Detecting user focus in OLAP analyses

Mahfoud Djedaini, Nicolas Labroche, Patrick Marcel, Veronika Peralta

Detecting user focus in OLAP analyses

Mahfoud Djedaini[1], Nicolas Labroche[1], Patrick Marcel[1], and Verónika Peralta[1]

University of Tours, France
firstname.lastname@univ-tours.fr

Abstract. In this paper, we propose an approach to automatically detect focused portions of data cube explorations by using different features of OLAP queries. While such a concept of focused interaction is relevant to many domains besides OLAP explorations, like web search or interactive database exploration, there is currently no precise formal, commonly agreed definition. This concept of focus phase is drawn from Exploratory Search, which is a paradigm that theorized search as a complex interaction between a user and a system. The interaction consists of two different phases: an exploratory phase where the user is progressively defining her information need, and a focused phase where she investigates in details precise facts, and learn from this investigation. Following this model, our work is, to the best of our knowledge, the first to propose a formal feature-based description of a focused query in the context of interactive data exploration. Our experiments show that we manage to identify focused queries in real navigations, and that our model is sufficiently robust and general to be applied to different OLAP navigations datasets.

1 Introduction

Interactive Data Exploration (IDE) is the task of efficiently extracting knowledge from data even if we do not know exactly what we are looking for [4]. Typically, an exploration includes several queries where the result of each query triggers the formulation of the next one. OLAP analysis of data cubes is a particular case of IDE, that takes advantage of simple primitives like drill-down or slice-and-dice for the navigation. For example, an analyst may explore several attributes and cross several dimensions, in order to find clues, causes or correlations to explain unexpected data values, until identifying the most relevant data subsets and deeply analyzing them. While OLAP has been around for more than 20 years, little is known about typical navigational behavior.

Exploratory Search, however, is a sub-domain of Information Retrieval that studies user behaviors during their explorations [11]. The basic model of exploration distinguishes two main phases. In a first phase, called exploratory browsing, users are likely to explore the space, as well as better defining and understanding their problem. At this stage, the problem is being limited, labeled, and a framework for the answer is defined. Over time, the problem becomes more clearly defined, and the user starts to conduct more targeted searches. In

this second phase, called focused phase, users (re)formulate query statements, examine search results, extract and synthesize relevant information.

Detecting focused phases in an exploration can be exploited in a variety of applications, for instance in the context of user exploration assistants. When focused, an analyst would expect more *precise* queries, related to what she is currently analyzing. On the contrary, when exploring the data, the analyst would prefer more *diverse* queries, for a better data space coverage. Focus detection could also be used in data visualization. In a focus phase, an analyst would prefer a highly focused interface, presenting to her in details what she is currently investigating. Oppositely, an analyst who is exploring the data would rather expect an interface presenting an overview of available data, highlighting the diversity of available dimensions of analysis.

In this paper, we propose an approach to automatically detect focused phases in OLAP explorations. While there exists no formal definition or consensual formula to decide whether an OLAP exploration or a query is focused or not, the concept of focus can be intuitively described by different characteristics that indicate a focused activity. Our hypothesis is indeed that a definition of focus is highly dependent of a fine characterization of the queries composing an exploration. For instance, the granularity level or the number of filters of a query, or the number of OLAP operations that separate two consecutive queries, are such characteristics. In our proposal, we identify a total of *19 characteristics* to finely describe different aspects of a query, either intrinsically, relatively to its predecessor or relatively to the whole exploration containing it. We show that it is possible to define a metric to quantify each of these characteristics. It is then possible to see the central question of defining a formal model of the focus based on the characteristics as a classification problem where the descriptive features are the metrics' scores and the output variable indicates if a query is focused or not. By choosing an appropriate classification approach and well specified metrics, our work demonstrates that it is possible to build an interpretable, yet efficient, model for the focus that is consistent with expert evaluation on real OLAP navigations and predefined behavioral patterns on simulated OLAP navigations that were defined agnostic of any focus definition.

The paper structure is as follows: Section 2 showcases an example for motivating our approach, which is introduced in Section 3. Section 4 describes the formal framework and Section 5 details the metrics used to characterize focus. Section 6 highlights experiments and discusses results. Finally, before concluding, Section 7 presents related works.

2 Motivating example

Our example is taken from the Star Schema Benchmark (SSB) specification [8][1]. This benchmark defines a workload consisting of 4 flights of queries. Each flight

[1] We redirect the reader to the SSB specification for the logical schema and the exact SQL text of the queries. In our example, we consider the instance is generated with a scale factor of 1.

Query flight 3	Q1	Q2	Q3	Q4
Where	year in [1992,1997], c_region=ASIA, s_region =ASIA	year in [1992,1997] c_nation = 'US', s_nation = 'US'	year in [1992,1997] (c_city='UKI1'or 'UKI5'), (s_city='UKI1' or 'UKI5')	yearmonth=Dec97, (c_city='UKI1'or 'UKI5'), (s_city='UKI1' or 'UKI5')
Group by	c_nation, s_nation, year	c_city, s_city, year	c_city, s_city, year	c_city, s_city, year
# cells	150	596	24	3
# tuples examined	200k	8k	329	5

Table 1. Description of SSB query flight 3. The measure sum(lo_revenue) is the same in all queries.

can be seen as an exploration over a 5 dimensional cube whose schema corresponds to the relational star schema defined by the benchmark. We particularly pay attention to flight number 3, that consists of 4 queries Q1, Q2, Q3 and Q4, analyzing revenue volume (see Table 1). Q1 asks for revenue generated for region Asia (for both suppliers and customers) between 1992 and 1997, by nations. It examines over 200,000 tuples of the fact table and produces 150 cells of the cube. Q2 asks for revenue in the United States at the city level, for the same time period. It examines 8000 tuples and produces 596 cells. Q3 remains at the city level but asks for revenue in the United Kingdom, and only for two cities in the UK, examining 329 tuples and producing 24 cells. These two cities remain selected in Q4 that drills the time dimension at the month level, to just one month, examining only 5 tuples and producing 3 cells.

It can be seen that the beginning of the navigation is not focused, while the second half (Q3 and Q4) start to focus on a particular zone in the cube. The benchmark specification actually accounts for this, indicating that the last query was deliberately specified to be a "needle-in-haystack" query. We now review what differentiates the first half of the navigation, which is exploratory, from the second, focused, part.

The first two queries are only loosely related to each other. They move by relatively big jumps in the data space. They are coarse in terms of the filters and the granularity levels used, which results in big portions of the fact table being analyzed to produce relatively large answer sets. This may induce high execution time but surely also high consideration time (time taken by the user to analyze the answer set).

On the contrary, queries 3 and 4 are separated by one OLAP operation (specifically, a drill down operation), and the text of query 4 is obtained from that of query 3 with only a few modifications. The queries become finer, in the sense that more filters are accumulated on finer granularity levels, targeting a smaller portion of the fact table. The result sets are also much smaller. Content-wise, the "needle-in-haystack" effect indicates that this focus is justified by something surprising in the data.

As we will see in the next sections, our study of real navigation logs collected from users corroborate these intuitive considerations. Specifically, we have observed in these logs that longer navigations tend to incorporate quite long focused

sequences, often at the middle or end of the navigation, corresponding to short jumps in the data space. In these focused sequences, queries are close to each other in terms of OLAP operations and text, and their answer sets often share cube cells. Being able to automatically detect such focus zones in navigations has many advantages, for example in experience sharing among users (the needle in the haystack, discovered in a former navigation, may be useful to other analysts) or suggesting recommendations in line with the user's immediate interests, and more generally to make user experience with big datasets less disorienting.

These observations led us to define 3 categories of features to characterize focused zones in navigations. The first category corresponds to intrinsic properties of the query taken independently of the other queries, like the filters, the answer set, etc. The second category positions a query respectively to its immediate predecessor in the navigation, for instance to detect OLAP operations. Finally, the last category positions the query relatively to the navigation itself, for instance to check if the query appears in a long chain of similar queries.

3 Characterizing and detecting focus phases

Our approach aims at automatically detecting focus phases in user explorations. As mentioned above, there is yet no formula for deciding whether a query is focused or not. However, as illustrated by the previous motivation example, an expert is able to recognize a focus activity by looking at various characteristics of the queries and the exploration.

In order to quantify these intuitive characteristics, we define a set of metrics. As such, these metrics characterize different aspects of a query: the user intention (e.g., the desired granularity expressed through the aggregation level), the results (e.g., the number of cube cells retrieved), as well as its relationship to other queries (e.g., the differences between a query and its predecessor).

Then, the question of formally characterizing a focused query can be expressed as a classification problem in which all queries can be represented by scores issued from the metrics and the class output variables is binary, either "focused" or "not focused". These are the only two classes we are able to define regarding the fuzzy notion of focus. The main difficulty in this case relates to the building of a proper corpus of annotated queries by experts, to learn the model from it. Not all the classifiers meet this requirement. Indeed, as the ability to interpret what makes a query focused is a major objective in our work, we limit ourselves to linear models that learn a weight for each metric's score and then output a focus score that is computed as weighted sum over the metrics values for each query. In this context, we use an off-the-shelve SVM classifier whose separative hyperplane equation provides the expected relation to qualify the focus of a query based on our metrics scores and their associated weights. Moreover, this formalization allows to understand in a very intuitive way how each metric contributes to the detection of focus.

As detailed in Section 6, we used a set of real explorations over a real data cube to train and test the classifier. All the queries of all the explorations were

labeled by a human expert, familiar with both the cube explored and the front end tool used for the explorations. Each query is then labeled either as focused or as exploratory. The labels we obtain are used as a ground truth for our classifier.

4 Formal framework

This section introduces the formal framework underlying our approach, in which explorations are treated as first class citizens.

Exploration. An exploration is a triple $\langle e, lts, ets \rangle$, where $e = \langle q_1, \ldots, q_p \rangle$ is a sequence of p OLAP queries, lts is a function that gives for a query its launch time-stamp, and ets is a function that gives for a query its evaluation time-stamp. With a slight abuse of notation, we note $q \in e$ if a query q appears in the exploration e.

During their explorations, users inspect the elements of a cube instance (or simply, cube) retrieved by a query.

Cube model. Without loss of generality, the OLAP queries we consider are dimensional aggregate queries over a data cube [2]. We consider cubes under a ROLAP perspective, where dimensions consist of one or more hierarchies, each hierarchy consisting of one or more levels. Formally, a cube schema consists of: i) n hierarchies, each hierarchy h_i being a set $Lev(h_i) = \{L_i^0, \ldots, L_i^d\}$ of levels together with a *roll-up* total order $\succeq_{h_i}$ of $Lev(h_i)$, ii) a set of *measure* attributes M, each defined on a numerical domain. For a n-dimensional cube, a *group-by set* is an element of $Lev(h_1) \times \ldots \times Lev(h_n)$.

The elements of a cube are called cells.

Cells. Cells are tuples $\langle m_1, \ldots, m_n, meas \rangle$ where the m_i are taken in $Dom(L_i^{j_i})$, $L_i^{j_i} \in Lev(h_i)$, for all i, and $meas$ is a measure value.

Query model. A query $\langle G, P, M \rangle$ is defined by a group by set G (identifying the query granularity), a set of boolean predicates P, and a set of measures M. The answer to a query q, denoted $answer(q)$, is the set of non empty cells whose coordinates are defined by the query group by set and selection predicates.

Among the set of cells in an answer, we distinguish between base cells and aggregated cells. In base cells, m_i are taken in $Dom(L_i^0)$. In aggregate cells, there exists one m_i not in $Dom(L_i^0)$. An aggregate cell $\langle m_1, \ldots, m_n, meas \rangle$ can be defined as the result of a query $\langle \{L_1, \ldots, L_n\}, \{L_1 = m_1, \ldots, L_n = m_n\}, \{m\} \rangle$ for some measure m.

5 Metrics

This section details the metrics we identified to describe quantitatively the different aspects of focus for a query. We organize these metrics in three categories:

i) intrinsic to the query, i.e., only related to the query itself, ii) delta metrics, i.e., dependent on the query's predecessor in the exploration, and iii) contextual, i.e., dependent on the complete exploration, to provide more context to the metric. Each of these 3 categories can be then refined into 3 subcategories. Some of them relate to (T) the text of the query and cube schema, (R) the result of the query and (C) the chronology of the session, be it the time, or the sequentiality. Formal definitions are only given for non trivial metrics. In metric definitions, let q_k be a query in an exploration $e = \langle q_1, \ldots, q_p \rangle$.

Cat	Metric name	Description	Coef
		Intrinsic metrics	
T	*Number of measures (NoM)*	Number of measures used in q_k	0,246
T	*Number of filters (NoF)*	Number of filters (or selections) used in q_k	0,553
T	*Number of aggregations (NoA)*	Number of aggregations (or projections) used in q_k	0,192
T	*Aggregation Depth (ADepth)*	Level of aggregation w.r.t. available cube levels	0,217
T	*Filter Depth (FDepth)*	Ratio of filtered data w.r.t. available cube data	0,147
R	*Number of cells (NoC)*	Number of non null cells in $answer(q_k)$	-0,395
R	*Relevant New Information (RNI)*	Amount of information in $answer(q_k)$	0,068
C	*Execution Time (ExecTime)*	Time taken for executing q_k	0,030
		Delta metrics	
T	*Iterative Edit Distance (IED)*	Edition effort required to get q_k from q_{k-1}	-0,201
R	*Iterative Recall (IR)*	Recall of $answer(q_k)$ w.r.t $answer(q_{k-1})$	0,008
R	*Iterative Precision (IP)*	Precision of $answer(q_k)$ w.r.t $answer(q_{k-1})$	0,203
C	*Consideration Time (ConsTime)*	Time taken by the user to consider $answer(q_k)$	0,084
		Contextual metrics	
T	*Click Per Query (CPQ)*	Number of subsequent queries at distance 1 from q_k	-0,100
T	*Click Depth (CD)*	Length of the query chain q_k belongs to	0,491
R	*Increase in View Area (IVA)*	Amount of new cells in q_k	-0,051
C	*Number of Queries (NoQ)*	Number of queries executed so far in the exploration	0,176
C	*Query relative position (QRP)*	Relative position of q_k within the exploration	-0,057
C	*Query Frequency (QF)*	Frequency at which the DB has been queried so far	0,019
C	*Elapsed Time (ElTime)*	Elapsed time since the beginning of the exploration	0,007

Table 2. Overview of the considered metrics. *For convenience, we also put in this table coefficients (Coef) of the features, that relate to the experiments (see Section 6).*

Intrinsic Metrics This category concerns metrics that are exclusively related to a given query, independently of the exploration the query belongs to. Most metrics of subcategories (T) and (R) follow the intuition that the more focused a user, the more complex and detailed the queries she evaluates and the fewest the number of cells. In other words, if the user carefully chooses measures and filters, and sufficiently drills down, she has a precise idea of what she is looking for. These features can be computed straightforwardly from query text or query results and their definition is omitted due to lack of space. *Aggregation Depth (ADepth)* defines the aggregation depth of the query relatively to the levels of the cube. Consider a cube with l levels, $depth(l_i, h_i)$ being the depth of the level l_i in hierarchy h_i, and noting $l_i \in P$ if level l_i appears in the set P of predicates of query q, $ADepth(q) = \frac{\sum_{l_i \in P} depth(l_i, h_i)}{l}$. *Filter Depth (FDepth)* is computed similarly by considering for each filter its corresponding level. *Relevant New*

Information (RNI) is a measure of entropy of the query result. For a query q_k, it evaluates the quantity of information contained in $answer(q_k)$. Formally, $RNI = 1 - (interest(answer(q_k)))$, where $interest$ measures the interestingness degree as a simple normalized entropy, defined by: $interest(C) = \frac{(-\sum_{i=1}^{m} p(i)\log(p(i)))}{\log(m)}$, with $|C| = m$, $C(i)$ is the i^{th} value of the set C and $p(i) = \frac{C(i)}{\sum_{i=1}^{m} C(i)}$ denotes the i^{th} cell occurrence. *Execution Time (ExecTime)* is also related to query complexity, assuming all queries are executed in the same environment. It is computed as $ExecTime(q_k) = ets(q_k) - lts(q_k)$.

Delta metrics They characterize a query relatively to the previous query in the exploration. Here the intuition is that the closer two consecutive queries, the more they have in common, the more focused the user. *Iterative Edit Distance (IED)* represents the edition effort, for a user, to express the current query starting from the previous one. It is strongly related to OLAP primitives operations, and computed as the minimum number of atomic operations between queries, by considering the operations of adding/removing a measure, drilling up/down, and adding/removing a filter. The considered cost for each observed difference (adding/removing) is the same. *Iterative Recall (IR)* and *Iterative Precision (IP)* are computed as classical recall and precision by considering the current query result cells as the retrieved set, and the previous query result cells as the relevant set. Thus, the larger the intersection between queries in terms of accessed cells, the more focused we are. Formally, $IR(q_k, q_{k-1}) = \frac{(answer(q_k) \cap answer(q_{k-1}))}{answer(q_{k-1})}$ and $IP(q_k, q_{k-1}) = \frac{(answer(q_k) \cap answer(q_{k-1}))}{answer(q_k)}$. We give a default neutral score to the first queries of the exploration, by defining $IR(q_1) = IP(q_1) = 0.5$. We approximate *Consideration Time (CT)* by computing the time between the end of execution of the current query and the beginning of execution of the subsequent one: $ConsTime(q_k) = lts(q_{k+1}) - ets(q_k)$. We fixed $ConsTime$ to a neutral value (the average of all the previous queries) for the last query of the exploration. $ConsTime$ does not take into consideration the size of visualized data, as this is an independent feature (in NoC). The importance granted to the combination of $ConsTime$ and NoC features is delegated to the SVM.

Contextual Metrics Contextual metrics characterize a query relatively to an exploration, and more specifically its position within it. In particular, a query occurring in different explorations, can get different scores for these metrics. The two contextual metrics in subcategory (T) adapt popular activity metrics used in Web Search. In this domain, *Clicks Per Query* is used to evaluate search engines through their Search Engine Results Pages (SERP). Given a SERP, CPQ represents the number of links in this page that have been clicked by the user. We adapt it by considering a click as obtaining a new query that differs in one operation from the current query. This model allows to represents typical user behaviors in front of OLAP systems. Formally, we count the number of queries occurring after q_k in the exploration, that are at edit distance one from q_k: $CPQ(q_k, e) = |\{q_p \in e \mid p > k, IED(q_k, q_p) = 1\}|$. In

web search, *Click Depth* evaluates the number of pages that have been successively visited, by following hyper links, from one result in a SERP. For a given query q_k, we adapt it by calculating the length of the chain of queries starting from q_k that are distant of one OLAP operation from their immediate predecessor, without discontinuity. $CD(q_k, e) = |S_{kp}|$, where S_{kp} is the longest subsequence of exploration e starting at query q_k and ending at query q_p inclusive, such that $\forall q_i, q_{i+1} \in e, IED(q_i, q_{i+1}) \leq 1$. *Increase in View Area (IVA)* characterizes the increase in terms of new cells in $answer(q_k)$ compared to all the cells seen during the previous queries of the exploration. Formally, $IVA(q_k, e) = \frac{|answer(q_k) \setminus \bigcup_{i \in [1,k-1]} answer(q_i)|}{|\bigcup_{i \in [1,k]} answer(q_i)|}$. *Number of Queries (NoQ)* represents the absolute number of previous queries in the exploration. It is useful to capture the correlation between the tediousness of an exploration and the focus. *Query Relative Position (QRP)* allows to capture the influence of the position of the query in the exploration on the focus. We expect queries at the beginning of an exploration to be more exploratory, and the ones at the end to be more focused. It is computed as the rank of the query in the exploration, normalized by the size of the exploration: $QRP(q_k, e) = \frac{k}{|e|}$. *Query Frequency* captures the engagement of the user by measuring how many queries she submits per unit of time. $QF(q_k) = NoQ(q_k)/ElTime(q_k)$. Finally, *Elapsed Time* computes the time from the beginning of the exploration: $ElTime(q_k, e) = ets(q_k) - lts(q_1)$.

6 Experiments

This section presents the setup and outcomes of the experiments we conducted to evaluate our approach. We first discuss to which extent the coefficients learned by the model to weigh each descriptive metric (see Section 3) are consistent with the human expertise. Then, we show that our model, once learned on a dataset, can be generalized to other OLAP navigation datasets without any significant loss in prediction rate.

6.1 Experimental setup

Data set We worked with a real database instance, namely a cube called *MobPro*, built from open data on workers mobility. In *MobPro*, facts represent individuals moves between home and workplace, and dimensions allows to characterize a move depending on its frequency, the vehicle used, the traveled distance, etc. The cube is organized as a star schema with 19 dimensions, 68 levels in total, and 24 measures. 37,149 moves are recorded in the facts table.

User explorations In this experiment, we asked 8 junior analysts (who are students in a master's degree specialized in BI), to analyze the cube using Saiku[2]. They were familiar with OLAP tools, but not necessarily with the data within *MobPro*. We gathered 22 explorations from the system logs. In total, these explorations represent 913 queries.

[2] http://meteorite.bi/products/saiku

Query labeling In order to learn our metric scores weights, we need to label the 913 queries, stating if they are focused or not. The 913 queries have been annotated by one expert, with the help of a web application specifically developed for this purpose. A second expert independently annotated 100 of those queries. Both experts are teachers in the masters degree in BI, and co-authors of this paper. On the 100 queries, a high agreement of 89% has been observed between the two experts, ensuring the representativeness of the 913 labels.

6.2 Model training

Using our set of 913 queries, each described by the 19 features and the label, we trained a linear SVM classifier. The linear SVM outputs coefficients that traduce the relative importance of each feature. As the metrics are not normalized, the weights learned may be due to SVM compensating for initial low or high values of the metrics, and not only due to the relative intrinsic importance of the features. With a reasonable assumption of normal distribution of our metrics, we used a z-score for normalizing each metrics scores independently before training our model. Z-score ensures that for a given feature, each value is expressed relatively to this feature variance.

We used 76% of our data (700 queries) for training our model, while the other 24% (213 queries) constitute the test set. Both training and test sets are described in table 3. We parameterized the SVM so that it performs a 10-fold cross validation while learning on the training set, and obtained an accuracy of 80%.

Description	Training set	Test set
# Queries	700	213
% focus	46,7%	59,2%
% non focus	53,3%	40,8%

Table 3. Description of training and test sets for linear SVM

Model discussion Feature coefficients we obtained are presented in table 2. By observing them, weights can be easily classified into 4 categories, using 2 dimensions that we call polarity and intensity. The impact of a metric on the focus can be positive/negative in terms of polarity, and high/low in terms of intensity. Impact polarity depends on the sign of the coefficient of the metrics, whereas impact intensity depends on the absolute value of the coefficient. Here, we highlight trends and discuss in details some of the features.

A focused analyst has a relatively well defined information need in mind, which is clearly evidenced by the weights discovered. Indeed, among the metrics related to text (T) and results (R), we observe that all the metrics that restrict the perimeter of the analyzed data (like NoF, FDepth, NoA, ADepth, NoM) have

a positive impact on focus. And as expected, metrics that relax the perimeter of analyzed data, like NoC and IVA, appear to have a negative impact on focus.

Metrics that characterize an important move within the data space have a negative impact on focus. IED and IVA are particularly concerned by this. Again, as expected, metrics that measure a closeness between two consecutive queries have a positive impact on the focus. IP is the best representative of that in the sense that its value decreases with the amount of new cells gathered compared to cells in the previous query.

Interestingly, most metrics relative to chronology (C) have little impact on the focus, with the notable exception of Number of Queries, which tends to confirm that focus phases indeed happen after rather long exploratory phases. Another rather surprising finding is that complex metrics in (R) like RNI do not show a significant impact on focus.

More generally, we observe that the importance of a feature is fairly related to the metrics categories and subcategories. No category or subcategory should be ignored, in the sense that all of them include metrics having high weights. A general trend is that metrics relative to the text of the query (T) have in general higher weights, which indicates that focus is highly correlated to the user intention expressed in the query syntax. Likewise, in general, intrinsic metrics tend to have a higher impact on the focus (as seen on NoC, NoF, NoM). But this is counterbalanced by the fact that CD, of category (C) has the second highest weight, meaning that the context of the exploration indeed provides semantics when assessing focus.

6.3 Model performance

We previously described the meaningfulness of the features coefficients provided by the SVM. We also conducted different experiments, described below, to check the robustness of the predictive power of our classifier.

Testing on artificial explorations The objective of this experiment is to validate our model on explorations whose focus is known. For that, we used CubeLoad [9] for generating realistic explorations. CubeLoad takes as input a cube schema and creates the desired number of sessions according to templates modeling various user exploration patterns. Patterns available in Cubeload simulate: (*Goal Oriented*) users with limited OLAP skills pursuing a specific analysis goal, (*Slice And Drill* and *Slice All*) more advanced users navigating with a sequence of slice and/or drill operations, (*Exploratory*) users tracking unexpected results with exploratory sessions.

Following the definition of the patterns, we expect *Goal Oriented* explorations to be highly focused, while *Exploratory* are expected to be much less focused. For this experiment, we used the SSB schema [8] and generated a collection of 49 explorations (500 queries) over it. Table 4 presents, per exploration pattern, the ratio of (non) focused queries as predicted by our algorithm. The average ratio of focused queries per exploration pattern confirms our expectations. *Goal*

Oriented explorations have a much higher ratio of focused queries compared to *Exploratory* ones. We also notice that *Exploratory* and *Slice All* have a similar ratio of focused queries. However, *Exploratory* explorations are much less focused than Slice All ones in terms of average focus intensity. *Slice and Drill* is correctly recognized as an intermediate behavior.

Patterns	Explorative	Goal oriented	Slice and Drill	Slice All
# of explorations	15	13	14	7
# of queries	171	123	126	80
% of focused queries	18,13	64,23	49,21	18,75
avg focus	-0,568	-0,070	-0,193	0,738
stdev focus	1,072	1,024	0,955	0,841

Table 4. Ratio of focus queries in terms of cubeload patterns

Testing on real explorations We created a test set of 213 labeled queries from the 22 explorations we collected over the $MobPro$ cube. We obtained an accuracy of $69,9\%$, meaning that $69,9\%$ of the queries have been similarly classified by the expert and the model. Moreover, our model is pretty balanced, as we could retrieve focused (resp., non focused) queries with an accuracy of $71,4\%$ (resp., $67,8\%$). These scores are good, especially given that a naive classifier that would always predict the focus class would reach an accuracy of $59,2\%$ on the test set.

6.4 Focus vs analyst skills

Analyst skills	A	B	C
min Focus	-0,558	-1,806	-2,366
max Focus	1,657	1,126	-1,174
avg Focus	0,241	-0,240	-1,767
stdev Focus	0,861	0,895	0,513

Table 5. Correlation between exploration focus and analyst skills

In general, a skilled analyst performs more focused explorations as she has a better knowledge of data. The objective of this experiment is to verify that our model is capable of retrieving this correlation.

We used a set of explorations, previously labeled by the same expert who labeled the queries for the focus. Our expert labeled each exploration with one of three labels: A, B, C. These labels categorize the user knowledge acquisition, from A when the user conducts a in depth analysis and benefits from it in terms of

knowledge, to C when the user conducts a very poor analysis and did not benefit from it. We used this as a ground truth. Besides, we used our model to predict the focus of the queries in these explorations. For each query, we predict not only its class (focus or not), but also the degree to which it belongs to the class, by computing the distance to the separation hyperplan found by the SVM. We compute the degree of focus of an exploration as the average of its queries focus degree. After matching explorations degree focus and skill label, we verify that our model is in accordance with intuition. Results are presented in table 5. From this table, it is easy to identify that classes A and C are clearly distinguished by their degree of focus. Users who acquired knowledge conducted more focused explorations in average, with a minimum (resp., maximum) of focus relatively low (resp., high) compared to the others. This reasoning is inversely true for exploration in class C. Class B is an intermediate situation, quite ambiguous, where it cannot be stated clearly that the skill has been mastered or not.

6.5 Computation efficiency

Besides the experiments that validate the robustness of our model, we evaluated the average computation time for each metric. Indeed, as we motivated focus detection as a way to improve user experience, we have to ensure that metrics computation runs in near real time. In average, it appears that for a given query, each metric computation does not require more than a few hundred of milliseconds. In average, the computation of all the metrics for a given query is 695 milliseconds, which is negligible given that the average consideration time for a query result is 11200 milliseconds. Having their metrics scores, the query classification is then instantaneous, which validates our approach.

7 Related Work

Analyzing user sessions has been studied for many years in web search. Recent works aim at characterizing the difficulty of search tasks and detecting variations in sessions. For instance, in [1], Athukorala et al. proposed a method for distinguishing between exploratory phases and lookup phases in the context of Information Retrieval. Their idea consists in experimentally discovering features that can be used for this distinction. They submitted several tasks to participants. Some of these tasks require exploration while other require more simple lookups. Based on objective measurements, they could identify that query length, completion time, and maximum scroll depth (in the browser), are the most distinctive indicators for distinguishing between exploratory/lookup tasks. Although they address a problem very similar to the one we tackle in this paper, they work at the exploration level, while our method gives finer results by characterizing each query of an exploration.

A recent trend in web search is to analyze web search sessions by means of machine learning, and more particularly with classifiers. In [3], the goal is to discover new intent and obtain content relevant to users' long-term interests.

They develop a classifier to determine whether two search queries address the same information need. This is formalized as an agglomerative clustering problem for which a similarity measure is learned over a set of descriptive features (the stemmed query words, top 10 web results for the queries, the stemmed words in the titles of clicked URL, etc.). Perhaps closer to focus detection is the work of Odijk et al. [7] for characterizing user struggling during web searches. They propose a method for distinguishing between users exploring search results from user struggling for satisfying a given information need. They tackle this problem by using an approach similar to what we propose in terms of methodology. They use a bunch of keyword features, and using a large real set of explorations, they trained a machine learning algorithm for learning how to differentiate between the two aforementioned types of explorations.

In the databases domain, however, to the best of our knowledge, only a handful of works were interested in analyzing real database sessions. As noted by Jain et al. [5], there exists only two real world SQL workloads available to the research community: the SQLShare workload [5] and the Sloan Digital Sky Server workload [10]. In [5], authors present their enhanced SQL web based tool. They made their tool available online for several years, permitting users to upload their datasets and perform advanced analysis. From this, they could gather a rich workload of SQL queries, performed on different datasets, by different users. Based on different metrics computed on each single query (length of the query text, number of distinct operators, query runtime, ...), they propose to characterize each query according to its complexity. An interesting aspect of this work is the investigation of metrics for measuring the cognitive load of users that is indirectly translated in the complexity of the queries they issue. Moreover, at the workload scale, authors show that queries have a high diversity, considering a query similarity measure based on the query text. Authors also propose to classify user behaviors as exploratory and analytical, based on the number of queries per dataset. According to them, a user is analyzing when she submits a lot of queries to the same dataset. This simple distinction is rather coarse compared to the 19 dimensions we use in this paper for analyzing the same distinction.

Nguyen et al. [6] propose an approach for discovering the most accessed areas of a relational database to characterize user interests. Their notion of user interest relies on the set of tuples that are more frequently accessed, and is expressed as selection queries (mostly range queries). They use DBSCAN to cluster user interests in the Sky Server dataset. Their similarity metric relies on Jaccard coefficient of the accessed tables and on overlapping of predicates. In this work also, only one metrics (most frequent accessed tuples) is used. Additionally, we note that, being tailored for range queries, their metric is inappropriate for OLAP queries that are mostly dimensional (i.e., point based), due to the nature of the hierarchical dimensions used to select data.

With the growing interest around exploratory search in the context of interactive database exploration, we believe that our work constitute a first important contribution for understanding the different aspects of user navigations in structured data.

8 Conclusion

Exploratory search considers a search as a complex interaction between a user and a system, including exploratory phases and focus phases. In this paper, we highlight the usefulness of detecting which phases a user is currently in, in the context of OLAP exploration of data cubes. We propose an automatic method, based on a state of the art machine learning algorithm, modeling this detection as a classification problem. To our knowledge, our contribution is a pioneer of its kind. We successfully built a model, trained on a relatively large set of real explorations. We validated experimentally our model on a test set of real explorations, as well as on an artificially, driven, state-of-the-art exploration generator. On top of that, we checked the coherence of our model by using it to detect how skilled is a data analyst.

We plan to follow up our investigations in two main directions. First, we will study the use of focus as a score for evaluating the overall quality of an exploration. Second, we plan to generalize our approach to relational databases and more general types of explorations.

References

1. K. Athukorala, D. Glowacka, G. Jacucci, A. Oulasvirta, and J. Vreeken. Is exploratory search different? A comparison of information search behavior for exploratory and lookup tasks. *JASIST*, 67(11):2635–2651, 2016.
2. M. Golfarelli and S. Rizzi. *Data Warehouse Design: Modern Principles and Methodologies*. McGraw-Hill, 2009.
3. R. Guha, V. Gupta, V. Raghunathan, and R. Srikant. User modeling for a personal assistant. In *WSDM*, pages 275–284, 2015.
4. S. Idreos, O. Papaemmanouil, and S. Chaudhuri. Overview of data exploration techniques. In *SIGMOD*, pages 277–281, 2015.
5. S. Jain, D. Moritz, D. Halperin, B. Howe, and E. Lazowska. Sqlshare: Results from a multi-year sql-as-a-service experiment. In *Proceedings of SIGMOD*, pages 281–293, 2016.
6. H. V. Nguyen, K. Böhm, F. Becker, B. Goldman, G. Hinkel, and E. Müller. Identifying user interests within the data space - a case study with skyserver. In *EDBT 2015*, pages 641–652, 2015.
7. D. Odijk, R. W. White, A. H. Awadallah, and S. T. Dumais. Struggling and success in web search. In *Proceedings of CIKM*, pages 1551–1560, 2015.
8. P. E. O'Neil, E. J. O'Neil, X. Chen, and S. Revilak. The star schema benchmark and augmented fact table indexing. In *TPCTC*, pages 237–252, 2009.
9. S. Rizzi and E. Gallinucci. Cubeload: A parametric generator of realistic OLAP workloads. In *CAiSE 2014*, pages 610–624, 2014.
10. A. S. Szalay, J. Gray, A. Thakar, P. Z. Kunszt, T. Malik, J. Raddick, C. Stoughton, and J. vandenBerg. The SDSS skyserver: public access to the sloan digital sky server data. In *Proceedings of SIGMOD*, pages 570–581, 2002.
11. R. W. White and R. A. Roth. *Exploratory Search: Beyond the Query-Response Paradigm*. Morgan & Claypool Publishers, 2009.