Towards a Framework for Semantic Exploration of Frequent Patterns
Behrooz Omidvar Tehrani, Sihem Amer-Yahia, Alexandre Termier, Aurélie Bertaux, Éric Gaussier, Marie-Christine Rousset

To cite this version:
Behrooz Omidvar Tehrani, Sihem Amer-Yahia, Alexandre Termier, Aurélie Bertaux, Éric Gaussier, et al.. Towards a Framework for Semantic Exploration of Frequent Patterns. IMMoA 2013 - International Workshop on Information Management in Mobile Application (in conjunction with VLDB 2013), Aug 2013, Riva del Garda, Trento, Italy. pp.7-14. hal-00881195

HAL Id: hal-00881195
https://hal.archives-ouvertes.fr/hal-00881195
Submitted on 8 Nov 2013

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L’archive ouverte pluridisciplinaire HAL, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d’enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.
Towards a Framework for Semantic Exploration of Frequent Patterns

Behrooz Omidvar Tehrani†, Sihem Amer-Yahia††, Alexandre Termier‡, Aurélie Bertaux‡‡, Eric Gaussier‡, Marie-Christine Rousset§
†LIG; ‡CNRS; ‡‡INRIA France
†firstname.lastname@imag.fr
‡‡aurelie.bertaux@inria.fr

ABSTRACT
Mining frequent patterns is an essential task in discovering hidden correlations in datasets. Although frequent patterns unveil valuable information, there are some challenges which limit their usability. First, the number of possible patterns is often very large which hinders their effective exploration. Second, patterns with many items are hard to read and the analyst may be unable to understand their meaning. In addition, the only available information about patterns is their support, a very coarse piece of information. In this paper, we are particularly interested in mining datasets that reflect usage patterns of users moving in space and time and for whom demographics attributes are available (age, occupation, etc). Such characteristics are typical of data collected from smart phones, whose analysis has critical business applications nowadays. We propose pattern exploration primitives, abstraction and refinement, that use hand-crafted taxonomies on time, space and user demographics. We show on two real datasets, Nokia and MovieLens, how the use of such taxonomies reduces the size of the pattern space and how demographics enable their semantic exploration. This work opens new perspectives in the semantic exploration of frequent patterns that reflect the behavior of different user communities.

1. INTRODUCTION
Nowadays, large amounts of user-generated content representing behavioral data are made available. This is particularly true for data generated by users carrying a mobile phone and moving in different geographic regions. The large size of such data hinders its effective exploration. Fortunately, user-generated data contains repetitive behavior that can be discovered using frequent pattern mining, a common method for discovering hidden patterns that capture some regularities or correlations in the data. Those patterns are used in decision support and can be displayed to analysts for further exploration. There has been extensive work on optimizing algorithms for mining patterns from large datasets. The problem is that there can be millions of automatically discovered patterns, hindering their analysis. Moreover, such patterns can be very long (i.e., containing many items), making them difficult to interpret.

Our goal is to develop a method that leverages the richness of underlying data to determine which patterns the analyst has to focus his attention on, and how to interpret those patterns. The two questions we ask ourselves are: i. How to explore frequent patterns that characterize subparts of a pattern space in a data-centric way, by giving legible and useful information to the analyst? The possibility to organize items forming a pattern along space and time taxonomies is a new opportunity to reduce the size of the pattern space and the length of patterns. ii. How to help an analyst better interpret a pattern by going beyond the notion of support? The availability of demographics information such as users’ age and occupation enables a semantic exploration of the space of support users of a pattern.

The most simple way to explore patterns is skimming through the list of frequent patterns, i.e., those for which there is enough evidence in the underlying dataset, or in other terms, those whose support (number of users who exhibit the behavior illustrated in the pattern), is above a given threshold. However, just like in Web search, a list that is more than few dozens of patterns long is infeasible to exploit effectively. Instead, we define two pattern exploration primitives each of which operates on a single pattern at a time. When applied to a pattern, abstraction reduces its size and as a side-effect, the size of the pattern space. Refinement, on the other hand, highlights different subsets of users forming the support of a pattern, making it more understandable to the analyst.

1.1 Abstraction and Refinement Examples
We propose to use two very different datasets, each of which contains a rich set of usage data and user demographics. Our focus is on Nokia, a small dataset (38 users) that contains application usage data on smartphones and that was made available to the research community as part of a challenge. Nokia contains one year of smartphone usage traces, from GPS position to applications used with a millisecond resolution. We also validate our approach on
2. FRAMEWORK

2.1 Data Model: Taxonomies

We are given a set of users \( U \), items \( I \), locations \( L \) and a database \( D \) of quadruples of the form \( < u, i, l, t > \) where \( u \in U \), \( i \in I \), and \( l \in L \) and \( t \), a time-stamp, represent the location and time user \( u \) has used (opened, watched, rated, voted, etc.) item \( i \).

Each user \( u \) is also described with attributes drawn from a set of attributes \( A \) representing demographics information such as \( \text{Gender} \) and \( \text{Age} \). We refer to each attribute in \( A \) as \( a_i \) and to its values as \( v_i \). The domain of values of attribute \( a_i \) is \( D_{a_i} \) with \( D_A = \cup D_{a_i} \). For example, if we use \( a_1 \) to refer to \( \text{Gender} \), it takes two values \( v_1 \) and \( v_2 \) representing \text{male} and \text{female} respectively.

Figure 1: \( \tau_A \): User Attribute Taxonomy

User attributes, items, location and time, are organized in hand-crafted taxonomies. The values of each user attribute in \( A \) are organized in a taxonomy \( \tau_A \) (Figure 1). Similarly, items in \( I \) (applications in NOKIA and movies in MOVIELENS) and locations in \( L \) are organized into their respective taxonomies \( \tau_I \) and \( \tau_L \). The set of all taxonomies is referred to as \( \tau \). We do not aim to show all the taxonomies we built for our datasets, rather we illustrate some examples that will be used later in the paper. In particular, the time taxonomy is omitted. Figure 2 shows a subset of the taxonomy we built for NOKIA applications. Figure 3 shows a subset of the location taxonomy for MOVIELENS. Finally, Figure 4 shows the taxonomy for MOVIELENS movies.

Figure 2: \( \tau_I \): Application Taxonomy

The intuition behind refinement is to enable the understanding of users that constitute the support of a pattern. The pattern \{ The Fugitive, Terminator 2, Men in Black, The Matrix \} in MOVIELENS has a support of 1054 users. Refinement reveals more information on those users by providing a mechanism for exploring their demographics. To enable that, we need a mechanism that identifies which subsets of users in the support of a pattern we need to focus on. In our example, if for instance most support users of the pattern are [28-33] years old, this age bracket would qualify as a refinement for that pattern. To enable that, refinement is a primitive that relies on a notion of \textit{salience} for user demographics in order to examine the distribution of values of different demographics attributes in a pattern.

1.2 Contributions

We formalize pattern abstraction and refinement as two pattern exploration primitives. We then study experimentally the potential of our primitives in reducing the space of patterns, making them more compact and hence more readable, and in providing a better understanding of the support users of patterns. Our work lays the ground for exploring how time, geography and item taxonomies as well as demographics attributes enable pattern exploration using behavior semantics.

The paper is organized as follows. Section 2 describes our data model, patterns and primitives. Section 3 contains an evaluation of our primitives. Related work is reviewed in Section 4. Finally, Section 5 concludes with a discussion of ongoing and planned efforts.
attribute values

user behavior and a pattern can be read as users with

itemset mining algorithm \[1\].

MovieLens

Nokia, using a closed frequent

patterns satisfying a minimum support value \(\sigma\) and denoted \([\|\cdot\|) = \|\cdot\|\). Typically, only

users is referred to as \(p\) satisfying

ber of users in

and relies on a taxonomy-based usage measure and re

powerful. Our abstraction method is not merely syntactic

to abstract items in a pattern into their parent item in the

taxonomy. The intuition behind abstraction is simple yet

portant of usage between sibling items and their parent in \(\tau_1\).

The intuition of taxonomy-based usage is that if most of

the usage of a given item is that of some of its children in

the taxonomy \(\tau_1\), those children could be replaced by their

parent in all patterns they appear in thereby reducing the

size of those patterns and making them more readable.

\begin{definition}

Taxonomy-Based Usage. Given a set of sibling items \(i_1, \ldots, i_n\) and their parent item \(i\) in the
taxonomy \(\tau_1\), their taxonomy-based usage in a pattern \(p\), de-
noted \(Pusage(p; i_1, \ldots, i_n)\), is the pro-

cportion of usage between sibling items and their parent in \(\tau_1\).

\end{definition}

Validity. Given an abstraction threshold \(\rho\) and a pattern \(p\) containing sibling
terminal \(\rho\) if \(Pusage(p; i_1, \ldots, i_n) \geq \rho\) and \(\forall i_1, i_n \notin \rho\) and \(i \in \rho\).

\begin{definition}

Maximal Pattern Abstraction. Given an abstraction threshold \(\rho\), we say that a pattern \(p_a\) is a max-
al abstraction of a pattern \(p\) iff \(Pusage(p_a; i_1, \ldots, i_n) \geq \rho\) is satisfied.

Let us now illustrate the definitions above on our datasets.

John. In this section, we show some examples of abstraction us-

taxonomies. In Nokia, pattern \(p = \{\text{Studying Full-time, Female, FG Thread, WLAN Wizard, Calculator, Calendar, Bluetooth, Contacts, Log, Web, Text message, Messaging}\}

has a support equal to 4. Given an abstraction threshold of

50%, we obtain a maximal abstraction of \(p\) using application
taxonomy into \(p_a = \{\text{Studying Full-time, Female, System, WLAN Wizard, Calculator, Calendar, Desktop Communication, Web App}\}\) where the highlighted applications are parent classes in the taxonomy.

The pie charts A, B and C in Figure 5 contain two other usages, one showing that

50% usage of System items is for FG Thread and another showing that 53.61% of usages of Web App items, is for Web.

Finally, in pie charts D and E of Figure 5, we show two examples of non-valid abstractions given a threshold equal to 50%. We see that 9% usage of Configuration items, is for WLAN Wizard and that 22% usage of Desktop Common items, is for Calculator and Calendar. Thereby, none of those could be abstracted in pattern \(p\).

As another example, pattern \(p = \{\text{Engineer, Age 18-45, Batman, Jurassic Park in NY, MA, MN, MI, OH, TN states of U.S.}\}\) becomes abstracted to \(p_a = \{\text{Engineer, Age 18-45,}

MovieLens

using closed frequent

patterns. In our context, patterns re-

tect. In order to enhance pattern readability, we pro-

(2.3 Abstraction Primitive

A pattern mining algorithm explores a very large space
(exponential in the number of items) and can return long

patterns (the lower the support threshold, the longer the

patterns). In order to enhance pattern readability, we pro-

pose to use semantic information provided in taxonomies
to abstract items in a pattern into their parent item in the

taxonomy. The intuition behind abstraction is simple yet

portant. Our abstraction method is not merely syntactic

and relies on a taxonomy-based usage measure and reflects

a way of approximating the interest of users. This approxi-
mation could be applied to items, time of day or to location.

2.3.1 Definitions

We define the usage of an item \(i\) for a set of users \(U \subseteq \mathbb{U}\),

usage\((V, i) = \{\{u, i \in D \mid u \in V\}\}\) as the number of
times users in the set \(V\) used item \(i\). The usage of an item \(i\) wrt a pattern \(p\), usage\((users(p), i)\) is the number of times

the item \(i\) has been used by users who satisfy \(p\).

\begin{definition}

Pattern Abstraction in Nokia and MovieLens

In this section, we show some examples of abstraction using
taxonomies. In Nokia, pattern \(p = \{\text{Studying Full-time, Female, FG Thread, WLAN Wizard, Calculator, Calendar, Bluetooth, Contacts, Log, Web, Text message, Messaging}\}

has a support equal to 4. Given an abstraction threshold of

50%, we obtain a maximal abstraction of \(p\) using application
taxonomy into \(p_a = \{\text{Studying Full-time, Female, System, WLAN Wizard, Calculator, Calendar, Desktop Communication, Web App}\}\) where the highlighted applications are parent classes in the taxonomy.

The pie charts A, B and C in Figure 5 contain two other usages, one showing that

50% usage of System items is for FG Thread and another showing that 53.61% of usages of Web App items, is for Web.

Finally, in pie charts D and E of Figure 5, we show two examples of non-valid abstractions given a threshold equal to 50%. We see that 9% usage of Configuration items, is for WLAN Wizard and that 22% usage of Desktop Common items, is for Calculator and Calendar. Thereby, none of those could be abstracted in pattern \(p\).

As another example, pattern \(p = \{\text{Engineer, Age 18-45, Batman, Jurassic Park in NY, MA, MN, MI, OH, TN states of U.S.}\}\) becomes abstracted to \(p_a = \{\text{Engineer, Age 18-45,}

\begin{table}[!ht]

\centering

\begin{tabular}{|c|c|c|}
\hline
Dataset & Pattern & Sup. (\%) \\
\hline
\hline
Nokia & Female, Age 39-45, Calculator, Calendar, Bluetooth, Clock, Messaging & 13 \\
\hline
MovieLens & The Fugitive, Terminator 2, Men in Black, The Matrix & 17 \\
\hline
\end{tabular}

\caption{Example Patterns}

\end{table}

2.3 Abstraction Primitive

A pattern mining algorithm explores a very large space
(exponential in the number of items) and can return long

patterns (the lower the support threshold, the longer the

patterns). In order to enhance pattern readability, we pro-

pose to use semantic information provided in taxonomies
to abstract items in a pattern into their parent item in the

taxonomy. The intuition behind abstraction is simple yet

portant. Our abstraction method is not merely syntactic

and relies on a taxonomy-based usage measure and reflects

2.2 Patterns

Given a database \(D\), we are interested in finding patterns

of the form \(p = \{v_1, \ldots, v_k, i_1, \ldots, i_m\} \subseteq \mathbb{D}\) and \(\{i_1, \ldots, i_m\} \subseteq \mathbb{I}\). In our context, patterns reflect

user behavior and a pattern \(p\) can be read as users with

attribute values \(v_1, \ldots, v_k\) used items \(i_1, \ldots, i_m\). The

number of users in \(D\) satisfying \(p\) is referred to as the support

of \(p\) and denoted \(\text{support}(p) = |\text{users}(p)|\). Typically, only

patterns satisfying a minimum support value \(\sigma\) are retained.

Table 1 contains example patterns and their support re-

trieved from Nokia and MovieLens, using a closed frequent

itemset mining algorithm \[1\].

The pie charts A of Figure 5 we can see that 87.68% of usage for item

Desktop Communication is for its children items Bluetooth, Contacts, Log, Text Message and Messaging. Pie charts B

and C of Figure 5 contain two other usages, one showing that

50% usage of System items is for FG Thread and another showing that 53.61% of usages of Web App items, is for Web.

Finally, in pie charts D and E of Figure 5, we show two examples of non-valid abstractions given a threshold equal to 50%. We see that 9% usage of Configuration items, is for WLAN Wizard and that 22% usage of Desktop Common items, is for Calculator and Calendar. Thereby, none of those could be abstracted in pattern \(p\).

As another example, pattern \(p = \{\text{Engineer, Age 18-45, Batman, Jurassic Park in NY, MA, MN, MI, OH, TN states of U.S.}\}\) becomes abstracted to \(p_a = \{\text{Engineer, Age 18-45,}

\begin{figure}[!ht]

\centering

\includegraphics[width=\textwidth]{Figure3.png}

\caption{\(\tau_L\): Location Taxonomy}

\end{figure}

\begin{figure}[!ht]

\centering

\includegraphics[width=\textwidth]{Figure4.png}

\caption{MovieLens Movie Taxonomy}

\end{figure}
Many pattern interestingness measures have been suggested in the literature as summarized in [2]. However, they were designed to pick representative patterns in a large pattern space. None of them was designed to explore users in the support of a pattern.

**Definition 4. Attribute Saliency.** The saliency of an attribute \( a \), w.r.t a pattern \( p \), \( sal(a,p) \), is a measure of interestingness of the attribute \( a \), for users in users(\( p \)).

We intentionally keep the definition of saliency general in order to explore different ones. Alternative measures are variance, entropy, or a measure that computes the ratio between distribution of values of pattern users for an attribute (say age) with that same distribution for all users in \( U \). Such a measure aims at selecting user attributes for which pattern users differ from all users.

\( sal(a,p) \) can be calculated using standard deviation or entropy measures. Standard deviation is the one we use in this paper in our examples and experiments and it measures the amount of variation or dispersion from the average, in a list of values.

Having a low standard deviation score for an attribute means its values tend to be very close to the average. Also having a high score for an attribute means its values are spread out over a large range of values.

**Definition 5. Valid Pattern Refinement.** Given a saliency threshold \( \mu \) and a pattern \( p \), we say that a pattern \( p_r \) is a valid refinement of a pattern \( p \) iff \( sal(a,p) \geq \mu \) and \( a \in p_r \) is a user attribute value.

We propose to calculate all patterns that constitute valid refinements of a given pattern \( p \) and associate the \( k \) best refinements (that is, patterns) to \( p \). Those refinements constitute alternative explorations of the support users of \( p \).

### 2.4.2 Pattern Refinement in Nokia and MovieLens

Consider again pattern \( p = \{ \text{Studying Full Time, Female, FG Thread, WLAN Wizard, Calculator, Calendar, Bluetooth, Contacts, Log, Web, Text Message, Messaging} \} \) in NOKIA, with a support equal to 4 users. We show how standard deviation can be used to explore users of this pattern.

We report in Figure 6, two attributes having an especially high saliency. On the left, Age distribution of the pattern users is shown. We can see that 3 out of 4 users are in age category [28-33]. This non-homogeneous distribution leads to high saliency for attribute Age for this pattern.

That is why standard deviation is a good measure for Age in pattern \( p \). This information indicates the existence of a super-pattern \( \{ p, \text{Age 28-33} \} \) with support 3 that may be of interest for further exploration.

### 3. Evaluation

The goal of this section is to evaluate abstraction and refinement primitives on NOKIA and MOVIELENS datasets. We propose quantitative and qualitative evaluations. We discuss some interesting results for each evaluation.
3.1 Datasets

Nokia consists of data from smartphones of some participants in the course of more than one year. For each user, all records of phone events and sensors like application usage, calendar, contacts, and call-logs are logged with a time-stamp. Personal information is anonymized in the data. In our study, we focused on application usage: the opening of applications by users indicating what they use their smartphones for, at any time of day. This dataset also includes responses to a questionnaire by some users in the experiment. Demographic attributes like gender, age group, and profession come from that questionnaire. Application usage records consist of an application ID and a time-stamp of when it was used. After removing some core system applications, we ended up with 170 applications.

MOVIELENS is the dataset published by the GroupLens research group1. We used the 1M ratings version that contains 1,000,209 anonymous ratings of 3,952 movies by 6,040 users. Rating records consist of a user ID (between 1 and 6040), movie ID (between 1 and 3052), a rating (based on a 5-star scale) and time. Each user provided at least 20 ratings. When user X has rated movie Y, it means X has watched Y. This is how we define the usage or consumption of a movie by a user. For each user, gender, age group, occupation and zip-code are provided. All demographic information is provided voluntarily by users.

We pre-processed the datasets and ran the LCM closed frequent itemset mining algorithm [1] with a minimum support threshold of 7% that resulted in 74723 patterns for Nokia and 50,299,230 patterns for MOVIELENS.

3.2 Abstraction Evaluation

In order to evaluate the benefit of abstraction, we propose to explore abstraction volume and pattern space reduction as described below.

3.2.1 Abstraction Volume

As seen in Definition 2, the abstraction primitive only abstracts group of items of a pattern if a condition is met. We want to evaluate how often this condition is met, depending on the abstraction threshold \( \rho \) chosen. We thus define an abstraction volume measure, which evaluates for each pattern the ratio between the number of abstractions performed (given \( \rho > 0 \)) and the maximal number of abstractions possible (case of \( \rho = 0 \)).

Given \( N \) the number of occurred abstractions in the pattern \( p \) and \( M \) the total number of classes of the taxonomy that have at least one of their child items in \( p \), the abstraction volume of \( p \) denoted by \( \theta \) is equal to \( (N / M) * 100 \). We perform abstraction volume experiment on patterns from Nokia mined with minimum support threshold of 7%. Patterns may include demographic information and applications. We applied the abstraction method using different abstraction thresholds \( \rho \) varying from 0 % to 100 %. Figure 7 shows the result of this experiment. The evolution of abstraction volume can be categorized into three different periods by two cutting points \( M_1 = 15\% \) and \( M_2 = 60\% \).

Before \( M_1 \) (where the abstraction threshold \( \rho \) is between zero and 15 %), we observe a very mild slope in the diagram and the abstraction volume decreases only 10 %. It shows that in Nokia, low values of \( \rho \) lead to many abstractions.

Choosing \( \rho \) in this range causes to have many abstractions with less attention to the usage of attributes. It will abstract nearly syntactically except for some extreme cases where usage is very low. Therefore, it is useful to select an abstraction threshold in this range only when data does not provide many usage information.

After \( M_2 \) (where \( \rho \) is higher than 60 %), we observe that the abstraction volume decreases drastically and it remains very close to zero. It means that in this range, the number of abstractions done is very low. Thus, choosing the abstraction threshold in this range is useless for simplifying the analysis.

Between \( M_1 \) and \( M_2 \) (where \( \rho \) is between 15 to 60 %), we observe that the plot has a derivative close to -1. Thus changing \( \rho \) in these values gives a predictable reduction in the number of abstractions performed. We thus consider that this range of \( \rho \) threshold values is the most interesting, and we will focus on it for most of the following experiments.

3.2.2 Pattern Space Reduction

Applying abstraction on distinct patterns will sometimes result in the same abstracted pattern. Hence, in addition to reducing pattern size, a beneficial side effect of the abstraction primitive is to reduce the size of the pattern space. We want to evaluate the scale of this pattern space reduction experimentally.

Given a support threshold \( \sigma \) and an abstraction threshold \( \rho \), the pattern space reduction is equal to \( 1 - \frac{|P_a|}{|P|} \), where \( P_a \) is the set of all abstracted patterns and \( P \) is the set of initial (not abstracted) patterns. For Nokia, we generated the set of maximal abstracted patterns using 4 different values for \( \sigma \) i.e. 10, 25, 50 and 75% by varying \( \rho \) from 0% (i.e. syntactic abstraction) to 100% (i.e. no abstraction). The result is shown in Figure 8.

As an example, using abstraction with \( \sigma = 25\% \) and \( \rho = 20\% \), the pattern space reduces to half of its initial size. The three periods mentioned in Figure 7 with the cutting points \( M_1 = 15\% \) and \( M_2 = 60\% \) are also visible in Figure 8, with a pattern space reduction between 20 and 30% in the most interesting range \([M_1, M_2]\). When fixing the abstraction threshold \( \rho \), the lower the support threshold, the higher the reduction of the pattern space. However, for low support values the gain in reduction is from lowering the support threshold. This can be explained by the

1http://www.grouplens.org/
fact that with lower support thresholds, longer patterns are produced, many of them having mostly the same items (can differ by one item or two only). Thus abstraction is more likely to abstract those patterns to the same pattern. However the lower the support value, the smaller the number of users supporting the patterns, which reduces the differences in usage values (recall that Nokia has only 38 users).

These results show that the space reduction given by abstraction is not negligible, and can help reduce the burden on the analyst.

To reach maximal abstraction, in worst case an item may be abstracted at most 3 times, coming from the depth of the Nokia application taxonomy (as shown in Figure 2). It is interesting to see the influence of successive iterations of the abstraction primitive, and the distribution of abstracted items in the different levels of the application taxonomy. This result is presented in Figure 9. For each application of the abstraction primitive, and thus each level of the taxonomy as shown by Figure 2, the percentage of patterns that got abstracted to a class of the taxonomy of that level is shown. The bars correspond to different abstraction thresholds $\rho$, the support threshold is fixed to $\sigma = 25\%$.

One can note that for too low abstraction thresholds ($\rho = 3\%$), 90% of patterns are abstracted to the top level of the taxonomy, which is the least informative: it confirms the poor interest of such low abstraction thresholds. Conversely excessively high abstraction thresholds ($\rho = 90\%$) lead to less that 20 % of patterns abstracted on the lower level of the taxonomy, and near no higher level abstraction: this is not enough to help the analyst. On the other hand, abstraction thresholds between the bounds $M_1$ and $M_2$ defined before lead to a reasonable percentage of patterns abstracted per level, with a decrease of more than 20% of patterns abstracted from level 1 to level 3. This indicates that the analyst will be presented with patterns containing a mixture of classes from the taxonomy, which is what is expected to help in the analysis.

### 3.3 Refinement Evaluation

The goal of refinement is to restrict the number of choices at each step of the exploration so that the analyst is not confronted with thousands of choices. Saliency allows to do it in a principled way, and only present potentially “interesting” choices to the analyst. We run two experiments to evaluate refinement. In the first one, we evaluate quantitatively the practical reduction in the number of choices. In the second one, we evaluate the quality of obtained refinements.

#### 3.3.1 Exploration Choice Reduction

For each pattern $p$, we count the number of exploration choices (i.e. number of valid refinements of $p$), and compute the average result for all patterns. The saliency threshold is fixed to $\mu = 50\%$, and we vary the support threshold $\sigma$. The results are shown in Figure 10 for MovieLens.

As shown in Figure 10, with the lowest possible support threshold $\sigma = 0\%$, there exist in average 13 exploration choices for patterns of MovieLens, which is an extreme case. For more realistic support thresholds such as $3 \leq \sigma \leq 6\%$, there exist between 1 and 3 choices in average: the analyst is not overwhelmed with choices, but is often offered more than one choice, which suffices to allow different directions of exploration. One can note that in this experiment, with $\sigma \geq 10$ there is only 1 choice on average: saliency becomes too strict. For such higher values of support the saliency threshold should be relaxed to get more choices, but then it would be less adapted to lower support values.

#### 3.3.2 Refinement Qualitative Evaluation
Beyond the quantitative aspect of refinement, we evaluate the quality of refinement primitive in an extensive user study. Through the survey, we evaluate the usefulness of refinement primitive by measuring if users find the provided histograms (plotting values of an attribute for support users) with the refinement primitive informative and prefer to observe the histograms that the primitive detects as more salient. We have used 2 patterns from Nokia and 2 from MovieLens.

In a comparative study, we present two histograms for each pattern (a salient and a random non-salient one) and ask the participant which histogram is more useful to be attached to the pattern.

In the second part of our study, we seek user feedback in order to independently evaluate the usefulness and meaningfulness of refinement. We present a pattern with the most salient histogram detected by our measure (standard deviation). We ask the participant if she considers the histogram informative. Table 2 shows overall results.

<table>
<thead>
<tr>
<th></th>
<th>Positive</th>
<th>Negative</th>
</tr>
</thead>
<tbody>
<tr>
<td>Comparative (%)</td>
<td>69</td>
<td>31</td>
</tr>
<tr>
<td>Independent (%)</td>
<td>83</td>
<td>17</td>
</tr>
</tbody>
</table>

Table 2: Qualitative Evaluation

The positive option for the comparative study means preference for the salient histogram, and the negative option, otherwise. Also the positive option for the independent study shows the percentage of participants that have found the associated histograms informative, and the negative option, otherwise.

The result above suggests to consider different saliency measures for different demographic attributes. We plan to explore that in future work.

Figure 12 shows the assessment of how informative histograms are for each pattern. Users found the histograms informative for all 4 patterns. For MOVIELENS patterns (3 and 4) the percentage of not informative responses is close to zero.

4. RELATED WORK

There has been quite a bit of research activity to help analysts explore the space of patterns. We can however safely claim that none of them is data-driven and a lot of work is left to the analyst. Moreover, existing pattern interestingness measures have been used for selecting representative patterns but none of them was used to explore the support users of a pattern. Finally, in both our primitives, we exploit usage data that is not present in the transaction dataset given as input to frequent pattern mining algorithms, but that appears in the original data. This allows us to go beyond what just traditional methods do based on the transaction datasets.

A constraint-based mining approach [5, 6] can also be seen as a pattern exploration mechanism where the analyst can iteratively tune the constraints to generate additional patterns to explore. Designing constraints is not an easy task and requires an appropriate knowledge of the dataset.

An approach is proposed to learn the model of prior knowledge of the analyst based on her exploration actions. In [7] the analyst has to order her exploration choice preference which puts more burden on the analyst. These methods can be complementary to ours to make a refinement biased towards previous choices by the analyst.

We exploit taxonomies for abstraction that helps reduce the space of patterns. Our taxonomy-based usage is an interestingness measure and it has the same principle as defined in previous work [9, 2]. The difference is that we calculate
our measure for items in a pattern and not necessarily for a whole pattern. The reason is that we are not interested in pruning a pattern, but in abstracting parts of it. The method used in [10, 11] is a top-down approach and is the most similar work to our abstraction method.

The idea of refinement for semantic exploration can be categorized as an interestingness mining approach. In our work, salient demographics attributes can be visualized as histograms associated with a pattern. In [12, 13, 14], the idea is to mine a small set of interesting patterns using novel interestingness measures. Those measures are computed for whole patterns and are used to select representative ones as opposed to explore users of a given pattern. A complete list of measures used in the literature can be found here [2].

5. DISCUSSION

In this paper, we addressed the problem that arises when analyzing the behavior of a large number of users with frequent pattern mining, namely the discovery of a large number of patterns and the length of each pattern. We proposed abstraction and refinement primitives to navigate in the space of frequent patterns and better understand their users. Our evaluation on two real datasets showed that abstraction reduces the size of the pattern space and produces more readable patterns. It also shows the usefulness of refinement in guiding the analyst in the exploration of the behavior of well-defined sets of users as dictated by the data.

Our primitives are key for the implementation of an interactive exploration framework for frequent usage patterns. To do so we need to devise an algorithm that combines our primitives in such a way that provides to the analyst a number of alternative navigations in the space of patterns. Instead of relying solely on the lattice induced by pattern mining (pattern generalization and specialization), our algorithm could guide the analyst in exploring different user communities formed by patterns in the lattice. We hence envision an interactive framework within which users alternate between exploring patterns and exploring user communities induced by them and described by a combination of demographics attributes. For example, an analyst could from the pattern Users of Communication and Web Search applications who live in Lausanne and see two alternative subsets of those users, ones who are students and live on EPFL Avenue and use those Messaging applications, and ones who are stay-at-home users and use Google in the afternoon. For each subset, the analyst could ask to see other patterns. This process is iterative and requires the ability to compute patterns and communities on-demand. We are currently exploring the use of scalable indexing techniques to enable such flexibility.

Refinement is a principled way of identifying user communities of interest for which activity is known. We consider this a starting point for exploring subsets of users and plan to use the algorithms developed in [15] and in [16] to do so. There is an opportunity to specialize the exploration depending on the type of action users perform in the underlying dataset. For example, in the case of collaborative rating such as MOVIELENS, rating exploration may require to search for subsets of users whose ratings are uniform or polarized w.r.t. the movies contained in a pattern as in [15] whereas in the case of NOKIA, the duration of usage of the set of applications embedded in a pattern is more appropriate. In our immediate future work, we plan to investigate the applicability of different action-aware exploration algorithms to complement pattern refinement.

6. REFERENCES