Pentland, and D. Lazer. Inferring Social Network Structure using Mobile Phone Data. *PNAS*, 2007.