Misleading Generalized Itemset Discovery

Luca Cagliero*, Tania Cerquitelli*, Paolo Garza*, Luigi Grimaudo*

Dipartimento di Automatica e Informatica, Politecnico di Torino,
Corso Duca degli Abruzzi 24, 10129, Torino, Italy.

Abstract

Frequent generalized itemset mining is a data mining technique utilized to discover a high-level view of interesting knowledge hidden in the analyzed data. By exploiting a taxonomy, patterns are usually extracted at any level of abstraction. However, some misleading high-level patterns could be included in the mined set.

This paper proposes a novel generalized itemset type, namely the Misleading Generalized Itemset (MGI). Each MGI, denoted as $X \triangleright \mathcal{E}$, represents a frequent generalized itemset X and its set $\mathcal{E}$ of low-level frequent descendants for which the correlation type is in contrast to the one of X. To allow experts to analyze the misleading high-level data correlations separately and exploit such knowledge by making different decisions, MGIs are extracted only if the low-level descendant itemsets that represent contrasting correlations cover almost the same portion of data as the high-level (misleading) ancestor. An algorithm to mine MGIs at the top of traditional generalized itemsets is also proposed.

*Corresponding author. Tel.: +39 011 090 7084. Fax: +39 011 090 7099.
Email addresses: luca.cagliero@polito.it (Luca Cagliero), tania.cerquitelli@polito.it (Tania Cerquitelli), paolo.garza@polito.it (Paolo Garza), luigi.grimaudo@polito.it (Luigi Grimaudo)

Preprint submitted to

The experiments performed on both real and synthetic datasets demonstrate the effectiveness and efficiency of the proposed approach.
Keywords: Generalized Itemset Mining, Data Mining, Taxonomies, Mobile Data Analysis

1. Introduction

Generalized itemset mining [26] is an established data mining technique that focuses on discovering knowledge hidden in the analyzed data at different abstraction levels. By exploiting a taxonomy (i.e. a set of is-a hierarchies built over the analyzed data) the mining process entails discovering patterns, i.e. the frequent generalized itemsets, that (i) have a frequency of occurrence (support) in the analyzed data higher than or equal to a given threshold and (ii) can include items at any level of abstraction. Low-level itemsets represent rather specific and detailed data correlations for which the corresponding support is unlikely to exceed the given threshold. On the other hand, high-level (generalized) itemsets provide a high-level view of the underlying data correlations. Hence, they could represent, at a high granularity level, the knowledge that remains hidden at a lower abstraction level. The interestingness of an itemset is commonly measured in terms of the strength of the correlation between its items [1, 9, 25]. To evaluate itemset correlation, in this paper we exploit an established correlation measure, i.e. the Kulczynsky (Kulc) correlation measure [34]. This measure has recently been adopted to perform high-level itemset correlation analysis [7]. Itemset correlation values are usually clustered in three different correlation types. Specifically, if an itemset X occurs less than expected in the analyzed data (i.e. the item

correlation value is between 0 and a given threshold *max_neg_cor*) then X is said to be *negatively correlated*; if it occurs more than expected (i.e. the item correlation value is above a given threshold *min_pos_cor*) then X shows a *positive correlation*, otherwise (i.e. whenever there is neither a positive nor a negative item correlation) X is said to be *not correlated*. Unfortunately, to support domain experts in making decisions not all of the mined high-level patterns can be trusted. Indeed, some misleading high-level itemsets could be included in the mining result. A generalized itemset X is, to some extent, misleading if (some of) the low-level X's descendants have a correlation type in contrast to those of X.

For example, let us consider the structured dataset that is reported in Table 1. Each record contains the record identifier (rid), the city, and the product description. The itemset mining process can be driven by the taxonomy in Figure 1, which generalizes cities and products as the corresponding nations and product categories. Table 2 reports the set of frequent generalized itemsets that are mined by enforcing a support threshold min_sup=1 and two correlation thresholds max_neg_cor=0.65 and min_neg_cor=0.8. The frequent generalized itemset X={(Product, Wearing), (City, Italy)} has a positive correlation type, whereas its frequent low-level descendant itemset Y={(Product, T-shirt),(City, Rome)} is negatively correlated (see Table 2). To estimate the extent to which X is misleading we evaluate the percentage of dataset records that are covered by both X and any of its contrasting low-level correlations. For example, the record with rid 3 is covered by both X and Y. In other words, 25% of the records that are covered by {(Product, Wearing), (City, Italy)} are in common with those covered by {(Product,

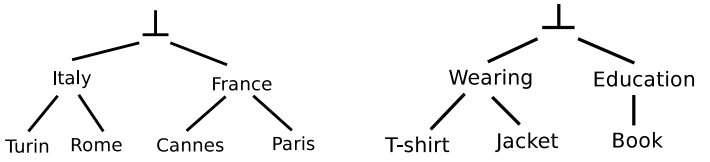
T-shirt),(City, Rome)}.

In this paper we propose: (i) a novel generalized itemset type, namely the Misleading Generalized Itemset (MGI); (ii) a MGI quality measure called Not Overlapping Degree (NOD) which indicates the extent to which the high-level pattern is misleading compared to its low-level descendants; and (iii) an approach to discovering a worthwhile subset of MGIs with NOD less than or equal to a maximum threshold max_NOD. Specifically, each MGI, hereafter denoted as $X \triangleright \mathcal{E}$, represents a frequent generalized itemset X and its set $\mathcal{E}$ of low-level frequent descendants for which the correlation type is in contrast to those of X. Experts need to analyze the misleading high-level data correlations separately from the traditional generalized itemsets and exploit such knowledge by making different decisions. To make this analysis possible, MGIs are extracted only if the low-level descendant itemsets that represent contrasting correlations cover almost the same portion of data as the high-level (misleading) ancestor X, i.e only if X represents a "clearly misleading" pattern. To do so, a maximum NOD constraint is enforced during the MGI mining process. Hence, unlike previous approaches (e.g. [7, 9]), we evaluate the degree of overlapping between the sets of records that are covered by a generalized itemset and its low-level (descendant) contrasting correlations. An algorithm to mine MGIs at the top of traditional generalized itemsets is also proposed.

The effectiveness of the proposed approach and the usability of the discovered patterns for supporting domain expert decisions are demonstrated by experiments performed on real-life data coming from two mobile applications and the UCI data repository [8]. Furthermore, the scalability of the

Table 1: Example dataset $\mathcal{D}$.

Id	City	Product
1	Turin	T-shirt
2	Turin	T-shirt
3	Rome	T-shirt
4	Paris	Jacket
5	Paris	Jacket
6	Cannes	Book
7	Turin	T-shirt



(a) Aggregation tree $AT_{location}$ defined on the Location attribute

(b) Aggregation tree $AT_{product}$ defined on the Product attribute

Figure 1: Example taxonomy built on $\mathcal{D}$'s attributes

algorithm has also been evaluated on synthetic datasets.

This paper is organized as follows: Section 2 introduces preliminary definitions; Section 3 formally states the MGI mining problem; Section 4 describes the MGI Miner algorithm; Section 5 discusses the performed experiments; Section 6 presents previous works and Section 7 draws conclusions and discusses some possible future developments of this work.

2. Preliminary definitions and notations

This paper addresses the problem of generalized itemset mining from structured data that are supplied with taxonomies. A structured dataset is a set of records. Each record is a set of items, which are defined as pairs (attribute_name, value). While attribute_name is the description of a data

Table 2: MGI mined from $\mathcal{D}$. min_sup = 1, max_neg_cor= 0.65, min_pos_cor= 0.80, and max_NOD = 100%.

Frequent generalized itemset (level≥2) [correlation type (Kulc value)]	**Frequent descendant [correlation type (Kulc value)]**	**Not overlapping degree (%)**
{(City, Italy)} [positive (1)]	{(City, Turin)} [positive (1)] {(City, Rome)} [positive (1)]	-
{(City, France)} [positive (1)]	{(City, Paris)} [positive (1)] {(City, Cannes)} [positive (1)]	-
{(Product, Wearing)} [positive (1)]	{(Product, T-shirt)} [positive (1)] {(Product, Jacket)} [positive (1)]	-
{(Product, Education)} [positive (1)]	{(Product, Book)} [positive (1)]	-
{(Product, Wearing), (City, Italy)} [positive (5/6=0.83)]	**{(Product, T-shirt), (City, Turin)} [positive (7/8=0.88)] {(Product, T-shirt), (City, Rome)} [negative (5/8=0.63)]**	**75**
{(Product, Wearing), (City, France)} [negative (1/2=0.50)]	**{(Product, Jacket), (City, Paris)} [positive (1)]**	**0**
{(Product, Education), (City, France)} [negative (2/3=0.66)]	**{(Product, Book), (City, Cannes)} [positive (1)]**	**0**

feature, value represents the associated information and belongs to the corresponding attribute domain. Since continuous attribute values are unsuitable for use in itemset mining, continuous values are discretized by a traditional preprocessing step [31]. For instance, Table 1 reports an example of structured dataset D that is composed of 3 attributes: the record identifier (rid), the city, and the product description.

A taxonomy is a set of is-a hierarchies built over the data attribute items. It consists of a set of aggregation trees, one or more for each dataset attribute, in which the items that belong to the same attribute domain are aggregated in higher level concepts. For example, let us consider the taxonomy that is reported in Figure 1. It includes two aggregation trees, one for each attribute in $\mathcal{D}$. By construction, we disregard the rid attribute for the subsequent

analysis. For each aggregation tree the leaf nodes are labeled with values belonging to the corresponding attribute domain, whereas each non-leaf node aggregates (a subset of) lower level nodes and is labeled with a value that is not in the attribute domain. Aggregation tree root nodes are labeled with the special value $\perp$. A pair (attribute_name, aggregation value), where aggregation_value is a non-leaf node label, is called *generalized item*. For instance, (City, France) is a generalized item that corresponds to a taxonomy non-leaf node which aggregates all of the French cities that occur in $\mathcal{D}$ (see Table 1 and Figure 1(a)). For the sake of simplicity, hereafter we consider only taxonomies that are composed of one aggregation tree per attribute.

A k-itemset (i.e. an itemset of length k) is defined as a set of k distinct items [2]. For instance, {(City, Turin), (Product, T-shirt)} is an example of itemset that occurs in $\mathcal{D}$ (see Table 1). Similarly, when dealing with structured datasets that are supplied with taxonomies, a generalized k-itemset is a set of k distinct items or generalized items. For instance, given the taxonomy reported in Figure 1, {(City, Italy), (Product, Wearing)} is an example of generalized 2-itemset.

Generalized itemsets are characterized by many properties [26]. For our purposes, we recall some notable properties in the following.

Coverage and support. A generalized itemset I is said to *cover* a given record $r_i \in \mathcal{D}$ if all of its (generalized) items are either contained in r_i or ancestors of items in r_i. I's support in $\mathcal{D}$ is defined as the ratio between the number of records in $\mathcal{D}$ that are covered by I and the total number of records in $\mathcal{D}$ [26]. A generalized itemset for which the support exceeds a given threshold min_sup is said to be *frequent*. For example,{(City, Italy), (Product, Wear-

ing)} has support $\frac{4}{7}$ in $\mathcal{D}$ because it covers the records with rids 1, 2, 3, and 7 (see Table 1). Given a set of generalized itemsets $\mathcal{I}$, for our purposes we also define the coverage of $\mathcal{I}$ with respect to $\mathcal{D}$, hereafter denoted as $cov(\mathcal{I}, \mathcal{D})$, as the ratio between the number of records in $\mathcal{D}$ that are covered by *any* itemset in $\mathcal{I}$ and the total number of records in $\mathcal{D}$. For example, together the itemsets {(City, Italy), (Product, Wearing)} and {(City, France), (Product, Wearing)} have coverage $\frac{6}{7}$ in $\mathcal{D}$, because they cover all records in $\mathcal{D}$ except for the one with rid 6. Given a single generalized itemset, from the above definitions it trivially follows that its coverage and support values in $\mathcal{D}$ are the same.

Level-sharing itemset. The level of an arbitrary (generalized) item i_j with respect to a taxonomy Γ is defined as the height of the Γ's subtree rooted in i_j. It indicates the item abstraction level according to the given taxonomy. Similar to [7, 14], we target the item correlations at same abstraction level, i.e. the itemsets that exclusively contain items with the same level. Such patterns are denoted as *level-sharing itemsets* [14]. The level of a level-sharing itemset I with respect to the taxonomy Γ, i.e. L[I, Γ], corresponds to that of any of its items.

Experts are expected to provide balanced taxonomy trees to effectively highlight contrasting correlations at different taxonomy levels. If the experts do not provide balanced taxonomy trees, as in [7], we rebalanced those taxonomy aggregation trees in the performed experiments. Specifically, given a taxonomy with maximal aggregation tree height H_{max}, for each aggregation tree with height $H < H_{max}$ we performed a depth-first visit. For each tree branch with depth less than H_{max} we added multiple copies of the top-level

item i as i's ancestors up to depth H_{max}.

Descent relationship. Given two generalized k-itemsets I_1 and I_2, I_1 is said to be a descendant of I_2, i.e. $I_1 \in \text{Desc}[I_2, \Gamma]$ if for every item $i_j \in I_1$ there exists an item $i_k \in I_2$ such that either i_j=i_k or i_j is a descendant of i_k with respect to the given taxonomy. For example, {(City, Turin), (Product, T-shirt)} is a descendant of {(City, Italy), (Product, Wearing)}.

Correlation. The itemset correlation measures the strength of the correlation between its items. In this paper, similar to [7], we evaluate the correlation of a generalized k-itemset I by means of the Kulczynsky (Kulc) correlation measure [34], which is defined as follows:

$$\text{kulc(I)} = \frac{1}{\text{k}} \sum_{j=1}^{\text{k}} \frac{sup(I, \mathcal{D})}{sup(i_j, \mathcal{D})} \tag{1}$$

where $sup(I, \mathcal{D})$ is I's support in $\mathcal{D}$ and i_j $[1 \leq j \leq k]$ is the j-th item in I. From Equation 1 it follows that Kulc values range between 0 and 1. Unlike many other traditional itemset correlation measures, Kulc has the null (transaction)-invariant property, which implies that the correlation measure is independent of the dataset size [34].

By properly setting maximum negative and minimum positive Kulc thresholds, hereafter denoted as *max_neg_cor* and *min_pos_cor*, the generalized itemsets may be classified as negatively correlated, uncorrelated, or positively correlated itemsets according to their correlation value. More specifically, generalized itemsets for which Kulc is between max_neg_cor and min_pos_cor consist of items that are not correlated with each other (i.e. their items are statistically independent), generalized itemsets for which Kulc is below

max_neg_cor show negative item correlation, whereas generalized itemsets for which Kulc is above min_pos_cor indicate a positive item correlation, i.e. their items co-occur more than expected. For the sake of brevity, we hereafter denote the above-mentioned correlation types as *uncorrelated*, *negative*, and *positive*, respectively.

3. The Misleading Generalized Itemset mining problem

Given a structured dataset $\mathcal{D}$ that is supplied with a taxonomy Γ and a minimum support threshold min_sup, the traditional frequent generalized itemset mining problem entails discovering all of the frequent generalized itemsets from $\mathcal{D}$.

Frequent generalized itemsets represent data correlations at different abstraction levels. On the one hand, low-level itemsets commonly represent rather specific and detailed data correlations. Unfortunately, they are unlikely to be frequent with respect to the enforced minimum support threshold. On the other hand, high-level itemsets provide a high-level viewpoint of the analyzed data, which could be useful for representing the infrequent knowledge at a higher abstraction level. However, some high-level itemsets could be deemed to be misleading, because their correlation type is in contrast to that of their low-level descendants. For instance, consider the example dataset and taxonomy reported in Table 1 and Figure 1, respectively. The frequent generalized itemset {(Product, Wearing), (City, Italy)} has a positive correlation type, whereas its frequent low-level descendant itemset {(Product, T-shirt), (City, Rome)} is negatively correlated (see Table 2). Since the type of the mined data correlation changes unexpectedly while performing a drill-down,

the high-level itemset is, to some extent, misleading.

To allow domain experts to discover and analyze the misleading high-level itemsets separately, we propose a new generalized pattern type, namely the M*isleading* G*eneralized* I*temset* (MGI). MGIs are patterns in the form $X \triangleright \mathcal{E}$, where X is a frequent generalized itemset of level $l \geq 2$ with either positive or negative correlation type, while $\mathcal{E}$ is the set of frequent level-$(l-1)$ X's descendants for which the correlation type is in contrast to that of X. A more formal definition follows.

Definition 3.1. MGI. *Let $\mathcal{D}$ be a structured dataset and Γ a taxonomy. Let min_sup be a minimum support threshold and max_neg_cor and min_pos_cor a maximum negative and a minimum positive correlation threshold. Let $\mathcal{LSGI}$ be the subset of frequent level-sharing generalized itemsets in $\mathcal{D}$ that are either positively or negatively correlated. Given a frequent level-sharing generalized itemset $X \in \mathcal{LSGI}$ of level $l \geq 2$, let Desc*[X,Γ] be the subset of level-$(l-1)$ X's descendants for which the correlation type is in contrast to that of X. An* MGI *is a pattern in the form $X \triangleright \mathcal{E}$, where $X \in \mathcal{LSGI}$ and $\mathcal{E}$=Desc*[X,Γ].*

For example, setting min_sup = 1, max_neg_cor=0.65, and min_pos_cor=0.8, the MGI {(Product, Wearing), (City, Italy)} ▷ {(Product, T-shirt), (City, Rome)} is mined from the example dataset in Table 1, because {(Product, Wearing), (City, Italy)} has a positive correlation (0.83), whereas its descendant itemset {(Product, T-shirt), (City, Rome)} is negatively correlated (0.63).

We define the level of an MGI $X \triangleright \mathcal{E}$ with respect to the input taxonomy Γ as X's level, i.e. $\mathrm{L}[X \triangleright \mathcal{E}, \Gamma] = \mathrm{L}[X, \Gamma]$. For example, {(Product, Wearing),

(City, Italy)} $\triangleright$ {(Product, T-shirt), (City, Turin)} is a level-2 MGI because {(Product, Wearing), (City, Italy)} has level 2.

Since a generalized itemset could have many low-level descendants that represent contrasting correlations, we evaluate the interest of an MGI $X \triangleright \mathcal{E}$ as the relative difference between the support of the ancestor generalized itemset X and the coverage of its low-level contrasting data correlations in $\mathcal{E}$. We denote this measure as the *Not Overlapping Degree* (NOD).

Definition 3.2. MGI**'s NOD measure.** *Let* $X \triangleright \mathcal{E}$ *be an* MGI*. Let* $sup(X, \mathcal{D})$ *be* X*'s support in* $\mathcal{D}$ *and* $cov(E, \mathcal{D})$ *the coverage of* $\mathcal{E}$ *in* $\mathcal{D}$*. The Not Overlapping Degree (NOD) of* $X \triangleright \mathcal{E}$ *is defined by:* $\frac{sup(X,\mathcal{D}) - cov(\mathcal{E},\mathcal{D})}{sup(X,\mathcal{D})}$.

Since the inequality $sup(X, \mathcal{D})$-$cov(\mathcal{E}, \mathcal{D}) \geq 0$ holds, it trivially follows that the MGI NOD values are between 0 and 1. The lower the NOD value is, the more significant the degree of overlapping between the contrasting low-level correlations in $\mathcal{E}$ and their common ancestor X. As an extreme case, when the contrasting descendant itemsets cover *every* record covered by X the MGI NOD value is 0. For example, the MGI {(Product, Wearing), (City, Italy)} $\triangleright$ {(Product, T-shirt), (City, Rome)} has a NOD value equal to $\frac{4-1}{4} = \frac{3}{4}$ because {(Product, Wearing), (City, Italy)} covers four records in $\mathcal{D}$ (i.e. the records with rids 1, 2, 3, and 7), whereas its descendant {(Product, T-shirt), (City, Rome)} covers one of them (i.e. the record with rid 3).

Experts could be interested in analyzing only the MGIs with a relatively low NOD value, because they represent clearly misleading high-level data correlations. Hence, we enforce a maximum NOD constraint to select only

the subset of MGIs with a NOD value less than or equal to a maximum NOD threshold max_NOD. As shown in Section 5, this worthwhile MGI subset is useful for supporting the expert-driven knowledge discovery process in a real-life application scenario.

Problem statement. Given a structured dataset $\mathcal{D}$, a taxonomy, a minimum support threshold, a maximum negative, and a minimum positive correlation threshold, and a maximum NOD threshold max_nod, the mining task addressed by this paper entails discovering from $\mathcal{D}$ all of the MGIs for which the NOD value is less than or equal to max_NOD.

4. The Misleading Generalized Itemset Miner algorithm

The Misleading Generalized Itemset Miner (MGI Miner) algorithm addresses the MGI mining problem that is stated in Section 3. The MGI extraction process entails the following steps: (i) Traditional frequent level-sharing generalized itemset mining and (ii) MGI extraction at the top of the previously extracted itemsets. MGI extraction is performed level-wise, i.e. level-1 MGIs are generated first. Next, at each step, MGIs with increasing level are generated until the top of the taxonomy is reached. Algorithm 1 reports a pseudo-code for the MGI Miner algorithm.

Frequent level-sharing itemset mining. Frequent level-sharing generalized itemsets are used to drive the MGI mining process (see line 1) because each MGI consists of a combination of them (see Definition 3.1). The traditional itemset extraction task is accomplished by an established projection-based itemset miner, i.e. the LCMv2 algorithm [12], which is an extension of the traditional FP-Growth algorithm [15]. Projection-based

Algorithm 1 MGI MINER algorithm

Input: a structured dataset $\mathcal{D}$, a taxonomy Γ, a maximum NOD threshold max_NOD, a minimum support threshold min_sup, a maximum negative and a minimum positive Kulc thresholds max_neg_cor and min_pos_cor
Output: the subset of all the MGIs $\mathcal{MGI}$

$\mathcal{LSGI}$ = mineTraditionalLevelSharingGeneralizedItemsets($\mathcal{D}$, Γ, min_sup)
$\mathcal{MGI} = \emptyset$
/ Generate* MGIs $X \triangleright \mathcal{E}$ *with level* $l > 1$ **/*
for l=2 to *maxlevel* **do**
  / for each frequent level-sharing generalized itemset one candidate* MGI *is generated */*
  for all X in $\mathcal{LSGI}[l]$ **do**
    / Create a level-*l *candidate* MGI $X \triangleright \mathcal{E}$ **/*
    insert the candidate MGI ($X \triangleright \mathcal{E}$) in $C[l]$
  end for
  / Populate the* $\mathcal{E}$ *set of the level-*l *candidate* MGI **/*
  for all it in $\mathcal{LSGI}[l-1]$ **do**
    / Retrieve the candidate itemset genit of level* l *that is ancestor of it and update genit.*$\mathcal{E}$ **/*
    $genit$ = retrieveAncestor($\mathcal{LSGI}[l]$,it,l,Γ);
    cor_type_genit=ComputeKulc($genit$,$\mathcal{D}$,max_neg_cor,min_pos_cor)
    cor_type_it=ComputeKulc(it,$\mathcal{D}$,max_neg_cor,min_pos_cor)
    / If the level-*$(l-1)$ *itemset it has a correlation type different from its ancestor genit then it must be added to genit.*$\mathcal{E}$ **/*
    if $cor\_type\_genit \neq cor\_type\_it$ **then**
      insert it into $genit.\mathcal{E}$
    end if
  end for
  / Select the level-*l *candidate* MGIs *with NOD less than or equal to max_NOD */*
  for all c in $C[l]$ **do**
    $c.NOD$ = ComputeNOD(c,$\mathcal{D}$,Γ);
    if $c.NOD \leq$ max_NOD **then**
      insert c into $\mathcal{MGI}[l]$
    end if
  end for
end for
return $\mathcal{MGI}$

itemset mining relies on the following steps: (i) creation and in-memory storage of an FP-tree-based dataset representation and (ii) frequent itemset extraction by recursively visiting the conditional FP-tree projections. We applied the following main modifications to a traditional FP-tree-based itemset miner [12]: (1) To efficiently cope with structured dataset, the itemset miner prevents the generation of the candidate itemsets that include couples of items corresponding to the same attribute. (2) To suit the traditional LCM implementation to generalized itemset mining, we adopted the strategy, first proposed in [26], of extending the dataset records by appending to

each record all of its item generalizations in Γ. (3) To prevent the generation of not level-sharing itemsets, the generation procedure of the conditional FP-tree projections related to a level-l item disregards the not level-l items. Frequent level-sharing generalized itemsets are stored in $\mathcal{LSGI}$ (line 1).

MGI **mining:** Once all the frequent level-sharing itemsets X are extracted, MGI Miner generates candidate MGIs in the form $X \triangleright \mathcal{E}$ and populates their $\mathcal{E}$ part with X's descendants for which the correlation type is in contrast to those of X. MGIs are mined by following a level-wise approach, i.e. climbing up the taxonomy stepwise until the top of the taxonomy is reached (lines 4-28). Performing a level-wise taxonomy evaluation prevents the need for multiple itemset scans. Indeed, level-l MGIs are generated from the sets of level-l and level-$(l-1)$ frequent level-sharing itemsets $\mathcal{LSGI}[l]$ and $\mathcal{LSGI}[l-1]$. While the level-l itemsets are used to populate the X part (lines 6-9), the level-$(l-1)$ itemsets that represent contrasting correlations are used to fill the $\mathcal{E}$ set (lines 11-20). Hence, while mining level-l MGIs all of the traditional frequent itemsets that have a level strictly less than $l-1$ can be discarded early. Finally, level-l MGIs for which the NOD value is less than or equal to max_NOD are selected and added to the output set (lines 22-27).

5. Experimental results

We performed a large suite of experiments to evaluate: (i) the usefulness of the MGIs mined from data that were acquired from a real-life context with the help of a domain expert (see Section 5.2); (ii) the impact of the algorithm parameters on the MGI Miner performance on benchmark datasets (see

Section 5.3); and (iii) the MGI Miner algorithm scalability on synthetic datasets (see Section 5.4).

The experiments were performed on a 3.30 GHz Intel® Xeon® CPU E31245 PC with 16 GB main memory running Linux (kernel 3.2.0).

5.1. Datasets

A brief description of the evaluated datasets is reported in the following paragraphs.

Real-life mobile datasets

To validate the usefulness of the proposed patterns, we ran experiments on two real mobile datasets that were collected by a research hub of an international leader in the telecommunication area. The two datasets were acquired by logging the user requests for two different mobile applications, namely *Recs* and $TeamLife$. The applications provide users with a set of services (e.g. weather forecasting, restaurant recommendations, and photo and movie uploads) through their mobile devices (i.e. smartphones or tablet PCs). Service requests coming from each application were collected in a separate log file (i.e. dataset). A more thorough description of the analyzed datasets and their corresponding taxonomies follows.

Recs. The *Recs* application is a recommender system that provides recommendations to users on entertainment activities (e.g. restaurants and museums). Each user can request a recommendation, vote for an item (i.e. an entertainment center), update a vote, upload a file or a photo to provide useful information about an item (i.e. a restaurant or a museum), and post a comment. Hence, a set of services is provided to the end users to perform the

described operations/services. The dataset contains the user requests that were submitted and that were obtained by logging the user requests over the time period of three months. For *Recs*, the following aggregation trees have been considered:

- date → month → trimester → year
- time stamp → hour → time slot (2-hour time slots)→ day period (AM/PM)
- user → gender
- service → service category

TeamLife. The *TeamLife* dataset was generated by logging the *TeamLife* application requests. *TeamLife* users can upload files, photos, and videos, share them with other system users, and post short messages. The uploading services (i.e. file, photo, and video uploading services) are aggregated into the UploadData service category. The dataset collects the user requests that were submitted over a time period of three months. For *TeamLife* we used a taxonomy that is similar to the one previously described for the *Recs* dataset.

UCI benchmark datasets

To analyze the MGI Miner algorithm performance we exploited a set of UCI benchmark datasets [8] with different characteristics in terms of number of records and attributes. The main dataset characteristics are summarized in Table 3.

The taxonomies built over the UCI datasets were generated as follows. To build the aggregation trees over the continuous data attributes, we applied

several equi-depth discretization steps with finer granularities [31]. Specifically, the finest discretized values were considered to be the data item values and thus became the taxonomy leaf nodes, while the coarser discretizations were exploited to aggregate the corresponding low-level values into higher level values. For the UCI datasets reported in Table 3 we created a 3-level taxonomy by applying a 10-bin equal frequency discretization to generate the level-1 items and a 5-bin discretization to generate the level-2 items. At the top of the hierarchy, the level-2 items were aggregated into the root node. In contrast, the aggregation trees built over the nominal data attributes were analyst-provided. The items for which no meaningful aggregation is available were aggregated directly into the root node.

A description of a representative UCI dataset coming from the census domain and its corresponding taxonomy follows.

Adult dataset. Adult collects census data about American people (e.g. education, occupation, marital status, race, and sex). We defined the following aggregation trees for the nominal attributes Education, Marital-status, and Native-country.

- Education (Preschool, 1st-4th grades, ..., 12th grade → Pre High School / HS-grad → High School / Assoc-acdm, Assoc-voc, Some College, Bachelors, Masters, Prof-school, Doctorate → Post High School)
- Marital-status (Civil married, Church married → Married / Separated, Divorced, Widowed → Unmarried)
- Native-country (England, Germany, ...→ Europe / China, Japan, Thailand, ...→ Asia / United-States, Canada, Mexico, ...→ America

Table 3: UCI and real mobile dataset characteristics and number of mined MGIs with max_neg_cor=0.6 and min_pos_cor=0.7.

	Dataset	**Rec.**	**Attr.**	**Number of items with level=1**	**Number of items with level>1**	**min_sup**	**Gen. Itemsets (level>1)**	**max_NOD**	MGIs
UCI	**Adult**	32,561	15	166	135	1%	353,622	1%	26
								5%	33
	Breast	699	11	742	45	1%	11,454	1%	9
								5%	36
	Cleve	303	14	110	20	1%	240,941	1%	1
								5%	1
	Crx	690	16	98	27	1%	1,457,397	1%	32
								5%	36
	Glass	214	11	306	44	1%	24,872	1%	25
								5%	25
	Heart	270	14	73	25	1%	630,495	1%	1
								5%	1
	Letter recognition	20,000	17	160	80	1%	503,328	1%	6
								5%	6
	Pima	768	9	85	40	1%	10,596	1%	3
								5%	3
	Pendigits	10,992	17	160	80	1%	437,364	1%	1
								5%	2
	Shuttle	43,500	10	89	42	1%	6,747	1%	81
								5%	108
	Vehicle	846	19	154	90	1%	4,717,399	1%	32
								5%	40
	Waveform	5,000	22	89	54	1%	13,589,519	1%	11
								5%	11
	Wine	178	14	133	65	1%	2,406,612	1%	5
								5%	5
Mobile	TeamLife	1,197	4	1,293	31	1%	225	10%	1
								15%	2
	Recs	5,668	4	3,979	39	0.15%	475	10%	1
								15%	1

/ South Africa, $\ldots \rightarrow$ Africa)

For the remaining nominal attributes no item aggregations (disregarding the root node) have been defined.

Synthetic datasets

We used the function 2 of the Quest IBM synthetic dataset generator [18], which was first exploited in [21] in the context of data classification, to generate synthetic data. The data generator automatically produces structured

datasets that are composed of a user-specified number of records and attributes. To automate the taxonomy generation procedure we extended the data generator source code as follows. Once a user has specified the required taxonomy height H, for each attribute the item values are treated as taxonomy leaf nodes, sorted into lexicographical order, and clustered into a subset of equal-frequency bins. Each bin is associated with a generalized item that aggregates all group members. Next, the high-level bins are further aggregated to each other and the procedure iterates until all the items are clustered in a unique node (i.e. the root node). At each generalization level the bin frequency is automatically derived from the taxonomy height and the attribute domain cardinality. For example, setting H to 3 a 27-value attribute domain is partitioned into 9 equal-frequency bins at level 1, 3 equal-frequency bins at level 2 and a unique bin at level 3. The extended generator code is available at [11].

5.2. Expert-driven MGI *validation in a mobile application scenario*

We evaluated the usefulness of the MGIs mined from the real-life data taken from a mobile scenario with the help of a domain expert. Table 4 reports two MGIs that were extracted from the TeamLife dataset by enforcing min_sup=1%, max_neg_cor=0.6, min_pos_cor=0.7, and max_NOD=15%. As an example, in this section we discuss their usability for supporting experts in planning marketing campaigns and resource allocation.

Each of the MGIs reported in Table 4 consists of a positively correlated level-2 generalized itemset X and a set $\mathcal{E}$ of negatively correlated low-level (descendant) itemsets. Let us consider the MGI 1 first. The traditional high-level itemset X={(User, Male), (Service, UploadData)} indicates that

Table 4: Examples of MGIs mined from TeamLife.

ID	MGI	**support (%)**	**NOD (%)**	X**'s correlation type (Kulc value)**
MGI 1	{(User, Male), (Service, UploadData)} ▷ { {(User,UserA), (Service,Photo)}, {(User,UserB), (Service,Photo)}, ... {(User,UserZ), (Service,File)}, {(User,UserZ), (Service,Photo)} }	70.0%	7.63%	positive (0.86)
MGI 2	{(Date, May), (Service, UploadData)} ▷ { {(Date,2009-05-01), (Service,Photo)}, {(Date,2009-05-06), (Service,File)}, ... {(Date,2009-05-29), (Service,Photo)}, {(Date,2009-05-31), (Service,Photo)} }	58.3%	12.9%	positive (0.76)

the UploadData mobile services (i.e. Photo, File, and Video) are frequently requested by male users. The domain expert could exploit such information for marketing purposes. For instance, he may recommend to male users services that belong to the UploadData category, while disregarding the specific type of service each user is actually interested in. However, analyzing MGI 1 the above pattern turns out to be misleading. In fact many of X's descendants (i.e. many combinations of a user with a specific UploadData service) show an opposite trend. Specifically, many male users appear to be negatively correlated with at least one of the services that belong to the UploadData category. Hence, recommending to male users all the UploadData services indiscriminately could be a suboptimal choice for marketing purposes. On the contrary, the expert should consider the correlation type of each descendant itemset separately in order to perform targeted recommendations. Note that the NOD value of the MGI 1 is 7.63%. Hence, the service requests that are covered by itemsets that represent contrasting correlations are approximately 92% of those that are covered by the (misleading) high-level itemset. Therefore, discovering MGIs rather than traditional itemsets

allows analysts to avoid planning non-personalized and possibly ineffective marketing campaigns. On the other hand, the domain expert may further investigate the interest of some specific users that show a contrasting correlation with one or more services of the UploadData category in order to offer them personalized promotions.

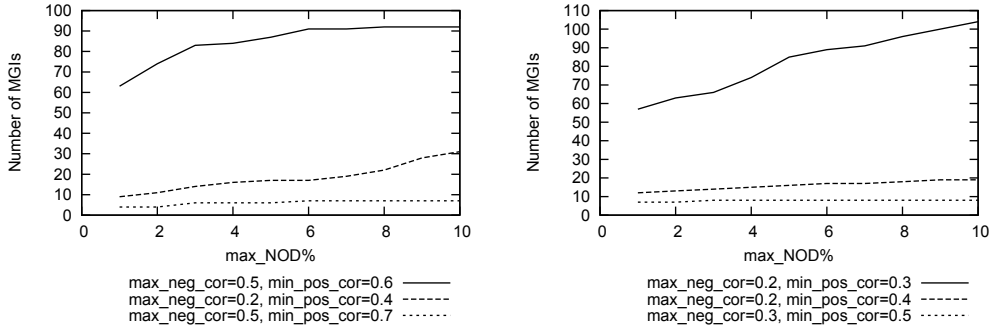
The domain expert deemed the MGI 2 to be interesting to support resource allocation/shaping. The generalized itemset X={(Date, May), (Service, UploadData)} indicates a positive correlation between the UploadData category and a specific month. Experts could exploit such knowledge to allocate dedicated resources to the UploadData service category in May, while disregarding the individual user's interests. However, analyzing MGI 2 may prompt the expert to perform a more conservative and accurate resource allocation and shaping. More specifically, it turns out that some UploadData services are requested less than expected during the early days of May. Since the subset of negatively correlated frequent descendants covers approximately 87% of the records that are covered by {(Date, May), (Service, UploadData)}, the high-level ancestor could be considered to be a misleading high-level pattern. Rather than allocating the resources for all the UploadData services indiscriminately, the network resource manager should perform a more selective resource allocation according to the actual daily service usage. For example, since the Photo service appears to be, on average, underused during most of the days of May, the manager should allocate a portion of its currently dedicated bandwidth to any other service of the same category (e.g., File or Video).

5.3. Algorithm parameter analysis

We analyzed the impact of the main MGI Miner algorithm parameters on the number of MGIs mined from the UCI datasets. In the following section, the effect of each algorithm parameter will be discussed separately.

5.3.1. Effect of the maximum NOD threshold

The maximum NOD threshold allows experts to select only the MGIs that represent an unexpected and clearly misleading pattern. It indicates the maximum portion of data that are covered by a generalized itemset and that are not covered by any of its low-level contrasting correlations.



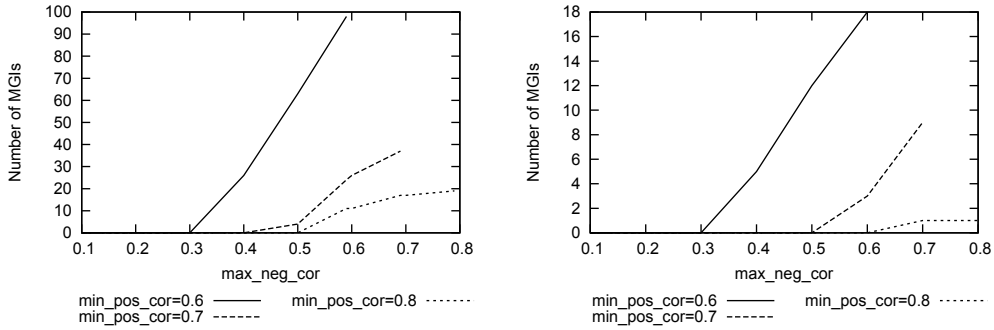
(a) Adult (b) Pima

Figure 2: Impact of the maximum NOD threshold on the number of mined MGIs. min_sup=1%.

Figures 3(a) and 3(b) plot the number of MGIs mined by varying the max_NOD value in the range [0, 10%] on two representative UCI datasets, i.e. Adult and Pima, respectively. As expected, lowering the maximum NOD threshold the number of extracted MGIs decreases more than linearly because of the higher selectivity of the enforced constraint. Note that even setting a relatively high max_NOD threshold value (e.g. 10%) the number of

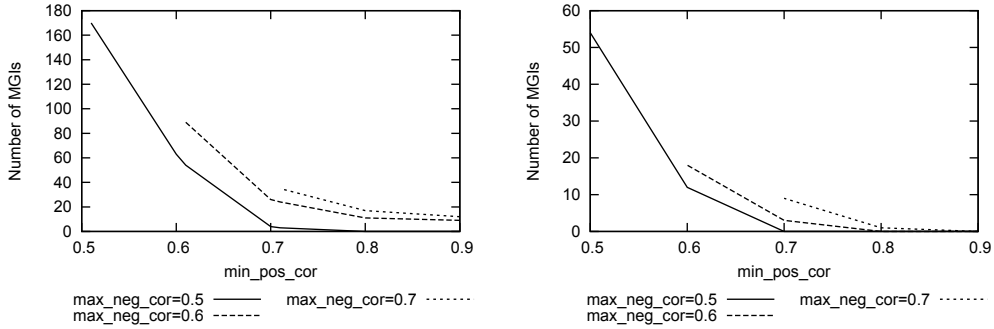
extracted MGIs remains limited (e.g. around 130 for the Adult dataset).

Similar results were obtained for the mobile dataset and the other UCI datasets. To provide an insight into the achieved results, Table 3 summarizes the results that were achieved by setting two representative max_NOD values.



(a) Adult (b) Pima

Figure 3: Impact of the maximum negative threshold max_neg_cor on the number of mined MGIs. max_NOD=1%, min_sup=1%.



(a) Adult (b) Pima

Figure 4: Impact of the minimum positive threshold min_pos_cor on the number of mined MGIs. max_NOD=1%, min_sup=1%.

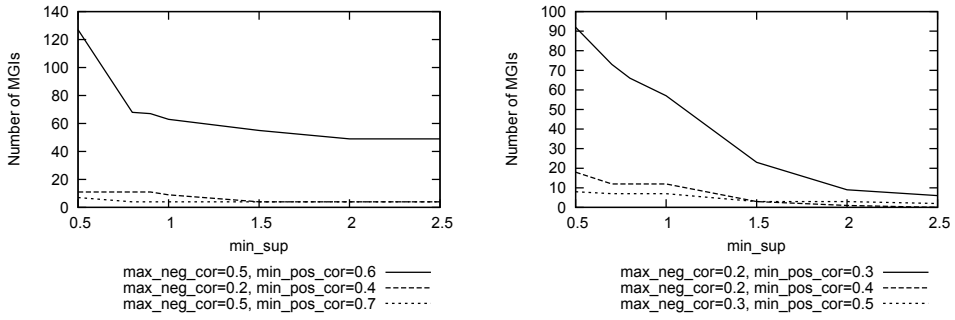
5.3.2. Effect of the correlation thresholds

Enforcing different maximum negative and minimum positive correlation thresholds can affect MGI Miner algorithm performance and the characteristics of the mined patterns. To analyze the impact of the positive and negative correlation thresholds separately, in Figures 3(a) and 3(b) we plotted the number of MGIs mined by varying max_neg_cor and by setting three representative min_pos_cor values on Adult and Pima, while in Figures 4(a) and 4(b) we analyzed the opposite situation, i.e. we varied min_pos_cor by setting three representative values for max_neg_cor on the same datasets. Note that since max_neg_cor $<$ min_pos_cor some curve points are missing.

As expected, the itemset correlation changes occur more frequently while setting closer max_neg_cor and min_pos_cor values. Moreover, the itemset correlation values appear to be unevenly distributed among the analyzed data. Specifically, the majority of the frequent generalized itemsets have a Kulc value between 0.4 and 0.7. Hence, setting max_neg_cor and min_pos_cor in such a value range yields a significant increase in the number of extracted MGIs, because the generalization process is likely to change the correlation type. On the other hand, setting the positive and the negative thresholds out of the above-mentioned value range yields a mined set cardinality reduction.

5.3.3. Effect of the minimum support threshold

The minimum support threshold min_sup significantly affects the characteristics of the results of the traditional itemset mining algorithms (e.g. Apriori [2], FP-Growth [15]). For this reason, we also analyzed the impact of min_sup on the characteristics of the mined patterns. Figures 5(a) and 5(b) report the number of MGIs extracted from Adult and Pima by varying



(a) Adult (b) Pima

Figure 5: Impact of the minimum support threshold min_sup on the number of mined MGIs. max_NOD=1%.

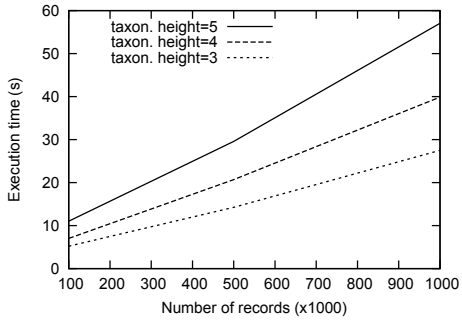
min_sup in the range [1%,10%] and by setting three representative pairs of min_pos_cor and max_neg_cor values.

The number of mined MGIs increases while lower min_sup values are enforced. This trend is mainly due to the combinatorial increase in the number of generated item combinations which yields a super-linear increase in the number of frequent traditional generalized itemsets. Although the curve slopes depend on the analyzed data distribution and the enforced correlation thresholds (see Section 5.3.2), the results that were achieved on datasets with different characteristics show rather similar trends.

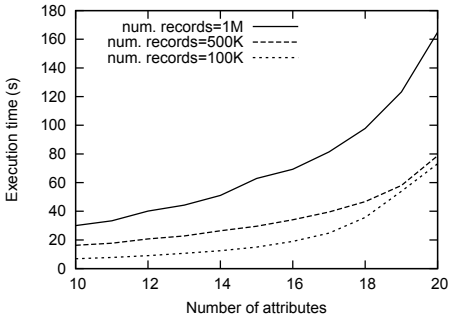
5.4. Scalability

We analyzed the MGI Miner algorithm scalability, in terms of execution time, on synthetic data with (i) the number of records, (ii) the number of attributes, and (iii) the taxonomy height.

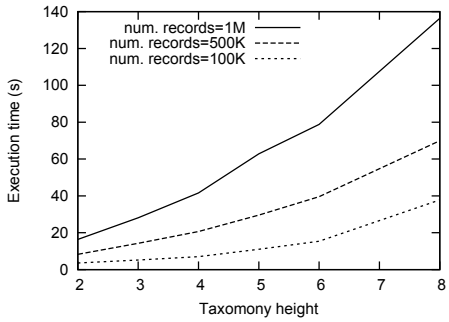
To evaluate the scalability with the number of records we varied the data cardinality in the range [10^5, 10^6] while setting the number of attributes to 15 and three representative taxonomy height values (i.e, 3, 4, and 5).



(a) Scalability with the number of records. Num. of attributes=15.



(b) Scalability with respect to the number of attributes. Taxonomy height=5.



(c) Scalability with respect to the taxonomy height. Num. of attributes=15.

Figure 6: MGI Miner scalability. min_sup=1%, max_NOD=1%, max_neg_cor=0.6, min_pos_cor=0.7.

The results, reported in Figure 6(a), show that the MGI Miner execution time scales roughly linearly with the number of records, because the data distribution remains approximately unchanged while increasing the dataset cardinality.

We also analyzed the impact of the number of attributes and the taxonomy height on the MGI Miner execution time. In the former case, we varied the dataset dimensionality in the range [10, 20], while considering a 5-level taxonomy and three representative dataset cardinalities (i.e. 10^5, 5×10^5, 10^6). In the latter, we varied the taxonomy height between 2 and 8, while considering three 15-attribute datasets with different size. The results, reported in Figures 6(b) and 6(c), show that the MGI Miner execution time scales more than linearly with both the number of attributes and the taxonomy height because of the combinatorial increase in the number of generated combinations. However, the execution time remains acceptable even when coping with rather complex datasets and taxonomies (e.g. approximately 140s for a 15-attribute dataset with 10^6 records and an 8-level taxonomy).

6. Related work

The generalized itemset and association rule mining problem was first introduced in [26] in the context of market basket analysis. The authors proposed an Apriori-based algorithm [3] to discover frequent itemsets and association rules at different abstraction levels from datasets that were supplied with taxonomies. However, since the mining process evaluates the input taxonomy exhaustively a large number of (possibly redundant) item combinations is generated. A step beyond towards the generation of a more compact

and humanly manageable pattern set has been made in [4, 19, 27, 28]. The proposed approaches enforce mining constraints to discover a worthwhile subset of frequent generalized itemsets or association rules. For example, in [4] the authors propose to push boolean constraints, which enforce the presence or the absence of an arbitrary item combination, into the mining process. In [27] the authors also take subset-superset and parent-child taxonomic relationships into account to avoid generating all the item combinations. More recently, an important research effort has also been devoted to discovering closed and maximal generalized itemsets [19, 28], which represent notable itemset subsets [23]. The authors in [6] propose to select only the frequent generalized itemsets that have at least one infrequent descendant to also consider rare but potentially interesting knowledge. Unlike [4, 19, 26, 27, 28], our approach does not focus on itemset pruning but rather it addresses the complementary issue of highlighting misleading high-level itemsets, which are represented, to a large extent, by their low-level contrasting correlations.

A significant effort has also been devoted to discovering frequent item correlations among large datasets [1, 7, 9, 25]. In this context, a pioneering work [9] proposes to evaluate association rule significance via the chi square test for correlation. The authors also exploit the upward closure of the chi square measure to discard some uninteresting candidate itemsets early. To extract negatively correlated item correlations, which are usually characterized by low support value [30], in [1, 25, 29] two novel itemset correlation measures, namely *collective strength* and *support expectation*, have also been proposed and used to perform indirect negative association rule mining. To evaluate item correlation independently of the dataset size in [34] a null-

invariant Kulczynsky measure has also been proposed [16]. In [7] the same measure has been exploited to discover flipping correlations among data that were supplied with taxonomies. Flipping correlations are itemsets for which the correlation type flips from positive to negative (or vice versa) when items are generalized to a higher level of abstraction for *every* generalization step. However, when coping with real-life data, item correlation flippings are not likely to occur at every generalization step. Furthermore, a generalized itemset may have many low-level contrasting correlations which are worth considering all together. Unlike [7], this paper addresses the complementary issue of discovering a worthwhile subset of misleading high-level itemsets which are covered, to a large extent, by contrasting correlations at lower abstraction levels.

Parallel research efforts have also been devoted to proposing optimization strategies to efficiently address generalized itemset mining [14, 17, 24, 33]. While the authors in [14] propose an Apriori-based top-down traversal of the search space, an FP-Growth-like approach to generalized itemset mining [24] and a mining algorithm [17] that exploits the vertical data format [35] have also been presented. In contrast, in [33] an efficient data structure is used to store and generalize low-level itemsets and association rules. Furthermore, the discovery of a succinct and non-redundant subset of frequent itemsets [5, 10, 20, 32] has also been investigated. Since the above approaches do not address misleading generalized itemset mining, their goal is somehow related to but different from those addressed by this work.

7. Conclusions and future research directions

This paper proposes to discover a novel generalized itemset type, called Misleading Generalized Itemsets (MGIs), from structured datasets that are supplied with taxonomies. MGIs represent misleading frequent high-level data correlations that are worth analyzing apart from traditional itemsets. Each MGI represents a frequent generalized itemset X and its subset of low-level frequent descendants for which the correlation type is in contrast to those of X. An MGI interestingness measure, named Not Overlapping Degree (NOD), is also proposed to select only the (misleading) high-level itemsets that represent almost the same portion of data that is covered by their low-level contrasting correlations. Furthermore, an algorithm to mine MGIs at the top of the traditional itemsets has also been proposed. The experimental results demonstrate the usefulness of the proposed approach for discovering interesting misleading itemsets from real mobile datasets.

We plan to extend our research work in the following directions: (i) The study of the applicability of the proposed approach to other real-life contexts (e.g. social network analysis [13], medical data analysis [22]), (ii) the use of different correlation measures (e.g. coherence [34]), and (iii) the pushing of more complex mining constraints into the MGI extraction process.

8. Acknowledgements

We are grateful to Telecom Italia Lab for providing us with the two mobile datasets.

[1] Aggarwal, C. C., & Yu, P. S. (1998). A new framework for itemset generation. In *Proceedings of the seventeenth ACM SIGACT-SIGMOD-*

SIGART symposium on Principles of database systems PODS '98 (pp. 18–24).

[2] Agrawal, R., Imielinski, T., & Swami (1993). Mining association rules between sets of items in large databases. In *ACM SIGMOD 1993* (pp. 207–216).

[3] Agrawal, R., & Srikant, R. (1994). Fast algorithms for mining association rules in large databases. In *Proceedings of the 20th VLDB conference* (pp. 487–499).

[4] Agrawal, R., & Srikant, R. (1997). Mining association rules with item constraints. In *KDD 1997* (pp. 67–73).

[5] Amphawan, K., Lenca, P., & Surarerks, A. (2012). Mining top-k regular-frequent itemsets using database partitioning and support estimation. *Expert Systems with Applications*, *39*, 1924 – 1936. URL: `http://www.sciencedirect.com/science/article/pii/S0957417411011663`. doi:10.1016/j.eswa.2011.08.055.

[6] Baralis, E., Cagliero, L., Cerquitelli, T., D'Elia, V., & Garza, P. (2010). Support driven opportunistic aggregation for generalized itemset extraction. In *5th IEEE International Conference on Intelligent Systems* (pp. 102–107).

[7] Barsky, M., Kim, S., Weninger, T., & Han, J. (2011). Mining flipping correlations from large datasets with taxonomies. *Proc. VLDB Endow.*, *5*, 370–381.

[8] Blake, C., & Merz, C. (2012). Uci repository of machine learning databases. Available at http://archive.ics.uci.edu/ml. Last accessed: 20/07/2012.

[9] Brin, S., Motwani, R., & Silverstein, C. (1997). Beyond market baskets: generalizing association rules to correlations. *SIGMOD Rec.*, *26*, 265–276.

[10] Calders, T., & Goethals, B. (2002). Mining all non-derivable frequent itemsets. In *Proceedings of the 6th European Conference on Principles of Data Mining and Knowledge Discovery* PKDD '02 (pp. 74–85).

[11] DBDMG (2013). Database and data mining group website: http://dbdmg.polito.it/wordpress/research/misleading-generalized-itemsets/. last accessed: 20/06/2013. URL: `http://dbdmg.polito.it/wordpress/research/misleading-generalized-itemsets/`.

[12] Gharib, T. F. (2009). An efficient algorithm for mining frequent maximal and closed itemsets. *Int. J. Hybrid Intell. Syst.*, *6*, 147–153.

[13] Guo, L., Tan, E., Chen, S., Zhang, X., & Zhao, Y. (2009). Analyzing patterns of user content generation in online social networks. In *Proceedings of the 15th ACM SIGKDD international conference on Knowledge discovery and data mining* (pp. 369–378).

[14] Han, J., & Fu, Y. (1999). Mining multiple-level association rules in large databases. *IEEE Transactions on knowledge and data engineering*, *11*, 798–805.

[15] Han, J., Pei, J., & Yin, Y. (2000). Mining frequent patterns without candidate generation. In *SIGMOD'00* (pp. 1–12).

[16] Hilderman, R. J., & Hamilton, H. J. (2001). *Knowledge Discovery and Measures of Interest*. Norwell, MA, USA: Kluwer Academic Publishers.

[17] Hipp, J., Myka, A., Wirth, R., & Guntzer, U. (1998). A new algorithm for faster mining of generalized association rules. *Proceedings of the 2nd European Symposium on Principles of Data Mining and Knowledge Discovery (PKDD98)*, (pp. 74–82).

[18] IBM (2009). IBM Quest Synthetic Data Generation Code. URL: `http://www.almaden.ibm.com/`.

[19] Kunkle, D., Zhang, D., & Cooperman, G. (2008). Mining frequent generalized itemsets and generalized association rules without redundancy. *J. Comput. Sci. Technol.*, *23*, 77–102.

[20] Mampaey, M., Tatti, N., & Vreeken, J. (2011). Tell me what I need to know: succinctly summarizing data with itemsets. In *Proceedings of the 17th ACM SIGKDD international conference on Knowledge discovery and data mining* KDD '11 (pp. 573–581).

[21] Mehta, M., Agrawal, R., & Rissanen, J. (1996). Sliq: A fast scalable classifier for data mining. In *EDBT* (pp. 18–32).

[22] Nahar, J., Imam, T., Tickle, K. S., & Chen, Y.-P. P. (2013). Association rule mining to detect factors which contribute to heart disease in males and females. *Expert Systems with Applications*, *40*, 1086 – 1093. URL:

http://www.sciencedirect.com/science/article/pii/S095741741200989X. doi:10.1016/j.eswa.2012.08.028.

[23] Pasquier, N., Bastide, Y., Taouil, R., & Lakhal, L. (1999). Discovering frequent closed itemsets for association rules. In *Proceedings of the 7th International Conference on Database Theory* ICDT '99 (pp. 398–416).

[24] Pramudiono, I., & Kitsuregawa, M. (2004). Fp-tax: tree structure based generalized association rule mining. In *DMKD '04* (pp. 60–63).

[25] Savasere, A., Omiecinski, E., & Navathe, S. B. (1998). Mining for strong negative associations in a large database of customer transactions. In *Proceedings of the Fourteenth International Conference on Data Engineering* ICDE '98 (pp. 494–502).

[26] Srikant, R., & Agrawal, R. (1995). Mining generalized association rules. In *VLDB 1995* (pp. 407–419).

[27] Sriphaew, K., & Theeramunkong, T. (2002). A new method for finding generalized frequent itemsets in association rule mining. In *Proceeding of the VII International Symposium on Computers and Communications* (pp. 1040–1045).

[28] Sriphaew, K., & Theeramunkong, T. (2004). Fast algorithms for mining generalized frequent patterns of generalized association rules. *IEICE Transactions on Information and Systems*, *87*, 761–770.

[29] Tan, P.-N., Kumar, V., & Srivastava, J. (2000). Indirect association: Mining higher order dependencies in data. In *Proceedings of the 4th*

European Conference on Principles of Data Mining and Knowledge Discovery PKDD '00 (pp. 632–637).

[30] Tan, P.-N., Kumar, V., & Srivastava, J. (2002). Selecting the right interestingness measure for association patterns. In *ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD'02)* (pp. 32–41).

[31] Tan, P.-N., Steinbach, M., & Kumar, V. (2005). *Introduction to Data Mining*. Addison-Wesley.

[32] Tatti, N. (2010). Probably the best itemsets. In *Proceedings of the 16th ACM international conference on Knowledge discovery and data mining* (pp. 293–302).

[33] Wu, C.-M., & Huang, Y.-F. (2011). Generalized association rule mining using an efficient data structure. *Expert Systems with Applications*, *38*, 7277 – 7290. URL: http://www.sciencedirect.com/science/article/pii/S0957417410013862. doi:10.1016/j.eswa.2010.12.023.

[34] Wu, T., Chen, Y., & Han, J. (2010). Re-examination of interestingness measures in pattern mining: a unified framework. *Data Min. Knowl. Discov.*, *21*, 371–397.

[35] Zaki, M., Parthasarathy, S., Ogihara, M., & Li, W. (1997). New algorithms for fast discovery of association rules. *3rd Intl. Conf. on Knowledge Discovery and Data Mining*, (pp. 283–286).