Knowl Inf Syst (2011) 28:283–310
DOI 10.1007/s10115-010-0359-z

REGULAR PAPER

CAS-MINE: providing personalized services in context-aware applications by means of generalized rules

Elena Baralis · Luca Cagliero · Tania Cerquitelli · Paolo Garza · Marco Marchetti

Received: 29 January 2010 / Revised: 22 July 2010 / Accepted: 25 October 2010 /
Published online: 12 November 2010

Abstract Context-aware systems acquire and exploit information on the user context to tailor services to a particular user, place, time, and/or event. Hence, they allow service providers to adapt their services to actual user needs, by offering personalized services depending on the current user context. Service providers are usually interested in profiling users both to increase client satisfaction and to broaden the set of offered services. Novel and efficient techniques are needed to tailor service supply to the user (or the user category) and to the situation in which he/she is involved. This paper presents the CAS-MINE framework to efficiently discover relevant relationships between user context data and currently asked services for both user and service profiling. CAS-MINE efficiently extracts generalized association rules, which provide a high-level abstraction of both user habits and service characteristics depending on the context. A lazy (analyst-provided) taxonomy evaluation performed on different attributes (e.g., a geographic hierarchy on spatial coordinates, a classification of provided services) drives the rule generalization process. Extracted rules are classified into groups according to their semantic meaning and ranked by means of quality indices, thus allowing a domain expert to focus on the most relevant patterns. Experiments performed on three context-aware datasets, obtained by logging user requests and context information for three real applications, show the effectiveness and the efficiency of the CAS-MINE framework in mining different valuable types of correlations between user habits, context information, and provided services.

Keywords Generalized association rules · Context-aware applications · User and service profiling · Itemset mining · Rule classification

E. Baralis · L. Cagliero · T. Cerquitelli
Dipartimento di Automatica e Informatica, Politecnico di Torino, Torino, Italy

P. Garza (✉)
Dipartimento di Elettronica e Informazione, Politecnico di Milano, Milano, Italy
e-mail: garza@elet.polimi.it

M. Marchetti
Telecom Italia Lab, Torino, Italy

1 Introduction

Context-aware applications allow service providers to adapt their services to the actual user needs, thus offering them personalized services depending on their current application context. Provided services could be personalized by exploiting either the current context of the user [7,18], or historic context and behavior of the user [8]. An effective user and service context profiling focuses on supporting user activities and service provisioning by means of a general-purpose system able to tailor service supply with high flexibility.

Research activities on context-awareness computing have been devoted both to providing different definitions of context-awareness and to building different context-aware applications (e.g., mobile phone applications [11], medical applications [38], and computer vision applications [40]). Context consists of any circumstantial factors or application context users are involved in. Thus, context-awareness means that the system is able to use context information. A system is context-aware if it can extract, interpret and use context information and adapt its functionalities to the current usage context.

This paper thoroughly describes the CAS-MINE framework to efficiently discover relevant relationships between user context data and currently asked services for both user and service profiling. To this aim, CAS-MINE performs: (i) Gathering, cleaning, and integration of service requests sent by users through mobile devices, (ii) discovery of recurrent and interesting rules in context data to support user and service profiling, and (iii) rule classification into different semantic groups.

When a user requests a given mobile service, the service provider may receive the following information: context data (e.g., GPS coordinates, temporal information), requested service, and user submitting the request. The collected data is cleaned and integrated into a common data structure to produce a service request log integrating all the context information. On this log different data mining techniques could be performed to extract recurrent and hidden patterns. For example, association rule extraction [1] is a widely used exploratory technique to effectively discover correlation among data. Thus, it finds application in a wide range of different domains (e.g., medical images [3], biological data [9]).

Traditional association rule extraction, driven by support and confidence constraints, may entail either (i) generating an unmanageable number of rules in case of low support thresholds, or (ii) discarding information associated to rare (infrequent) rules, even if its hidden knowledge might be relevant to the analyst. Different approaches are needed to effectively manage different data granularities during the mining activity. To provide a high-level abstraction of both user habits and service characteristics depending on the context, CAS-MINE efficiently extracts recurrent patterns in the form of generalized association rules [29]. A lazy (analyst-provided) taxonomy evaluation performed on different attributes (e.g., a geographic hierarchy on spatial coordinates, a classification of provided services) drives the rule generalization process. This approach prevents discarding relevant but infrequent knowledge by opportunistically generalizing only low support itemsets.

Consider for example a context-aware application that provides a weather forecast service to users. It produces a log file holding information on users and their requests for different (mobile) services. Recurrent patterns in its log file may be represented by means of association rules. For example, to describe a user request for the weather service, the following rule might be extracted.

$$\texttt{user} : \text{John}, \texttt{time} : 5.05\,\text{p.m.} \Rightarrow \texttt{service} : \text{Weather}\ (sup = 0.005\%)$$

where *sup* is the rule support and describes the observed frequency of the rule in the dataset. The mining process is typically driven by enforcing a minimum support threshold.

Mining the above rule would require a very low minimum support to be considered during the rule extraction process. A high-level knowledge might be provided by the following (generalized) association rule

$$\texttt{user} : \text{commuter employee}, \texttt{time} : \text{between 5 and 6 p.m.} \Rightarrow \texttt{service} : \text{Weather}\ (sup = 1.2\%)$$

which shows the weather service requests performed by all commuter employees at the end of the common day shift, during the time slice between 5 p.m. and 6 p.m. It provides more general knowledge about the time at which weather service requests are submitted by a category of employees. The generalization step may be driven by an analyst-provided taxonomy defined over the user and time attributes. The second rule is characterized by a higher support than the first and is lazily extracted only if the former rule does not meet the support threshold during the extraction process. The knowledge in this rule may be exploited for user profiling. For example, this information could be exploited by the service provider to include in the personalized page of each commuter employee, from 5 p.m. to 6 p.m., the weather service. This personalized approach allows commuter employees to access more quickly the most frequently used services depending on the timeslot.

Since the analysis of the (usually) large set of extracted rules is not a trivial task, a classification of the rules into groups according to their semantics in the context-aware application has been proposed. The classification step has been based on general rule structures denoted as rule templates. Different rule templates have been devised and discussed. Finally, to allow a domain expert to focus on the most relevant patterns, the most interesting rules have been ranked according to well-known quality indices (e.g., lift). CAS-MINE final output is a set of categorized rules that characterize user habits and service characteristics depending on the current context. It highlights correlations and recurrences of interesting patterns among data. Experiments performed on three real context-aware datasets provided by Telecom Italia show the efficiency and effectiveness of the CAS-MINE framework in characterizing users and services by highlighting interesting rules.

The paper is organized as follows. Section 2 presents an overview of the CAS-MINE framework and the main features of its building blocks. Section 3 describes how context-aware data are collected and integrated, while Sect. 4 formally defines the generalized association rule mining problem and how it is addressed in CAS-MINE . Section 5 thoroughly describes a classification of the extracted rules into interesting groups according to their semantics in the context-aware applications. In Sect. 6, a rich set of experiments to validate the proposed framework is presented and discussed. Section 7 compares our approach with previous works. Finally, Sect. 8 draws conclusions and discusses future works.

2 The CAS-MINE framework

The CAS-MINE framework is a context-aware environment to effectively perform both user and service profiling. It allows shaping service provisioning by considering the current context of the user. By discovering recurrent patterns on user habits, service providers can partition users into pre-defined categories over which service provisioning may be modeled and personalized. Furthermore, service profiling may allow providers to effectively shape service supply, promotions, and system size depending on the actual application usage.

CAS-MINE exploits the GENIO algorithm [4], an efficient algorithm to discover generalized association rules. The mining process is driven by a (analyst-provided) taxonomy that

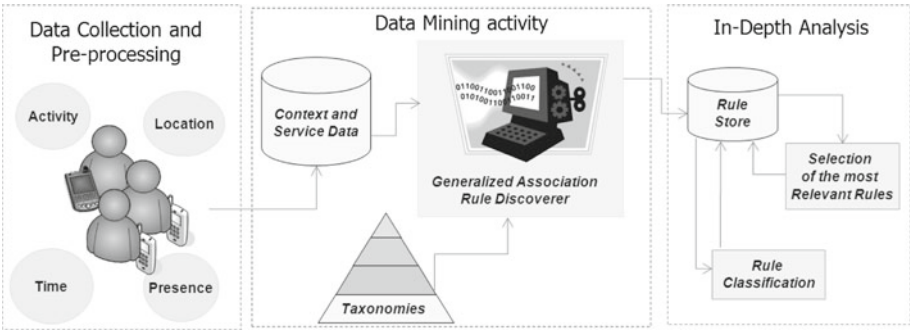


Fig. 1 The CAS-MINE framework architecture

allows the conceptual aggregation of items. The main architectural blocks of the CAS-MINE framework (shown in Fig. 1) are described in the following.

Data collection and pre-processing. Context knowledge is provided by different and heterogeneous sources (e.g., mobile devices) collecting information about the context of the user submitting the request (e.g., GPS coordinates, temporal information), and the requested services (e.g., service description). Next, data are cleaned by removing irrelevant and redundant information and integrated into a common data structure.

Mining activity. The aim of the mining activity block is to discover interesting correlations and recurrent patterns in context data. In the CAS-MINE framework, interesting patterns are extracted in the form of *generalized association rules*, i.e., rules that represent general correlations among context data. The generalization step is performed by means of an analyst-provided taxonomy (i.e., is-a hierarchy) defined on structured data. Analysts should provide meaningful sets of aggregation hierarchies based on their knowledge on context information concerning user requests collected in the source dataset. Context data can be aggregated at different granularity levels to discover more informative and compact knowledge in a flexible way.
The extraction of generalized association rules is performed by means of a (traditional) two-step process: (i) Frequent generalized itemset extraction and (ii) rule generation from the extracted frequent itemsets. Since the GENIO algorithm [4] is known to be more efficient than previous approaches [13] in automatically extracting interesting generalized itemsets from structured data, CAS-MINE exploits GENIO to perform the first step. The extraction of generalized rules (i.e., the second step) is performed by our implementation of the rule mining step of the Apriori algorithm [2].

In-depth analysis. The aim of the in-depth analysis block is twofold: (i) Selection of the most relevant rules and (ii) rule classification. Currently, extracted patterns are ranked by exploiting the *lift correlation measure* to effectively discriminate between reliable and unreliable patterns. The lift measure highlights rule correlation, thus overcoming the support and confidence well-known drawbacks without requiring complex computations. However, other data quality indices [34] can be easily integrated into the CAS-MINE framework as well.
The rule classification step categorizes extracted rules according to their semantic interpretation in context-aware applications. Thus, extracted rules are partitioned in two main classes: user rules and service rules. User rules characterize user habits at different aggregation levels

and allow service providers to offer personalized services tailored to the current user context. Service rules, instead, describe service characteristics and allow service providers to adapt service provisioning to the current context, independently of the requesting user. For each class, some relevant rule templates have been identified and discussed.

A more detailed description of each block and its functionalities is presented in the following sections.

3 Data collection and pre-processing

Since service requests typically depend on the requesting user environment, a collection of information describing the user context at request submission may enhance the quality of the provided services. The data collection and pre-processing block of the CAS-MINE framework handles data collection through services on mobile devices that collect user and service context information (e.g., temporal information, GPS coordinates, service description). To ensure user privacy, context data retrieval and processing in a context-aware environment requires both the user informed consent on personal data treatment and compliance with laws in force.

Due to the distributed nature of mobile systems, separate logs have been recorded in different systems, describing different parts of the user activity. The data collection phase takes as input the raw context data provided by different, maybe heterogeneous, sources and join them into a unique data repository.

The pre-processing phase aims at making raw input data fully compatible with the repository format. During the joining process, data are tailored to a common data structure by means of a data cleaning process. Data cleaning also discards useless or redundant information and correctly manages missing values. After preprocessing, the collected context information can be modeled as a structured dataset, where records represent service requests. Each record identifies a different service request. A record is a set of items, where an item is a couple *(attribute_name, value)*. While *attribute_name* is the description of the represented information (e.g., the label *timestamp*), *value* is the collected information (e.g., timestamp 11:10 a.m.). A more formal definition follows.

Definition 3.1 (*Item*) Let t_i be a label, called attribute, which describes a data feature. Let Ω_i be the discrete domain of attribute t_i. An item $t_i = value_i$ assigns the value $value_i \in \Omega_i$ to attribute t_i.

In the case of continuous attributes, the value range is discretized into intervals, and the intervals are mapped to consecutive positive integers.

Definition 3.2 (*Structured context dataset*) Let $\mathcal{T} = \{t_1, t_2, \ldots, t_n\}$ be a set of attributes and $\Omega = \{\Omega_1, \Omega_2, \ldots, \Omega_n\}$ the corresponding domains. A structured context dataset D is a collection of records, where each record r is a set of items and contains at most one item for each attribute in $\mathcal{T}$.

A structured context dataset holds information on both service requests performed by different users and the corresponding application context in which requests are submitted. Each record is a set of items describing a specific user service request. Attributes describe the represented information (e.g., *user identifier*, *service*, *time*) and take values (e.g., *ID54*, *weather*, 4:06 p.m.) in the corresponding attribute domains.

Definition 3.3 (*Itemset*) Let $\mathcal{T} = \{t_1, t_2, \ldots, t_n\}$ be a set of attributes and $\mathcal{I} = \{t_1 = value_1, t_2 = value_2, \ldots, t_n = value_n\}$ contain the enumeration of all the items in the

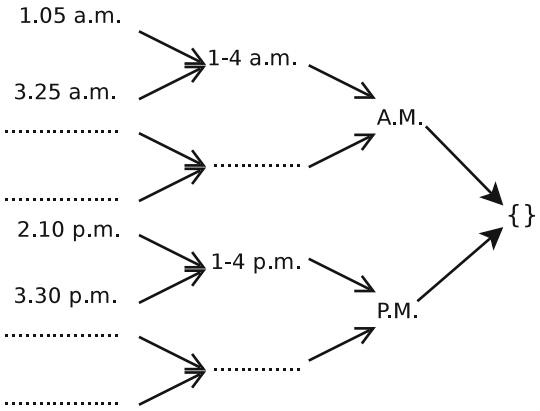


Fig. 2 A rewriting rule tree RR_{time} for the *time* attribute

corresponding structured context dataset. An itemset $X \subseteq \mathcal{I}$ is a set of items. *attr(X)* $\subseteq \mathcal{T}$ denotes the set of all attributes characterizing the items in X. Each attribute $t_i \in \mathcal{T}$ may occur at most once in *attr(X)*.

An itemset of length k is also called k-itemset.

4 Generalized association rule extraction

Generalized association rules provide a high-level domain knowledge abstraction that allows a compact representation of general correlations among context data. Section 4.1 formally defines generalized association rules and provides many examples in the context-aware service domain, while Sect. 4.2 describes how the knowledge discovery process takes place.

4.1 Generalized association rules

The discovery of generalized association rules is driven by analyst-provided aggregation hierarchies. Consider, for example, the *time* attribute, which defines the submission time of a service request. A simple aggregation hierarchy that may be devised by a domain analyst is shown in Fig. 2. The aggregation hierarchy aggregates the submission time of the service request by 4-hour timeslot, and a.m./p.m. time periods. The root (represented as {}) aggregates all values allowed for the *time* attribute. An aggregation hierarchy can be formally defined by means of a rewriting rule tree.

Definition 4.1 (*Rewriting Rule Tree*) Let t_i be an attribute and Ω_i its domain. A rewriting rule tree RR_i is a tree representing a predefined hierarchy of aggregations over values in Ω_i. RR_i leaves are all the values in Ω_i. Each non-leaf node in RR_i is an aggregation of all its children. The root node aggregates all values for attribute t_i.

In the context-aware service profiling domain, many different rewriting rule trees may be defined for the *time, date, service, phone number,* and *location* attributes. Analysts should provide meaningful rewriting rule trees (i.e., aggregation hierarchies) based on their knowledge on context data collected into the source dataset. A taxonomy is a collection of rewriting rule trees.

Definition 4.2 (*Taxonomy*) Let $\mathcal{T} = \{t_1, t_2, \ldots, t_n\}$ be a set of attributes and $\rho = \{RR_1, \ldots, RR_m\}$ a set of rewriting rule trees defined on $\mathcal{T}$. A taxonomy $\Gamma \subseteq \rho$, is a forest of rewriting rule trees. Γ contains at most one rewriting rule tree RR_i in ρ for each attribute t_i in $\mathcal{T}$.

A taxonomy needs to be unambiguous to guarantee termination of the rewriting process. This property is (trivially) proved by Theorem 1.

Theorem 4.1 *Let Γ be a taxonomy defined by Definition 4.2. Γ is not ambiguous.*

Proof Let RR_i in Γ be a rewriting rule tree for an arbitrary attribute t_i. By Definition 4.2, Γ contains at most a single rewriting rule tree for t_i. Thus, only a single rewriting path, given by the corresponding rewriting rule tree RR_i belongs to Γ for any arbitrary attribute t_i. It trivially follows that Γ is not ambiguous. □

A generalized item (or itemset) is built by exploiting the rewriting rule trees in a given taxonomy. For example, consider again the rewriting rule tree RR_{time} shown in Fig. 2. It defines several aggregation values (e.g., 1–4 a. m.) at different abstraction levels. Each of these values characterizes a generalized item (e.g., $t_{time} = 1\,\text{a.m.} - 4\,\text{a.m.}$). The formal definitions of generalized item and generalized item support follow.

Definition 4.3 (*Generalized item*) Let t_i be an arbitrary attribute, Ω_i its domain, and RR_i a rewriting rule tree defined over values in Ω_i. A generalized item $t_i = expression_i$ assigns the value $expression_i$ to attribute t_i. $expression_i$ is a non-leaf node in RR_i, defining an aggregated value over values in Ω_i. $leaves(expression_i) \subseteq \Omega_i$ defines the set of leaf nodes descendants of $expression_i$ in RR_i.

The support of a generalized item in a given dataset D may be counted by summing the frequency of all leaf values in D defined by the rule rewriting tree including it.

Definition 4.4 (*Support of a generalized item*) Let D be a structured dataset, $t_i = expression_i$ a generalized item, and RR_i the corresponding rewriting rule tree defined over the domain of t_i. The support of $t_i = expression_i$ is the sum of the (observed) frequency in D of $leaves(expression_i)$.

A generalized itemset may include both items and generalized items, as stated by the following definition.

Definition 4.5 (*Generalized itemset*) Let $\mathcal{T} = \{t_1, t_2, \ldots, t_n\}$ be a set of attributes and $\mathcal{I} = \{t_1 = value_1, t_2 = value_2, \ldots, t_n = value_n\}$ contain the enumeration of all the items in the corresponding structured context dataset. Let Γ be a taxonomy on attributes in $\mathcal{T}$, and $\mathcal{E} = \{t_1 = expression_1, t_2 = expression_2, \ldots, t_m = expression_m\}$ be the set of generalized items derived by all rewriting rule trees in Γ. A generalized itemset Y is a subset of $\mathcal{I} \bigcup \mathcal{E}$. Each attribute $t_i \in \mathcal{T}$ may occur at most once in Y.

For example, given the following 2-itemset

$$(\texttt{service} : \text{FlightStat}, \quad \texttt{time} : 3.05\,\text{p.m.})$$

the time attribute may be generalized by means of the rewriting rule tree in Fig. 2. This generalization process may yield, e.g., the generalized itemset

$$(\texttt{service} : \text{FlightStat}, \texttt{time} : 1\,\text{p.m. to} \quad 4\,\text{p.m.})$$

To define the generalized itemset support, we firstly introduce the concept of generalized itemset matching.

Definition 4.6 (*Generalized itemset matching*) Let D be a structured dataset and Γ a taxonomy on D. A generalized itemset X matches an arbitrary record $r \in D$ if and only if for all (possibly generalized) items $x \in X$

1. $x \in \mathcal{I}$ (i.e., x is an item) and $x \in r$, or
2. $x \in \mathcal{E}$ (i.e., x is a generalized item) and $\exists i \in leaves(x)$ such that $i \in r$

Definition 4.7 (*Generalized itemset support*) Let D be a structured dataset and Γ a taxonomy on D. The support of a generalized itemset X is given by the number of records $r \in D$ matching X divided by the cardinality of D.

Definition 4.8 (*Disjoint generalized itemsets*) Let A and B be two arbitrary generalized itemsets. A and B are disjoint iff $attr(A) \cap attr(B) = \emptyset$.

Definition 4.9 (*Generalized Association Rule*) Let A and B be two disjoint generalized itemsets. A generalized association rule is represented in the form $A \Rightarrow B$, where A and B are the body and the head of the rule, respectively.

A and B are, respectively denoted as antecedent and consequent of the generalized rule $A \Rightarrow B$. Generalized association rule discovery is driven by rule support, whose formal definition follows.

Definition 4.10 (*Generalized Association Rule Support*) Let $A \Rightarrow B$ be a generalized association rule. Its support s is the support of the generalized itemset $A \cup B$.

In general, the support represents the prior probability of A and B (i.e., its observed frequency) in the source dataset.

Definition 4.11 (*Generalized Association Rule Confidence*) Let $A \Rightarrow B$ be a generalized association rule. Its confidence c is given by $\frac{s(A \cup B)}{s(A)}$.

The confidence of a rule $A \Rightarrow B$ is the conditional probability of the generalized itemset B given the generalized itemset A.

For example, the following generalized association rule might be exploited to tailor the airline flight statistics service to the actual user needs.

$$(\texttt{service} : \text{FlightStat} \rightarrow \texttt{time} : \text{from 1 p.m. to 4 p.m.}) \; (s = 10\%, c = 88\%)$$

The concept of generalized itemset descendant is exploited by the rule extraction algorithm described in Sect. 4.2.

Definition 4.12 (*Generalized Itemset Descendant*) A (generalized) itemset X is a descendant of a generalized itemset Y if (i) X and Y have the same length and (ii) for each item $y \in Y$ there exists at least an item $x \in X$ that is a descendant of y.

4.2 Generalized association rule discoverer

Generalized association rules are discovered by means of a two-step process: (i) Frequent generalized itemset extraction and (ii) rule generation from the extracted frequent itemsets. The first step exploits the GENIO algorithm [4], and the second step is performed by our implementation of the rule mining step of the Apriori algorithm [2]. The mining process is driven by the support threshold, which drives the itemset extraction process.

Since itemset mining is known to be the most computationally expensive task [1], we focus on the first step. Given a context dataset, a taxonomy, and a minimum support threshold, the GENIO algorithm discovers (a) frequent itemsets and (b) frequent generalized itemsets whose descendants are infrequent. More specifically, a generalized itemset is opportunistically mined if and only if at least one of its descendants is infrequent. The generalization process is driven by an analyst-provided taxonomy (i.e., a set of rewriting rule trees). A pseudo-code of the GENIO generalized itemset discoverer is reported in Algorithm 1. A detailed description of the algorithm is provided in [4].

Algorithm 1 Generalized Itemset Discoverer

Require: minimum support *min_sup*, taxonomy Γ, dataset D
Ensure: L, set of generalized frequent itemsets
1: $k = 1$, $L = \emptyset$
2: C_1 = set of items in D
3: **repeat**
4: scan D and count support for each $c \in C_k$
5: $Gen = \emptyset$ // generalized itemset container
6: **for all** c in C_k **do**
7: **if** support of $c <$ *min_sup* **then**
8: *new_gen_itemsets* = taxonomy_evaluation(Γ, c)
9: update Gen with *new_gen_itemsets*
10: **end if**
11: **end for**
12: **if** $Gen \neq \emptyset$ **then**
13: scan D and count support for each itemset in Gen
14: **end if**
15: $L_k = \{$ itemsets in $\{C_k \cup Gen\}$ whose support $\geq$ *min_sup* $\}$
16: $k = k + 1$
17: C_k = candidate_generation(L_{k-1})
18: **until** $C_k \neq \emptyset$
19: **return** L

GENIO, similarly to Apriori [2], is a level-wise algorithm, which, at each iteration, generates all frequent itemsets of a given length. At an arbitrary iteration k, the Apriori algorithm performs two steps: (i) Candidate generation, in which all k-itemsets are generated from *(k-1)*-itemsets, (ii) candidate pruning, to discard candidate itemsets which cannot be frequent. Finally, actual candidate support is counted by scanning the dataset.

GENIO follows the same level-wise approch. However, GENIO (i) manages infrequent (i.e., rare) itemsets by triggering the taxonomy evaluation (lines 6–11) and (ii) exploits the characteristics of the structured dataset to effectively prune candidates (line 17). More specifically, once the support of each candidate itemset has been computed (line 4), generalized versions of infrequent ones are generated by evaluating the taxonomy (line 8) and inserted in the Gen set (line 9). In particular, given an infrequent candidate Y, taxonomy climbing yields the generalized version of each item in Y by applying on each item $t_j = value_j$ in Y the corresponding rewriting rule tree RR_j. Generalized itemsets of Y are all the itemsets obtained by replacing one or more items in Y with their corresponding generalized items. Hence, taxonomy climbing on Y potentially generates a set of generalized itemsets. The updating procedure of the generalized pattern set Gen (line 9) prevents the insertion of generalized itemsets previously generated by different infrequent descendants. If Gen is not empty, the support of each generalized itemset in Gen is computed by performing a further scan of the dataset (line 13). The opportunistic approach to generalized itemset mining leads to a lazy taxonomy climbing, triggered by infrequent itemsets only. Thus, GENIO only extracts generalized itemsets having at least one infrequent descendant.

For example, consider the following itemset on the airline flight service

(service : FlightStat, time : 3.05 p.m.)

mined from a context dataset. If a rewriting rule tree is only available on the *time* attribute (see Fig. 2), the following generalized itemset is generated.

(service : FlightStat, time : 1 p.m. to 4 p.m.)

If the generated itemset is found to be frequent, the generalization process stops. Otherwise, the rewriting rule tree is exploited again to climb up one more level in the hierarchy. Differently from above, suppose now that a rule rewriting tree $RR_{service}$ is also available for the *service* attribute. Thus, the FlightStats *service* can be aggregated into a high-level generalization TRAVELS. The generalization process now generates the following three itemsets.

(service : FlightStat, time : 1 p.m. to 4 p.m.)
(service : TRAVELS, time : 3.05 p.m.)
(service : TRAVELS, time : 1 p.m. to 4 p.m.)

To further prune candidate itemsets, GENIO exploits the uniqueness of attribute labels and values in each record of a structured dataset (e.g., a service request record may contain one *service* attribute taking at most a single value). For example, suppose that after counting item support, m frequent 1-itemsets tagged *service* and n tagged *user* (all with different attribute values) are extracted. Apriori exhaustive candidate generation would produce $\binom{m+n}{2}$ possible combinations. Since each attribute is allowed only once in each record, only $m \cdot n$ 2-itemsets (obtained by combining one item tagged *service* and one item tagged *user*) are relevant combinations. GENIO only generates this candidate subset, thus significantly reducing the computational cost and memory requirements.

5 Supporting in-depth rule analysis

The in-depth analysis block of CAS-MINE addresses (i) the ranking of the most relevant rules by exploiting the lift quality index [34] (see Sect. 5.1) and (ii) the classification of interesting rules useful for effectively supporting user and service profiling in context-aware applications (see Sect. 5.2).

5.1 Ranking relevant rules

Many quality measures [34] may support selection and ranking of the most interesting rules. The rules mined by CAS-MINE are sorted by means of the lift index [34], which measures the (symmetric) correlation between body and head of the extracted rules. The lift of a (generalized) association rule $A \Rightarrow B$ is defined as [34]

$$\text{lift (A, B)} = \frac{\text{c}(A \Rightarrow B)}{\text{s(B)}} = \frac{\text{s}(A \Rightarrow B)}{\text{s(A) s(B)}} \quad (1)$$

where $\text{s}(A \Rightarrow B)$ and $\text{c}(A \Rightarrow B)$ are, respectively the rule support and confidence, and $\text{s}(A)$ and $\text{s}(B)$ are the supports of the rule antecedent and consequent. If lift(A,B) = 1, the itemsets A and B are not correlated, i.e., they are statistically independent. Lift values below 1 show negative correlation, while values above 1 indicate a positive correlation between itemsets A and B.

Both positively and negatively correlated rules are selected by CAS-MINE to highlight interesting situations. For instance, positively correlated rules highlight the preferred user services, while negatively correlated rules identify the services used less than expected. Similarly, the interest of rules having a lift value close to 1 may be marginal. Hence, CAS-MINE ranks the mined rules according to their lift value to focus the analysis on the set of most (positively or negatively) correlated rules.

5.2 Rule classification

The rule classification block categorizes correlated rules in classes to effectively address context-aware profiling. Since service providers are mainly interested in profiling both users and services, the CAS-MINE framework identifies two main classes of generalized association rules: (i) User rules, described in Sect. 5.2.1 and (ii) service rules, described in Sect. 5.2.2.

To classify extracted rules, we propose rule templates that define the general structure of interesting subsets of generalized association rules. All rules that share the same template are characterized by the same attribute(s), but not necessarily the same values, in the body and in the head of the rule.

Definition 5.1 (*Generalized association rule template*) Let $\mathcal{T} = \{t_1, t_2, \ldots, t_n\}$ be a set of attributes. A generalized association rule template is represented in the form $X \Rightarrow Y$, where X and Y (the body and the head of the rule template, respectively) are two disjoint sets of attributes (i.e., $X \subseteq \mathcal{T} \wedge Y \subseteq \mathcal{T} \wedge X \cap Y = \emptyset$).

5.2.1 User rules

User rules characterize user habits at any aggregation level. These rules allow service providers to offer personalized services depending on the current context of the user. Hence, provided services can be adapted to actual user needs. In user rules, the attribute "user" always appears either in the body or in the head of the rule.

User rules are further partitioned in more specific categories. For each category, a set of interesting rule templates is defined. For the sake of simplicity, rule templates are currently defined on length-2 and length-3 rules. The most interesting user rule templates are summarized in Tables 1 and 2. They always include the user attribute in the body of the rule. For each template, an example and a short explanation are provided. User rules compliant with a given template describe user context knowledge at the appropriate [1] hierarchical level. Thus, they are suitable for context-aware user profiling.

Table 1 shows length-2 user rule classes. These classes are characterized by a single attribute both in the body and in the head of the rule. They have been further semantically partitioned into (a) common requested services (denoted as *RS*), defining the service type a user is mainly interested in, and (b) context of the requested service, identifying when or where users request for services (denoted with prefix *AU*). For the *AU* category, three different rule templates have been defined: The *AU-T* template identifies the time period at which the user requests services. The *AU-D* template models the period of the year in which the user requests services. The *AU-P* template defines the location (place) of the user requesting services.

User rules not falling into the above classes may belong to two categories: (a) The rule is a specialization of rules belonging to templates in Table 1 (i.e., it includes a superset

[1] The appropriate level depends on the selected support threshold.

Table 1 Length-2 user rule classes

Class	Question	Rule template	Example
AU-T	Given a user (or a user category), in what time period does he request services (or classes of services)?	$\{user\} \Rightarrow \{time\}$	$\{user = John\} \Rightarrow \{time =$ 6 − 7 p.m.$\}$ means that user John submits service requests between 6 and 7 p.m
AU-D	Given a user (or a user category), in what period of the year does he request services (or classes of services)?	$\{user\} \Rightarrow \{date\}$	$\{user = John\} \Rightarrow \{date = winter\}$ means that user John submits service requests during the winter
AU-P	Given a user (or a user category), where does he request services?	$\{user\} \Rightarrow \{place\}$	$\{user = John\} \Rightarrow \{place =$ $OFFICE\}$ means that user John requests services in his office
RS	Given a user or a user category, which services (or class of services) is he interested in?	$\{user\} \Rightarrow \{service\}$	$\{user = John\} \Rightarrow \{Service =$ $SMS\}$ means that user John requests the SMS service

of the attributes of rule templates in Table 1), or (b) it references different attributes, thus representing knowledge that need not to be separately classified. Table 2 defines templates for a subset of the specialized rules in category (a). Specialized templates are grouped into two subsets, analogously to templates in Table 1. The first six rule templates in Table 2 are specializations of the *AU* class template, while the last rules are specializations of *RS*. For instance, the specialized rule template $\{user\} \Rightarrow \{place, time\}$ adds the time attribute to the rule template $\{user\} \Rightarrow \{place\}$. Hence, it specializes the application usage class by also considering the correlation with the time period of the user requests.

All the categories in Tables 1 and 2 focus on different characteristics of the user–system interaction. User personalization has the goal of enhancing user-friendliness of the applications by capturing valuable recurrences in user habits. The knowledge provided by user rule analysis can be exploited to (i) select the first service (default service) to be suggested to connected user, (ii) automatically complete service parameters, (iii) plan future promotions, (iv) suggest the appropriate service type in a given context, and (v) automatically invoke a specific service when the user is in a given context. The proposed user rule classification is an effective tool to highlight hidden knowledge which may be relevant to this purpose. In Sect. 6.3.2, these templates are exploited for the analysis of real context datasets and the concrete usage of extracted rules is also discussed.

5.2.2 Service rules

Service rules describe service characteristics, at any hierarchical level, regardless of the requesting users. These rules allow service providers to shape service provisioning to the current context. In service rules, the attribute "service" always appears either in the body or in the head of the rule.

Similarly to user rules, service rules have been partitioned in more specific categories to support rule analysis. For each category, a set of interesting rule templates has been defined. Two significant subsets of rule templates for service rules of length 2 and 3 are summarized in Tables 3 and 4. They always include the service attribute in the body of the rule. Since interesting knowledge on service exploitation typically concerns its usage context (e.g., time or location), or the service parameters the user requires, four service templates of length 2 have been defined in Table 3. For a given service, the *ST* template defines the usage time, *SD*

Table 2 Length-3 user rule classes

Class	Question	Rule template	Example
AU-PT	Given a user (or a user category), where does he request services and in which time period?	$\{user\} \Rightarrow \{place, time\}$	$\{user = John\} \Rightarrow \{place = office, time = morning\}$ means that user John requests application services during the morning in his office
AU-PD	Given a user (or a user category), where does he request services and in which period of the year?	$\{user\} \Rightarrow \{place, date\}$	$\{user = John\} \Rightarrow \{place = office, date = winter\}$ means that user John requests application services during the winter in his office.
AU-PPa	Given a user (or a user category), where does he request services and which service parameters are specified?	$\{user\} \Rightarrow \{place, param\}$	$\{user = John\} \Rightarrow \{place = office, param = OUT\}$ means that user John requests outgoing application services in his office
AU-DT	Given a user (or a user category), in what time and year periods does he request services?	$\{user\} \Rightarrow \{date, time\}$	$\{user = John\} \Rightarrow \{date = June, time = morning\}$ means that user John requests application services in the morning during June
AU-PaT	Given a user (or a user category), in what daily time periods does he request and which service parameters are specified?	$\{user\} \Rightarrow \{param, time\}$	$\{user = John\} \Rightarrow \{param = OUT, time = morning\}$ means that user John requests outgoing application services in the morning
AU-PaD	Given a user (or a user category), in what period of the year does he request services and which service parameters are specified?	$\{user\} \Rightarrow \{param, date\}$	$\{user = John\} \Rightarrow \{param = OUT, date = winter\}$ means that user John requests outgoing application services in winter
RS-T	Given a user (or a user category), what service (class) does he request and in what time period?	$\{user\} \Rightarrow \{service, time\}$	$\{user = John\} \Rightarrow \{service = CALL, time = 2 - 6\,\text{p.m.}\}$ means that user John requests the CALL service during the afternoon
RS-D	Given a user (or a user category), what service (class) does he request and in what period of the year?	$\{user\} \Rightarrow \{service, date\}$	$\{user = John\} \Rightarrow \{service = CALL, date = December\}$ means that user John requests the CALL service in December
RS-P	Given a user (or a user category), what service (class) does he request and where?	$\{user\} \Rightarrow \{service, place\}$	$\{user = John\} \Rightarrow \{service = CALL, place = office\}$ means that user John requests the CALL service in his office
RS-Pa	Given a user (or a user category), what service (class) does he request and with which service parameters?	$\{user\} \Rightarrow \{service, param\}$	$\{user = John\} \Rightarrow \{service = CALL, param = OUT\}$ means that user John requests the CALL service for outgoing calls

Table 3 Length-2 service rule classes

Class	Question	Rule template	Example
ST	Given a service (or a class of services), at which time period is it requested?	$\{service\} \Rightarrow \{time\}$	$\{service = WEATHER\} \Rightarrow \{time = morning\}$ means that weather forecasts are requested in the morning
SD	Given a service (or a class of services), in which period of the year is it requested?	$\{service\} \Rightarrow \{date\}$	$\{service = WEATHER\} \Rightarrow \{date = June\}$ means that weather forecasts are requested during June
SP	Given a service (or a class of services), where is it requested?	$\{service\} \Rightarrow \{place\}$	$\{service = CALL\} \Rightarrow \{place = office\}$ means that the CALL service is requested in the office
SPa	Given a service (or a class of services), which parameters are requested?	$\{service\} \Rightarrow \{params\}$	$\{service = CALL\} \Rightarrow \{param = OUT\}$ means that the CALL service is requested for outgoing calls

Table 4 Length-3 service rule classes

Class	Question	Rule template	Example
SPPa	Given a service (or a class of services), where is it requested and with which parameters?	$\{service\} \Rightarrow \{place, param\}$	$\{service = WEATHER\} \Rightarrow \{place = home, param = TODAY\_FORECAST\}$ means that daily weather forecasts are requested at home
SPT	Given a service (or a class of services), where is it requested and in which time period?	$\{service\} \Rightarrow \{place, time\}$	$\{service = WEATHER\} \Rightarrow \{place = home, time = evening\}$ means that weather forecasts are requested at home in the evening
SPD	Given a service (or a class of services), where is it requested and in which period of the year?	$\{service\} \Rightarrow \{place, date\}$	$\{service = WEATHER\} \Rightarrow \{place = home, date = summer\}$ means that weather forecasts are requested at home in summer
SPaD	Given a service (or a class of services), in what period of the year is it requested and with which parameters?	$\{service\} \Rightarrow \{param, date\}$	$\{service = CALL\} \Rightarrow \{param = OUT, date = week-end\}$ means that outgoing calls are requested during the week-end
SPaT	Given a service (or a class of services), in what time period is it requested and with which parameters?	$\{service\} \Rightarrow \{param, time\}$	$\{service = CALL\} \Rightarrow \{param = OUT, time = afternoon\}$ means that outgoing calls are requested during the afternoon
STD	Given a service (or a class of services), in what time and year periods is it requested?	$\{service\} \Rightarrow \{date, time\}$	$\{service = CALL\} \Rightarrow \{date = winter, time = afternoon\}$ means that calls are requested in winter during the afternoon

identifies the yearly period during which it is used, *SP* defines the usage location, while *SPa* identifies the service parameters. Table 4 reports a subset of the specializations of the rule templates in Table 3. For example, the specialized rule template $\{service\} \Rightarrow \{place, time\}$ adds either the place attribute to the *ST* rule template, or the time attribute to the *SP* rule

template. Hence, it specializes the context, defined in terms of both place and time, in which a given service is frequently requested.

Service rules support service providers in shaping the offered services to the user needs. The extracted knowledge can be exploited by service providers to (i) size system resources and (ii) define a default profile for new users. The exploitation of service rule templates to effectively support these activities is discussed in Sect. 6.3.2, which reports several service rule examples discovered in real context datasets and discusses their usage.

6 Experimental results

We evaluated the CAS-MINE framework by means of a large set of experiments addressing the following issues: (i) The characteristics of the mined knowledge (Sect. 6.2), (ii) the quality of the mined rules (Sect. 6.3), and (iii) the rule extraction performance (Sect. 6.4) in terms of execution time and number of mined rules.

All the experiments were performed on a 3.2 GHz Pentium IV system with 2 GB RAM, running Ubuntu 8.04. The CAS-MINE framework was implemented in the Python programming language [26].

6.1 Real context datasets

Three real context datasets, called *mDesktop*, *Recs*, and *TeamLife*, were provided by Telecom Italia Lab. Regarding privacy concerns related to real context data usage, please notice that (i) experimental data were collected from voluntary users that provide their whole informed consent on personal data treatment for this research project and (ii) real user names were hidden throughout the paper to preserve identities.

mDesktop dataset. The Telecom Italia mobile desktop application provides different services to users (e.g., weather forecast) through mobile devices. The *mDesktop* application provides 26 different services. The mDesktop dataset contains 4,487 records providing information on requested services and context (e.g., time and location, when available) of the logged users. The analyzed dataset includes the requests of 20 different (trial) users over a time period of one year.

To perform generalized rules mining, a taxonomy including the following rewriting rule trees has been defined.

- date → month → trimester → year
- timestamp → hour → timeslot (2 h timeslots)→ day period (AM/PM)
- service → class of service
- latitude:longitude → city → country
- phone number → call type (PERSONAL/BUSINESS)

We also considered different rewriting rule trees for the timestamp attribute. In particular, we considered different timeslots (e.g., 4 h timeslots and 8 h timeslots), which provided similar analysis results.

Recs dataset. The *Recs* system provides recommendations to users on restaurants, museums, movies, and other entertainment activities. Each user can request a recommendation, enter a score, or update a score for an entertainment center. The *Recs* system provides these three services to the end users. The analyzed dataset was obtained by logging the requests of 20 users and their locations over a time period of three months. The dataset contains 5,814 records. For the *Recs* dataset, the following rewriting rule trees have been considered:

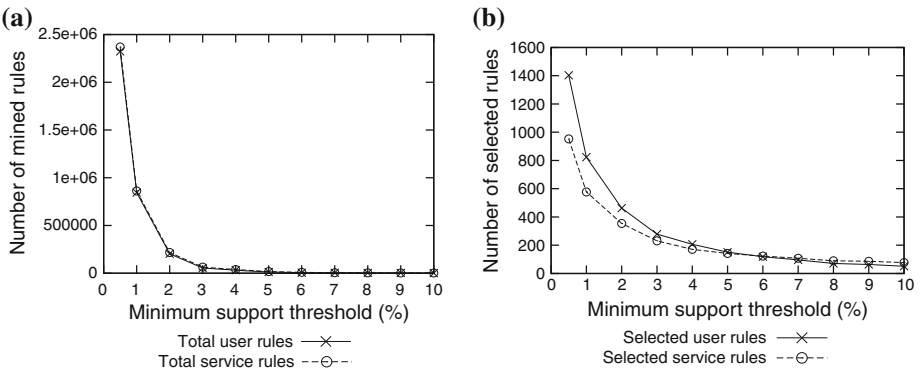


Fig. 3 Number of user and service rules by varying the minimum support threshold. Minconf = 0%. **a** *Recs* dataset: number of user and service rules. **b** *Recs* dataset: number of user and service rules selected by CAS-MINE

- date → month → trimester → year
- timestamp → hour → timeslot (2 h timeslots)→ day period (AM/PM)
- latitude:longitude → city → country

TeamLife dataset. The *TeamLife* dataset was generated by logging the activities of the users of the *TeamLife* system. The users of *TeamLife* can upload files, photos, or videos and share them with the other users. Four services are offered, and the logged users of the *TeamLife* system are 20. Also this dataset includes context information, in particular time and location of the users. The dataset includes 1,197 user requests collected over a time period of three months. For the *TeamLife* dataset, the same taxonomy of the *Recs* dataset has been exploited.

6.2 Characteristics of the rules mined by CAS-MINE

To characterize the rules discovered by CAS-MINE , we analyze the following issues: (i) The effect of the support threshold on the number of extracted patterns (Sect. 6.2.1), (ii) the impact of the generalization process (Sect. 6.2.2), and (iii) the rule distribution among templates (Sect. 6.2.3).

6.2.1 Effect of the support threshold

Since the minimum support threshold enforced during the mining step significantly affects the number of extracted rules, in Fig. 3 we report both (i) the number of user and service rules and (ii) the number of user and service rules selected by CAS-MINE (i.e., the rules belonging to the classes defined in Sect. 5.2). The analysis was performed without enforcing any minimum confidence threshold. Since the obtained results are comparable for all the three datasets, *Recs* is discussed as representative one (see Fig. 3).

The number of mined rules significantly increases for minimum support values lower than 1% (see Fig. 3a). Hence, it becomes difficult to exploit the extracted knowledge to create user and service profiles, since too many rules are available for each user and service. However, many rules either represent irrelevant information from an applicative point of view or are (longer) specializations of other rules. By exploiting the user and service templates presented in Sect. 5.2, the number of selected rules (see Fig. 3b) significantly decreases and becomes manageable. The number of selected rules is up to three orders of magnitude smaller than

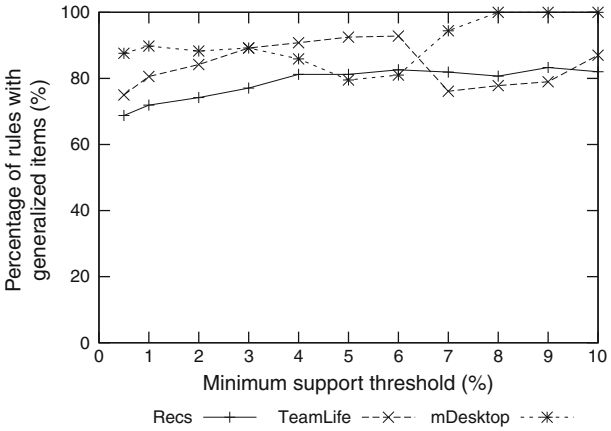


Fig. 4 Percentage of rules with generalized items in the set of rules selected by CAS-MINE when varying the minimum support threshold. Minconf = 0%

the total number of extracted user and service rules for the *Recs* dataset (see Fig. 3b). For the *TeamLife* and *mDesktop* datasets (detailed charts are not reported for lack of space), the number of selected rules is on average at least an order of magnitude smaller than the total number of extracted user and service rules, independently of the value of the minimum support threshold. Hence, CAS-MINE templates allow selecting a smaller set of rules that are interesting also from an applicative point of view. In particular, discarded rules are, for a large majority, specializations of other rules. A reduced number of rules include attribute combinations deemed as not relevant by the analysis of a domain expert. Some interesting applications of the selected rules are discussed in Sect. 6.3.

6.2.2 Impact of generalization

Figure 4 shows, for different settings of minimum support and for all datasets, the percentage of rules including at least one generalized item on the set of rules extracted by CAS-MINE . For all datasets, the percentage of rules containing generalized items is at least equal to 70%.

Since in the CAS-MINE framework infrequent items are aggregated during the extraction process, the percentage of generalized rules increases when the support threshold is increased. When very high support thresholds (e.g., 10%) are enforced, most extracted rules include generalized items belonging to the top levels of the taxonomies (e.g., `location`: ITALY $\Rightarrow$ `date`: 2008). These rules are usually too general to provide interesting knowledge. Differently, when lower support thresholds are enforced (e.g., in the range 1–4%), the extracted generalized rule set includes also non-top level elements of the taxonomy and more interesting knowledge is mined.

The extraction of generalized rules allows highlighting correlations that traditional association rules would hide because of their low support. The CAS-MINE approach allows a small set of low support association rules to be lazily aggregated into a higher support generalized association rule satisfying the support threshold.

6.2.3 Rule distribution among rule templates

Extracted rules are analyzed by investigating how rules are spread among the classes defined in Sect. 5.2. Figure 5 shows, for all datasets, the number of extracted rules in each class with

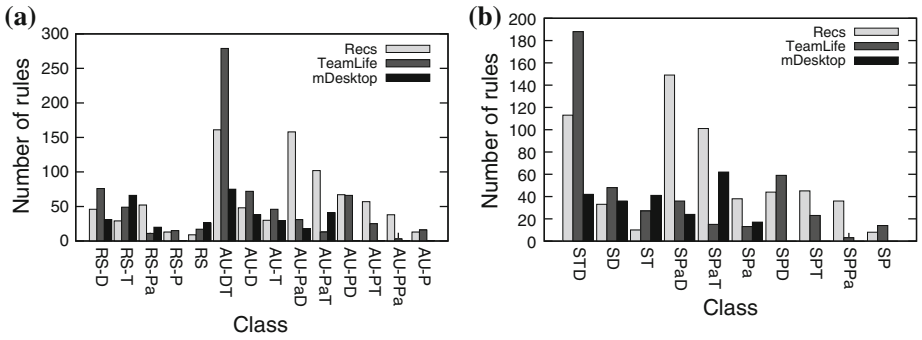


Fig. 5 Number of selected rules for each subclass of interest. Minsup = 1%. **a** User rules. **b** Service rules

a minimum support threshold equal to 1%. Results for user rules are reported in Fig. 5a, while the ones obtained for service rules are reported in Fig. 5b. For the *Recs* and *TeamLife* datasets at least one rule is extracted for each class of user rules (see Fig. 5a). Differently, for the *mDesktop* dataset some classes are empty. Since a very large number of user requests in the log file of the *mDesktop* application do not report the user location, the top level items in the place hierarchy are characterized by a support lower than 1%. Hence, rule templates characterized by the place attribute in the head are not populated.

For user rules (see Fig. 5a), the class with the largest number of rules is *AU-DT*, characterized by template $\{user\} \Rightarrow \{date, time\}$. The cardinality of *AU-DT* is larger than that of its shorter versions *AU-D* and *AU-T*. This effect is due to the peculiar characteristics of the date and time attributes. In particular, both the date and time attributes (i) are always specified together in the timestamp of the logged event, (ii) are characterized by a number of distinct values larger than the number of distinct values of the other attributes, and (iii) are characterized by hierarchies with four levels. Thus, the generalization process generates many relevant combinations of these attribute values, which yield the described behavior. In general, any attribute characterized by a large number of distinct values and a deep hierarchy (tree) may generate a large set of rules.

The number of extracted service rules for each dataset and class is reported in Fig. 5b. Similarly to user rules, also for service rules the templates including the date and time attributes are the most populated. In particular, the time and date combination (template *STD*) and the combinations with the param attribute (templates *SPaD* and *SPaT*) generate on average many rules.

6.3 Quality of the mined generalized rules

The rule quality analysis is focused on defining interestingness measures able to represent both the significance and utility of the extracted knowledge. Different objective measures [34] (e.g., lift, Pearson correlation coefficient, Jaccard measure, Mutual information) can be exploited to rank the extracted patterns according to their degree of interest. Then, only high ranked rules are presented to the analyst. In CAS-MINE , the lift measure is used to highlight the most interesting rules and is discussed in Sect. 6.3.1. The validation of the extracted knowledge, performed by domain experts, i.e., employees of Telecom Italia, is discussed in Sect. 6.3.2. The domain experts that manage the analyzed services acknowledged that the rules discovered by CAS-MINE represent valuable knowledge for both user and service profiling.

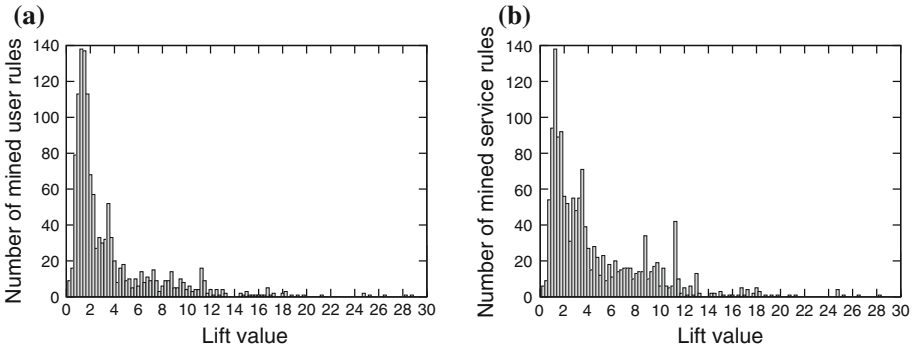


Fig. 6 *mDesktop* dataset: number of rules depending on the lift value. Minsup = 1%, Minconf = 0%. **a** *mDesktop* dataset: number of user rules. **b** *mDesktop* dataset: number of service rules

6.3.1 Ranking interesting rules by means of the lift measure

As discussed in Sect. 5.1, the lift measure is used by CAS-MINE to rank mined rules and to highlight the most interesting ones. The distribution of lift values on rules extracted on *mDesktop* dataset is analyzed as a rapresentative example. Fig. 6 shows (i) the histogram of the number of mined user rules (Fig. 6a) and (ii) the histogram of the number of mined service rules (Fig. 6b) depending on their lift value. Rules were extracted by setting the minimum support threshold to 1% and without enforcing any confidence threshold (i.e., Minconf = 0%). Rule numbers for lift values greater than 30 are not reported to enhance readability of the plot. The numbers of user and service rules with lift greater than 30 are 43 and 40, respectively.

Figure 6 shows that the majority of the mined rules is positively correlated (i.e., lift value greater than 1) in both charts. Lift values greater than 10 highlight a reduced set of rules worth a careful inspection. Some of them will be discussed in Sect. 6.3. A limited percentage of extracted rules is negatively correlated (i.e., 7.9% of user rules and 4.6% of service rules). Also in this case, lift values below 0.5 highlight a small set of negatively correlated rules. Some of them are discussed in Sect. 6.3.

6.3.2 Domain expert validation

The extracted rules have been analyzed by domain experts to assess the effectiveness of the CAS-MINE framework in discovering interesting and useful knowledge. The domain experts suggested possible usage scenarios for context-aware user and service profiling.

Habits of specific users (or user categories). The habits of users may be characterized by some kind of recurrence. For example, the following rules allow to discover valuable knowledge about a generic user of the *mDesktop* application, named *Rossi*. [2] They have been mined by enforcing a support threshold equal to 1% (i.e., absolute threshold = 45).

1. Class RS
   (a) `user`: Rossi $\Rightarrow$ `service`: CALL ($sup = 1.3\%, conf = 53\%, lift = 41.5$)
   (b) `user`: Rossi $\Rightarrow$ `service`: SMS ($sup = 1.1\%, conf = 47\%, lift = 41.5$)
2. Class $AU - T$

[2] Due to privacy concerns actual individual names are not provided.

(a) `user`: Rossi ⇒ `hour`: PM ($sup = 2.3\%, conf = 94\%, lift = 25.4$)

The first two above rules belong to the RS rule template. They highlight that user *Rossi* is interested in two specific services, CALL and SMS, with rule confidence close to 50%. They provide relevant knowledge on this user attitudes. If a larger support threshold is enforced, for instance 2% (absolute threshold = 90), the following rule is extracted.

– `user`: Rossi ⇒ `service`: Communication ($sup = 2.4\%, conf = 100\%, lift = 22.8$)

This rule, with confidence 100%, is a high-level grouping of the two former rules. It shows that *Rossi* is exclusively interested in the *Communication* service superset, which groups the CALL and SMS services.

Rule templates, described in Sect. 5.2, also allow characterizing different user habits. They entail (i) The service type users are mainly interested in, (ii) the context in which requests are commonly submitted, and (iii) the parameters that are frequently used. For instance, rule 2(a) highlights an intensive system usage by user *Rossi* during the afternoon/evening (confidence 94%).

Profiling user habits by exploiting negative correlation. The following rule, discovered in the *TeamLife* dataset, belongs to class RS, but, differently from previous rules, it is characterized by negative correlation (lift lower than 1).

– `user`: Verdi ⇒ `service`: PHOTO ($sup = 1.3\%, conf = 12\%, lift = 0.16$)

This rule shows that the user *Verdi* frequently uses the PHOTO service (the support of the rule is 1.3%). However, since the lift value is close to 0, it means that *Verdi* uses the PHOTO service less than expected. A user specific marketing action may target *Verdi* to promote the PHOTO service.

This information can also be exploited for cross selling purposes. Given a frequently used service, e.g., the FILE service in the positively correlated rule below

– `user`: Verdi ⇒ `service`: FILE ($sup = 9.94\%, conf = 86.9\%, lift = 6.30$)

a cross selling marketing action could consider several services belonging to the same aggregate group and select, as a service to be promoted, a negatively correlated service in the same group. Advanced knowledge on user habits can thus be exploited to generate promotions of (similar) rarely requested services (e.g., the PHOTO service for user *Verdi*).

Profiling services. Service rules highlight frequently used services and frequently asked parameters for each service, independently of the specific user who submits the requests. In the *mDesktop* dataset, by enforcing a minimum support threshold equal to 1% (absolute threshold = 45), the following generalized rules are extracted.

1. class SD
   – `service`: TWITTER ⇒ `month`: February ($sup = 2.9\%, conf = 31\%, lift = 1.9$)
2. class SPa
   (a) `service`: CALL ⇒ `inout`: OUT ($sup = 1.1\%, conf = 89\%, lift = 48.9$)
3. class SP
   (a) `service`: TLWIDGET ⇒ `location`: Turin ($sup = 1.2\%, conf = 50\%, lift = 5.0$)

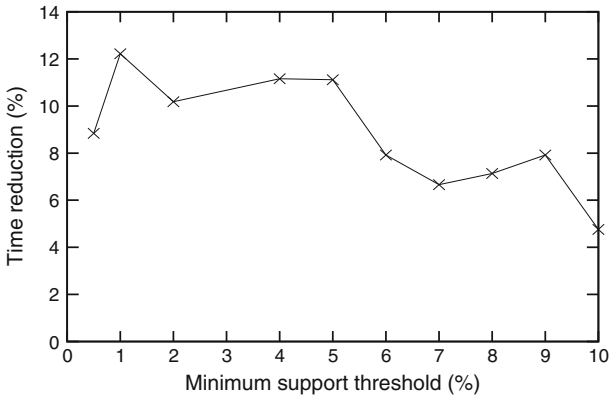


Fig. 7 *R*ecs dataset: comparison between GENIO and Cumulate

The first rule, belonging to the *SD* class, describes the period of usage of a specific service. Similar rules can be exploited to suggest default services, which could be temporally updated during the year depending on the services expected to be interesting in the considered period. The second rule, belonging to the *SPa* class, highlights the correlation between a service type and its parameters. In this case, call services are mainly exploited to perform outgoing calls. Finally, the last rule, belonging to class *SP*, describes the location in which a specific remote service is commonly used. This information could be useful both to size the provider system and to promote services in particular cities or regions.

6.4 Rule extraction performance

We analyzed the performance of the GENIO algorithm by both comparing its performance with a traditional generalized rule miner [29] and analyzing the scalability of the proposed approach.

A set of experiments was performed on the three real datasets to compare the difference, in terms of execution time and number of mined rules, between GENIO and a traditional generalized rule mining algorithm, i.e., Cumulate [29]. Since similar results were obtained for all datasets, *Recs* is discussed as a representative example. Since GENIO adopts a lazy taxonomy evaluation, it generates fewer itemsets, and hence rules, with respect to Cumulate. Due to the correlation between the execution time and the number of extracted rules, GENIO is also faster than Cumulate. The time reduction, in percentage, varies in the range [5–12%], depending on the minimum support threshold value (see Fig. 7), while the corresponding rule set reduction is between 37 and 52%. Usually, higher reduction time values are obtained when low support threshold values are enforced (in the range [1–5%]). As expected, when a high support threshold is enforced the two algorithms mine almost the same rule set. Hence, also the execution time of the two algorithms becomes similar when the support threshold increases.

The reported results show that the usage of GENIO, instead of a traditional generalized rule mining algorithm such as Cumulate [29], allows reducing both the execution time and the number of discovered rules.

We also analyzed the scalability of the rule mining algorithm with respect to the cardinality of the dataset on synthetic datasets generated by means of the TPC-H generator [36]. By varying the scale factor parameter, tables with different cardinalities are generated.

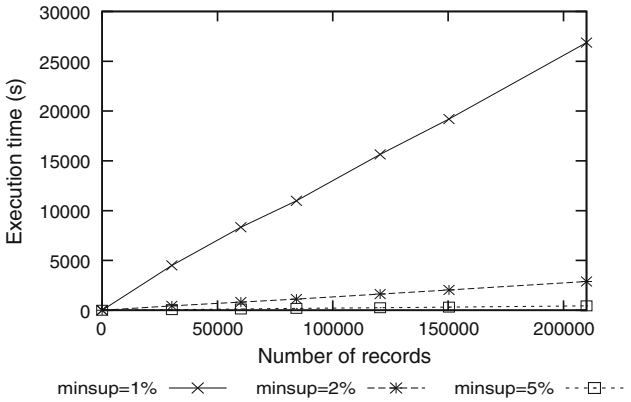


Fig. 8 Scalability of the GENIO algorithm on the TPC-H datasets

We generated datasets of size ranging from 30,000 to 210,000 records with 12 categorical attributes. Generalized itemsets have been mined from the *lineitem* table, while the *part*, *nation*, and *region* tables provided the taxonomy on line items.

Figure 8 plots the extraction time for different support values. It shows that the proposed algorithm scales well also on large datasets. Since the number of extracted itemsets grows for low support values (e.g., 1%), the generalization process becomes computationally more expensive. However, the overall CPU time is still acceptable for the lowest considered support on the largest dataset.

We also separately analyzed the execution time of the two steps of the mining activity (i.e., itemset and rule mining). On average, the execution time of the itemset mining step typically accounts for more than 95% of the total execution time, while the remaining time is devoted to the rule mining step.

7 Related work

An in-depth literature review focused on context-aware systems has been presented and discussed in [17], in which a classification of the most significant related papers is proposed. In particular, the analyzed papers are classified according to the architectural layer of a general context-aware system to which each paper refers (e.g., application layer, middleware layer, network layer). Differently, in [6] the authors propose a survey focused on data-oriented context models. Firstly, a set of currently available context models is described, and then a comprehensive evaluation framework is introduced to allow application designers to select the appropriate context model for a given target application. Main contextual information retrieval evaluation methodologies are also discussed in [33].

The usage of statistical and machine learning techniques (e.g., rule induction, neural networks, Bayesian networks) or data mining techniques (e.g., classification algorithms) has been proposed to exploit context data in building more accurate predictive user models (e.g., [24,35], and [42]) and user profiles (e.g., [23]). Usually, different service and application models are tailored to the user and to the situation in which he/she is involved by means, for example, of rule based or probabilistic approaches. These models are then exploited to (i) suggest applications and services depending on what are the user interests in his/her current

situation [37], (ii) perform customer segmentation [19], and (iii) personalize information retrieval tasks [10,21]. Differently from previously cited works, we address the problem of user and service profiling by means of generalized association rules, which are classified in semantic groups by exploiting a set of rule templates.

Integration of context information and data mining is also discussed in [28]. However, in the proposed context-based data mining framework, the context information is exclusively used to integrate multiple datasets, thus allowing consolidated mining on multiple physical datasets. Hence, the context knowledge neither directly drives the mining algorithms nor guides the selection of the mined patterns. Contextual information has been recently adopted in [28] for interactive postmining of association rules [1] driven by both ontologies and ad-hoc rule schemas [22], respectively representing user knowledge and expectations.

Context-awareness in the specific domain of mobile applications has been addressed from many different points of view. For example, context information has been used to analyze the collaboration between a mobile terminal user and another party [11] (i.e., another user or a mobile service) or to address context-based data reduction in mobile environments [15] for tailoring a service to both the current user context and the characteristics of the used mobile device. The analysis of the logs containing user locations, provided by the mobile devices, has been also performed by means of data mining algorithms. For example, [20] presents a location-based recommendation system for mobile devices which performs context-based data mining by means of a decision tree classification algorithm, while [37] addresses context-aware service and location pattern discovery in mobile Web environments by means of multiple-level association rules. In [37], the authors exploit a taxonomy composed of only two hierarchies (i.e., the location and the service hierarchies). Other context information (e.g., time and date) is not considered. Furthermore, patterns for location-based service provisioning are extracted by means of an exhaustive taxonomy evaluation. Differently from [37], the approach proposed in this paper entails (i) a lazy taxonomy evaluation to prevent redundant knowledge mining followed by post-pruning, (ii) the exploitation of a richer taxonomy possibly including a hierarchy for each available attribute (e.g., time, date, service parameters), (iii) an efficient extraction algorithm, exploiting the characteristics of structured datasets to effectively prune candidates, (iv) an effective categorization of the discovered knowledge by means of rule templates, and (v) a wider model applicability to general context-aware services outside the scope of the mobile Web environment.

A parallel effort has been devoted to the design and development of novel algorithms to efficiently extract generalized association rules. This issue was firstly addressed in [29,31] to perform market basket analysis. The first generalized association rule mining algorithm [29] generates itemsets by considering, for each item, its parents in the hierarchy. Hence, candidate frequent itemsets are generated by exhaustively evaluating the taxonomy and, thus, by producing a large amount of redundant patterns. Exhaustive generalized rule mining has been preliminary approached in [29] by applying traditional rule mining on a preprocessed source dataset, in which transactions were preliminarily extended by adding all the possible generalizations of the relative data items. However, a post-pruning step is required to prune redundant rules (i.e., rule containing multiple (generalized) items belonging to the same attribute) for making mined knowledge manageable by domain experts.

Generalized rule mining from both categorical and quantitative data was proposed in [30], by extending the concept of boolean association rules introduced in [1]. While categorical attributes are aggregated by following a user provided taxonomy, quantitative data are aggregated by means of a data-dependent information loss measure.

One step further toward a more efficient extraction process for generalized association rules was based on new optimization strategies [12,13,16]. In [16], faster support counting

is provided by exploiting the TID intersection computation, which is common in algorithms designed for the vertical data format [41]. Differently, in [12,13] an optimization based on a top–down hierarchy traversal is proposed. It identifies in advance itemsets that cannot be frequent in the dataset by means of the Apriori principle. The discovery of interesting multiple-level association rules is driven by a level-dependent multiple support threshold enforcement when level-sharing itemsets are extracted. However, multiple support thresholds, jointly with level-sharing mining, are shown to be effective only in specific application contexts in which a suitable parameter setting can be devised in advance. Furthermore, the algorithms proposed in [12,13] still show a limited pruning effectiveness on redundant patterns. The scalability and extraction time issues in generalized association rule mining have been also addressed in [16,25,32] by, respectively exploiting in [16] a TID intersection computation for faster support counting, in [25] a FP-tree-based generalized rule mining algorithm, and in [32] parent-child and subset-superset relationships in the lattice of generalized itemsets.

All the state-of-the-art generalized (multi-level) association rule mining approaches perform a similar exhaustive taxonomy evaluation by extracting most of the frequent patterns at any level of abstraction and, thus, showing a limited pruning effectiveness. A huge amount of rules is mined and valuable pattern discovery is, usually, left as a post-processing step, thus requiring the enforcement of high support thresholds. CAS-MINE aims at overcoming this issue by means of the GENIO algorithm, which lazily evaluates analyst-provided taxonomies, thus preventing redundant knowledge extraction. GENIO effectiveness in mining a compact set of interesting patterns is exploited for supporting business decisions in both user and service profiling.

The idea of exploiting generalized association rules for context-aware user and service profiling was firstly introduced in [5]. This paper significantly extends the preliminary version of the CAS-MINE framework presented in [5]. Furthermore, the CAS-MINE framework has been thoroughly evaluated by means of an in-depth experimental validation that demonstrates the effectiveness of the proposed approach in discovering interesting generalized association rules for user and service profiling from real context datasets provided by Telecom Italia.

8 Conclusion and future works

Context-aware applications exploit implicit context factors to enhance explicit user requests by offering personalized services depending on the current application context of each user. In this paper, the CAS-MINE framework has been proposed to support context-aware user and service profiling. To provide a high-level abstraction of both user habits and service characteristics depending on the context, generalized association rules are exploited. The mining process is driven by analyst-provided taxonomies on different attributes to prevent discarding relevant but infrequent knowledge, while preserving the compactness of the mined rule set. Furthermore, different rule templates have been devised to organize extracted rules in semantic classes. Experimental validation has shown the effectiveness of the CAS-MINE framework in extracting and selecting valuable patterns to support user and service profiling in context-aware applications.

Future developments of this work will address: (i) the enforcement of the rule template constraints directly in the mining process, (ii) the automatic inference of meaningful taxonomies from the input context datasets, similarly to [39], and their optimization to effectively drive the generalization process, (iii) the extension of the static CAS-MINE framework to dynamic knowledge mining, by following an approach similar to [27], and (iv) the application of the opportunistic generalization approach to the well-known FP-Growth algorithm [14].

Acknowledgments The authors would like to thank Vincenzo D'Elia for fruitful discussions and for implementing the Genio algorithm. The work was supported by the Telecom Italia grant T-010.

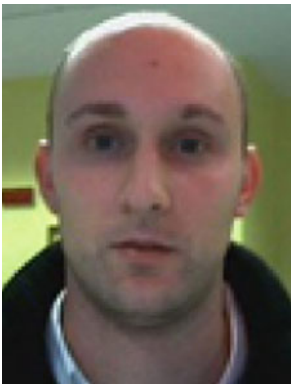
References

1. Agrawal R, Imielinski T, Swami AN (1993) Mining association rules between sets of items in large databases. In: Buneman P, Jajodia S (eds) SIGMOD conference. ACM Press, New york, pp 207–216
2. Agrawal R, Srikant R (1994) Fast algorithms for mining association rules in large databases. In: Bocca JB, Jarke M, Zaniolo C (eds) VLDB. Morgan Kaufmann, Los Altos, pp 487–499
3. Antonie M-L, Zaïane OR, Coman A (2001) Application of data mining techniques for medical image classification. In: Zaïane OR, Simoff SJ (eds) MDM/KDD. University of Alberta, Edmonton, pp 94–101
4. Baralis E, Cagliero L, Cerquitelli T, D'Elia V, Garza P (2010) Support driven opportunistic aggregation for generalized itemset extraction. In: IEEE conference of intelligent systems. IEEE, pp 102–107
5. Baralis E, Cagliero L, Cerquitelli T, Garza P, Marchetti M (2009) Context-aware user and service profiling by means of generalized association rules. In: Velásquez JD, Ríos SA, Howlett RJ, Jain LC (eds) KES (2), vol 5712 of Lecture Notes in Computer Science. Springer, Berlin, pp 50–57
6. Bolchini C, Curino C, Quintarelli E, Schreiber FA, Tanca L (2007) A data-oriented survey of context models. SIGMOD rec 36(4):19–26
7. Bradley NA, Dunlop MD (2005) Toward a multidisciplinary model of context to support context-aware computing. Hum Comput Interact 20(4):403–446
8. Byun H, Cheverst K (2004) Utilizing context history to provide dynamic adaptations. Appl Artifici Intell 18(6):533–548
9. Cong G, Tung AKH, Xu X, Pan F, Yang J (2004) Farmer: finding interesting rule groups in microarray datasets. In: Weikum G, König AC, Deßloch S (eds) SIGMOD conference. ACM, pp 143–154
10. Daoud M, Tamine-Lechani L, Boughanem M (2009) Towards a graph-based user profile modeling for a session-based personalized search. Knowl Inf Syst 21(3):365–398
11. Häkkilä J, Mäntyjärvi J (2005) Collaboration in context-aware mobile phone applications. HICSS, IEEE Computer Society, p 33a
12. Han J, Fu Y (1995) Discovery of multiple-level association rules from large databases. In: Dayal U, Gray PMD, Nishio S (eds) VLDB. Morgan Kaufmann, Los Altos, pp 420–431
13. Han J, Fu Y (1999) Mining multiple-level association rules in large databases. IEEE Trans Knowl Data Eng 11(5):798–804
14. Han J, Pei J, Yin Y (2000) Mining frequent patterns without candidate generation. In: Chen W, Naughton JF, Bernstein PA (eds) SIGMOD conference. ACM, New york, pp 1–12
15. Heuer A, Lubinski A (1998) Data reduction—an adaptation technique for mobile environments. Interactive Applications of mobile Computing (IMC'98), pp 1–2
16. Hipp J, Myka A, Wirth R, Güntzer U (1998) A new algorithm for faster mining of generalized association rules. In: Zytkow JM, Quafafou M (eds) PKDD, vol 1510 of Lecture Notes in Computer Science. Springer, Berlin, pp 74–82
17. Hong J, Suh E, Kim S-J (2009) Context-aware systems: a literature review and classification. Expert Syst Appl 36(4):8509–8522
18. Jameson A (2001) Modelling both the context and the user. Pers Ubiquitous Comput 5(1):29–33
19. Jiang T, Tuzhilin A (2009) Improving personalization solutions through optimal segmentation of customer bases. IEEE Trans Knowl Data Eng 21(3):305–320
20. Lee B-H, Kim H-N, Jung J-G, Jo G (2006) Location-based service with context data for a restaurant recommendation. In: Bressan S, Kü ng J, Wagner R (eds) DEXA, vol 4080 of Lecture Notes in Computer Science. Springer, pp 430–438
21. Leung KW-T, Lee DL (2010) Deriving concept-based user profiles from search engine logs. IEEE Trans Knowl Data Eng 22(7):969–982
22. Liu B, Hsu W, Mun L-F, Lee H-Y (1999) Finding interesting patterns using user expectations. IEEE Trans Knowl Data Eng 11(6):817–832
23. Nurmi P, Salden AH, Lau SL, Suomela J, Sutterer M, Millerat J, Martin M, Lagerspetz E, Poortinga R (2006) A system for context-dependent user modeling. In: Meersman R, Tari Z, Herrero P (eds) OTM workshops (2) vol 4278 of Lecture Notes in Computer Science. Springer, pp 1894–1903
24. Oliver N, Garg A, Horvitz E (2004) Layered representations for learning and inferring office activity from multiple sensory channels. Comput Vis Image Underst 96(2):163–180
25. Pramudiono I, Kitsuregawa M (2004) Fp-tax: tree structure based generalized association rule mining. In: Das G, Liu B, Yu PS (eds) DMKD. ACM, New york, pp 60–63

26. Python (2009) Python website. http://www.python.org
27. Shen B, Yao M, Wu Z, Gao Y (2010) Mining dynamic association rules with comments. Knowl Inf Syst 23(1):73–98
28. Singh S, Vajirkar P, Lee Y (2003) Context-based data mining using ontologies. In: Song I-Y, Liddle SW, Ling TW, Scheuermann P (eds) ER, vol 2813 of Lecture Notes in Computer Science. Springer, pp 405–418
29. Srikant R, Agrawal R (1995) Mining generalized association rules. In: Dayal U, Gray PMD, Nishio S (eds) VLDB. Morgan Kaufmann, Los Altos , pp 407–419
30. Srikant R, Agrawal R (1996) Mining quantitative association rules in large relational tables. In: Jagadish HV, Mumick IS (eds) SIGMOD conference. ACM Press, pp 1–12
31. Srikant R, Vu Q, Agrawal R (1997) Mining association rules with item constraints. In: Heckerman D, Mannila H, Pregibon D (eds) KDD. AAAI Press, Menlo Park, USA, pp 67–73
32. Sriphaew K, Theeramunkong T (2002) A new method for finding generalized frequent itemsets in generalized association rule mining. ISCC, IEEE Computer Society, pp 1040–1045
33. Tamine-Lechani L, Boughanem M, Daoud M (2010) Evaluation of contextual information retrieval effectiveness: overview of issues and research. Knowl Inf Syst 24(1):1–34
34. Tan P-N, Kumar V, Srivastava J (2002) Selecting the right interestingness measure for association patterns. In: Zaïane D, Goebel R (eds) KDD. ACM, New york, pp 32–41
35. Tapia EM, Intille SS, Larson K (2004) Activity recognition in the home using simple and ubiquitous sensors. In: Ferscha A, Mattern F (eds) Pervasive, vol 3001 of Lecture Notes in Computer Science. Springer, pp 158–175
36. TPC-H (2009) The TPC benchmark H. Transaction Processing Performance Council. http://www.tpc.org/tpch/default.asp
37. Tseng S-M, Tsui C-F (2004) Mining multilevel and location-aware service patterns in mobile web environments. IEEE Trans Syst Man Cybern Part B 34(6):2480–2485
38. Vajirkar P, Singh S, Lee Y (2003) Context-aware data mining framework for wireless medical application. In: Maník V, Retschitzegger W, Stepánková O (eds) DEXA, vol 2736 of Lecture Notes in Computer Science. Springer, pp 381–391
39. Woon WL, Madnick SE (2009) Asymmetric information distances for automated taxonomy construction. Knowl Inf Syst 21(1):91–111
40. Yang M, Wu Y, Hua G (2009) Context-aware visual tracking. IEEE Trans Pattern Anal Mach Intell 31(7):1195–1209
41. Zaki MJ, Parthasarathy S, Ogihara M, Li W (1997) New algorithms for fast discovery of association rules. In: Heckerman D, Mannila H, Pregibon D (eds) KDD. AAAI Press, Menlo Park, USA, pp 283–286
42. Zukerman I, Albrecht DW (2001) Predictive statistical models for user modeling. User Model User-adapt Interact 11(1–2):5–18

Author Biographies

Elena Baralis received the masters degree in electrical engineering and the PhD degree in computer engineering from the Politecnico di Torino. She has been a full professor in the Dipartimento di Automatica e Informatica, Politecnico di Torino, since January 2005. Her current research interests are in the field of databases, in particular, data mining, sensor databases, and bioinformatics. She has published more than 50 papers on journals and conference proceedings. She has served on the program committees of several international conferences and workshops, among which are VLDB, ACM CIKM, DaWak, ACM SAC, and PKDD. She has managed several Italian and EU research projects.



Luca Cagliero received the master's degree in computer and communication networks from the Politecnico di Torino in 2008. Since January 2009, he has been a PhD student in computer engineering in the Dipartimento di Automatica e Informatica, Politecnico di Torino. His current research interests are in the areas of data mining and database systems. In particular, he is investigating the application of novel classification and association rule mining approaches to very large databases.

Tania Cerquitelli received the masters degree in computer engineering and the PhD degree from the Politecnico di Torino, Torino, Italy, and the masters degree in computer science from the Universidad De Las Ameéricas Puebla. She has been a post-doctoral researcher in computer engineering in the Dipartimento di Automatica e Informatica, Politecnico di Torino since January 2007. Her research interests include the design of efficient algorithms to perform large-scale data mining, innovative data mining techniques for sensor readings, and novel algorithms to extract high-level abstraction of the mined knowledge (e.g., generalized association rules).

Paolo Garza received the masters and PhD degrees in computer engineering from the Politecnico di Torino. He has been an assistant professor (with non-tenure track position) at the Dipartimento di Elettronica e Informatica, Politecnico di Milano, since June 2010. His current research interests include data mining and database systems. In particular, he has worked on the classification of structured and unstructured data, clustering, and itemset mining algorithms.

Marco Marchetti graduated in Computer Science, started his professional career in July 1989 at CERN (European Organization for Nuclear Research) where he was involved in activities regarding "Human Machine Interface Systems". Since September 1991 he began his activity at ALENIA SPAZIO, where he contributed in the SAX (X-Ray SAT) Project. Since May 1994 he began his activity at TILAB, where he contributed in TIM-VAS Project, dealing with WAP Technology related issues. He was Project Leader in several Projects supporting the Telecom Italia Marketing in the innovation of Web Services Enterprises Offer. He also was involved in projects aiming at developing Info Mobility solutions over public mobile network (GPRS/GSM). He is currently involved in research projects aiming at developing context-aware platform and services dealing with Data Mining technology and recommender systems.