A-BI+: A Framework for Augmented Business Intelligence

# A-BI$^+$: A Framework for Augmented Business Intelligence

Matteo Francia, Matteo Golfarelli*, Stefano Rizzi

*DISI – University of Bologna, Viale Risorgimento 2, 40136 Bologna, Italy*

**Abstract**

Augmented reality allows users to superimpose digital information (typically, of operational type) upon real-world objects. The synergy of analytical frameworks and augmented reality opens the door to a new wave of situated analytics, in which users within a physical environment are provided with immersive analyses of local contextual data. In this paper, we propose an approach named A-BI$^+$ (Augmented Business Intelligence) that, based on the sensed augmented context (provided by wearable and smart devices), proposes a set of relevant analytical queries to the user. This is done by relying on a mapping between the objects that can be recognized by the devices and the elements of the enterprise multidimensional cubes, and also by taking into account the queries preferred by users during previous interactions that occurred in similar contexts. A set of experimental tests evaluates the proposed approach in terms of efficiency, effectiveness, and user satisfaction.

*Keywords:* Augmented reality, OLAP, Query recommendation

## 1. Introduction

With the disruptive advances in pervasive computing and industry 4.0, business intelligence is shifting its focus on the integration of (internal) enterprise and (external) contextual data. In this direction, the synergy of analytical frameworks and augmented reality opens the door to *situated analytics*[1], in which users within a physical environment are provided with immersive analyses of local contextual data. Indeed, *augmented reality* (AR), a variation of virtual reality, allows users to superimpose digital information upon real-world objects [2], thus determining an augmented environment. Nowadays digital data returned to users are typically *operational*, meaning that they either describe

*Corresponding author

*Email addresses:* m.francia@unibo.it (Matteo Francia), matteo.golfarelli@unibo.it (Matteo Golfarelli), stefano.rizzi@unibo.it (Stefano Rizzi)

[1]Situated analytics has been defined as spatially-organized data representations attached to relevant entities for the purpose of understanding and decision-making [1]

the current state of the visualized objects (e.g., the temperature of a machine) or suggest the operations to be carried out (e.g., instructions to use the machinery). Conversely, limited attention has been devoted to providing *analytical* reports that can be used by decision makers to evaluate the behavior and performance of the visualized objects from a tactical and strategic point of view, for instance with reference to their past history or to other objects of the same type.

This new goal opens relevant research challenges and revamps many issues related to business intelligence and recommendation systems [3]. Indeed, when working with high-dimensional contextual data (the multidimensional nature of the context is well understood [4, 5]), identifying insightful queries and visualizations is not trivial [6] and requires several research issues to be solved [1]: How can data be sensed and accessed in real time? How is the sensed context mapped to enterprise data? Which data is salient to the user analysis? How do users interact with the retrieved data?

To the best of our knowledge, none of the context-aware recommender systems proposed in the literature addresses the above questions with reference to situated analytics in general, and to AR in particular. While the research on computer vision [7, 8] and situated visualization [9, 10, 11] is vivid, not much has been done to set up a business intelligence process bridging context sensing, data visualization, and decision making.

In this paper we envision and formalize a foundation for *Augmented Business Intelligence* (A-BI$^+$), a framework empowering AR users with context-aware analytical information under visualization and time constraints. The context is modeled as a set of recognizable environment objects (e.g., a machine in a manufacturing environment) plus a set of additional user/environmental information (e.g., the user role and the room temperature). The analytical information returned is tailored to the context currently perceived by the user and comes in the form of reports obtained by running OLAP queries on the enterprise multidimensional cubes. The quantity of data returned must be limited in size and focused on the context to meet performance constraints and be easily interpretable by the user; furthermore, the intrinsic dynamics of AR applications ask for right-time (reasonably a few seconds) responsiveness of A-BI$^+$.

An overview of the A-BI$^+$ framework is given in Figure 1 which shows Bob, a controller working for a company producing fitness equipment and wearing AR smart devices featuring sensors of different types. We assume the pattern recognition capabilities necessary to recognize the context are provided by these smart devices. Bob's task is to optimize the production based on the assembling times of manufacturing devices, also considering the sale volumes of the different products. When he stares at an assembly machine, the AR glasses he is wearing recognize the machinery and some nearby objects (in the picture, a seat being assembled with an exercise bike); a context is generated accordingly, also including data about the current date and time and Bob's position and role.

Our goal in this setting is to suggest to Bob in real time the set of analytical queries over the enterprise multidimensional cubes that are more relevant

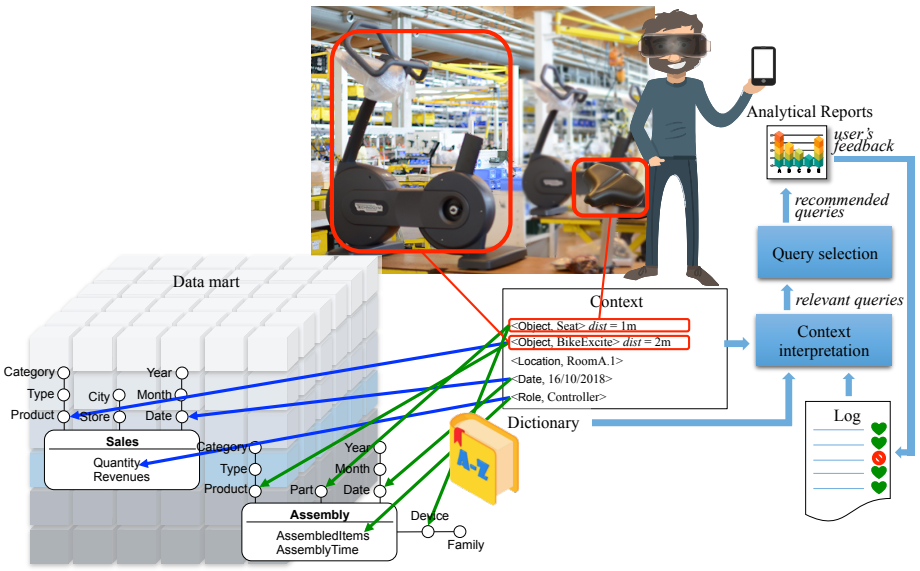


Figure 1: Overview of the A-BI$^+$ framework

according to both a-priori knowledge and feedback given by users in similar contexts. Relevant figures could be the number of produced items, the assembly times, and the revenues for the product being sensed. With reference to Figure 1, this task is carried out by the *Context interpretation* component using a collaborative filtering approach that relies on the query log. Although A-BI$^+$ supports the possibility of learning meaningful queries from the log, its capability of returning the right information primarily comes from some a-priori knowledge provided by domain experts. This choice is not simply a solution to the well-known cold-start problem (i.e., the problem of providing significant recommendations when user feedbacks are still insufficient); it is rather a design choice aimed at enabling the system to give a useful answer in complex context scenarios, where learning from the log would require too many examples. The a-priori knowledge is modeled through a set of mappings between the potentially recognizable environment objects (stored in a *dictionary*) and the multidimensional elements of the enterprise cubes. Rather than proposing the most relevant query only, A-BI$^+$ proposes a set of alternative queries to Bob; all of them are related to the current context but they are different enough to offer to the user different flavors of the same information. This phase is implemented by the *Query selection* component. At this time, Bob can either execute one of the proposed queries or express a new query to obtain a different report. Finally, Bob gives his feedback on the proposed queries, which is stored in the log.

A-BI$^+$ can be applied to different application domains ranging from healthcare [12] to factories [13, 14]; for this reason, the main modeling choices under-

lying our approach (e.g., how to define the relevance of an object in a context) have been formalized in a domain-independent fashion, while domain-dependent examples are provided in the context of AR in a factory where fitness equipment is produced.

To sum up, the main contributions of this paper are:

1. We envision an A-BI$^+$ framework, its functional architecture, and the user/system interaction process.
2. We explain how a-priori expert knowledge can be modeled by mapping context objects to relevant multidimensional elements.
3. We describe an efficient algorithm to generate relevant and diverse queries to be returned to the user.
4. We propose a collaborative filtering approach to let the system learn from user feedback.

A-BI$^+$ extends our previous contribution [15], named A-BI, mainly by (i) generalizing queries to operate on multiple cubes (*drill-across queries* in the OLAP terminology) to better fit real decision-making contexts; (ii) generalizing the object-to-cube mappings to map onto sets of multidimensional elements, so as to enhance their expressiveness; (iii) giving a new definition of query relevance and the corresponding formalization of the query selection problem as an optimization problem; and (iv) providing an extensive experimental evaluation based on a real manufacturing environment.

The remainder of the paper is organized as follows. Section 2 describes the related work in the field of context-aware recommendation systems. Section 3 formalizes the A-BI$^+$ framework. Sections 4 and 5 describe the context interpretation and query selection components, respectively. Section 6 describes the results of experimental tests that measure the performance of A-BI$^+$. Finally, Section 7 sums up our contribution and gives future research directions.

## 2. Related work

The A-BI$^+$ framework can be classified as a *recommender system* in the area of *business intelligence* based on a context made of *augmented entities*. Despite the huge amount of work in these areas, to the best of our knowledge, no approach lies at their intersection.

Over the years, scholars have highlighted the importance of exploiting contextual information to provide focused recommendations with the nature of contexts being quite heterogeneous (a summarized description in provided in Table 1), for instance space and time [27], query logs [18, 19], statistics on results [23, 24] or databases [21], user interests [16], and social data [20]. Given such heterogeneity, other contributions (e.g., [30, 31, 20]) address the integration of contextual data to provide a common ground (e.g., a global schema [31] or an application programming interface [20]) enabling recommendation from

| | User input | Context | MD | RT | C | D |
|---|---|---|---|---|---|---|
| [16] | OLAP query | User profile | ✓ | | | |
| [17] | OLAP session | OLAP session log | ✓ | ✓ | | |
| [18] | SQL query | Query logs | | ✓ | | ✓ |
| [19] | OLAP query | Dashboards, reports | ✓ | | | |
| [20] | SPARQL query | Web documents | | | | |
| [21] | SQL query | Database statistics | | ✓ | | ✓ |
| [22] | SQL query | Result feedback | | | ✓ | |
| [23] | SQL query | Result statistics | | | | |
| [24] | SQL query | Result statistics | | ✓ | | ✓ |
| [25] | Web query | Clicks, query log | | | | |
| [26] | Web query | Clicks, query log | | | | ✓ |
| [27] | Web query | Location | | ✓ | | ✓ |
| [28] | Web query | Query logs | | | | |
| [29] | OLAP query | Query logs | ✓ | | | |
| A-BI$^+$ | none | Physical env., log | ✓ | ✓ | ✓ | ✓ |

Table 1: Comparison of recommender systems in terms of user input, context type, recommendation of multidimensional OLAP queries (MD), real-time constraint (RT), cardinality constraint (C), and query diversification (D)

multiple data sources. The previous context types have been widely adopted in several applications where the recommendation process is activated by an explicit user-defined input statement (e.g., query or keywords). Examples of applications are web query categorization [28, 17], recommendation [16, 25], and diversification [26]; query completion [18, 19]; localized web keywords suggestion [27]; and interactive exploration of databases [22]. The main differences between A-BI$^+$ and these works are: (1) the multidimensional nature of the data handled and returned, (2) the nature of the context as well as the type of user/system interaction that triggers the recommendation, and the presence of (3) real-time, (4) cardinality and (5) diversification constraints. As to (1), multidimensional and hierarchical data support recommendations at different granularities, which intrinsically makes finding the best recommendation more complex; as to (2), physical contexts require ad-hoc solutions to choose the relevant context elements due to application specificities (e.g., object engagement) and to the possibility of having in the context elements that are perceived but that are not relevant for the user; as to (3), (4), (5), these constraints are required by immersive applications [1].

Recommender systems in business intelligence applications are well surveyed in [32]. Recommendations typically involve multidimensional queries [29] or sessions [17] (i.e., query sequences) using query logs as contexts. These approaches are based on collaborative filtering techniques that do not synthesize new queries from existing ones, but pick queries from the log depending on their similarity score. Conversely, A-BI$^+$ allows the generation of queries not already present in the log by combining similar queries from the log and contextual information into a set of diverse queries. This assumption collocates A-BI$^+$ as a hybrid ap-

proach to recommendation [32], differentiating A-BI$^+$ from the above-mentioned contributions in multidimensional recommendation systems. Note that diversification and multiple recommendations are used to better meet user interests [33]. A further advantage of A-BI$^+$ over pure collaborative filtering approaches is that A-BI$^+$ does not suffer from the so-called cold start problem, since it is able to return an appropriate recommendation even when the log is empty [34].

In the area of AR and situated analytics, contexts play an even more central role. There, a context is the set of objects recognized in the environment that acts as situated stimulus (i.e., object properties) to be translated into inputs for a search query; it is augmented with virtual information and is returned to the user [10]. Physical environment becomes a source of contextual information in [35], where user interaction with a physical environment is leveraged to retrieve operational data of interest. The usage of AR as an interactive medium opens to a natural data exploration and is especially helpful when the analysis goals are not specified [11]. Scholars focus on finding proper visualization to embed operational data in physical objects [9, 10, 11], with a particular effort on the implementation of toolkit allowing the rapid prototyping of such visualizations [36]. Although [9] considers multidimensional data, it is not specified how the process of information retrieval and analysis of data at multiple level of details is carried out. Besides, these approaches do not include collaborative filtering to discover potentially useful information. Interestingly, although [9] does not consider analytical data, it introduces a mantra for situated analytics: "details first, analysis, then context-on-demand" which contradicts the well-known mantra "overview first, zoom and filter, then details-on-demand" [37] of classical visualization systems. Indeed, when it comes to pure augmented visualizations, information is directly attached to single objects [9, 10], assigning higher priority to details than to generic information.

## 3. Preliminaries

We start this section by introducing a formal setting to manipulate multidimensional data; for simplicity we will consider linear hierarchies only.

**Definition 1 (Hierarchy).** *A* hierarchy *is defined as a triple* $h = (L_h, \succeq_h, \geq_h)$ *where:*

*(i)* $L_h$ *is a set of categorical* levels*; each level* $l \in L_h$ *is coupled with a* domain $Dom(l)$ *including a set of* members *(all domains are disjoint);*

*(ii)* $\succeq_h$ *is a* roll-up *total order of* $L_h$*; and*

*(iii)* $\geq_h$ *is a* part-of *partial order of* $\bigcup_{l \in L_h} Dom(l)$.

*Exactly one level* $dim(h) \in L_h$*, called* dimension*, is such that* $dim(h) \succeq_h l$ *for each other* $l \in L_h$*. The part-of partial order is such that, for each couple of levels* $l$ *and* $l'$ *such that* $l \succeq_h l'$*, for each member* $u \in Dom(l)$ *there is exactly one member* $u' \in Dom(l')$ *such that* $u \geq_h u'$.

For instance, for a temporal hierarchy Time, it is Date $\succeq_{\mathsf{Time}}$ Month $\succeq_{\mathsf{Time}}$ Year and 02/12/2019 $\geq_{\mathsf{Time}}$ Dec2019 $\geq_{\mathsf{Time}}$ 2019.

**Definition 2 (Group-by Set).** *Given a set of hierarchies $H$, a* group-by set *of $H$ is a subset $G$ of levels in the hierarchies of $H$ including at most one level for each hierarchy. The roll-up order on the hierarchies of $H$ induces a partial order on the group-by sets of $H$ as follows:*

$$G \succeq G' \Leftrightarrow \forall l' \in G' \ \exists l \in G \ s.t. \ l \succeq_h l'$$

**Definition 3 (Cube and Data Mart).** *Given a set of hierarchies $H$, a* cube *over $H$ is defined as a triple $c = (G_c, M_c, \omega_c)$ where:*

*(i)* $G_c$ *is a* group-by set *of $H$;*

*(ii)* $M_c$ *is a set of numerical measures; each measure $m \in M_c$ is coupled with one aggregation operator $op(m) \in \{\mathtt{sum}, \mathtt{avg}, \ldots\}$; and*

*(iii)* $\omega_c$ *is a partial function that maps the tuples of members for the levels of $G_c$ to a numerical value for each measure $m \in M_c$.*

*A* data mart *is a couple of a set of hierarchies $H$ and a set $C$ of cubes over $H$, $\mathcal{M} = (H, C)$.*

The levels, members, and measures of cube $c$ are given the generic name of *md-elements* of $c$. Note that the $\omega_c$ is defined as partial since cubes are normally *sparse*; the *cardinality* of $c$ is the number of tuples of members that are mapped through $\omega_c$ and is denoted with $|c|$.

**Example 1 (Data Mart).** *Our working example (Figures 1 and 2) includes two cubes,* Sales *and* Assembly*, which share hierarchies* Product *and* Date*; the two cubes are completed by hierarchies* Store *and* Device*, and* Part*. Sales are described by measures* Quantity *and* Revenues*, while Assembly is described by measures* AssembledItems *and* AssemblyTime*; all measures are additive, i.e., they are coupled with the* `sum` *operator. A member of the* Part *level is* Seat*, while a member of* Product *is* BikeExcite*.* □

In the A-BI$^+$ framework, cubes are queried through GPSJ (Generalized Projection / Selection / Join) queries, a well-known class of queries that was first studied in [38]. A GPSJ query is composed of joins, selection predicates, and aggregations. Remarkably, having all the cubes in $\mathcal{M}$ defined over the same set of hierarchies $H$ corresponds to assuming that the cubes share a set of *conformed dimensions*, which enables the formulation of *drill-across queries* (queries joining two or more cubes).

**Definition 4 (Query).** *A query $q$ on data mart $\mathcal{M} = (H, C)$ is a triple $q = (G_q, P_q, M_q)$ where $G_q$ is a group-by set of $H$; $P_q$ is a set of Boolean equality clauses defined over members of levels of $H$ whose conjunction defines the selection predicate for $q$; $M_q$ is the set of measures whose values are returned by $q$. Let $C_q \subseteq C$ be the subset of cubes such that at least one of their measures is part of $M_q$; query $q$ is well-formed if*

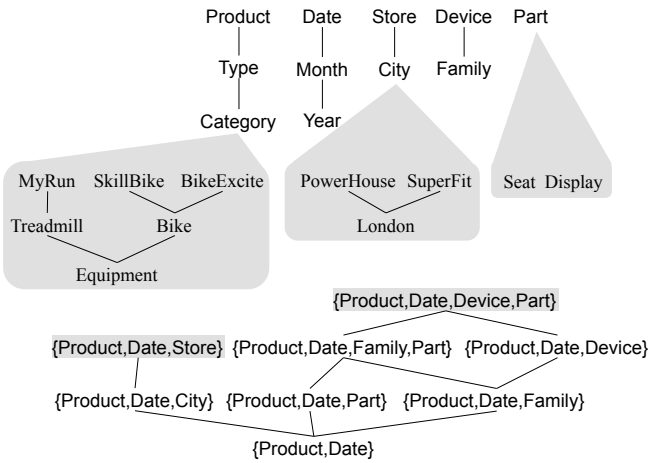


Figure 2: Roll-up total orders (top), part-of partial orders (middle), and an excerpt of the group-by set partial order (bottom) for our working example; the group-by sets of the Sales and Assembly cubes are in gray

*(i) all predicates in $P_q$ are* external*, i.e., they are expressed on levels that are less or equal to a level in $G_q$ in the roll-up order [39]; and*

*(ii) all the measures it returns are available at the required granularity:*

*(a) $\forall c \in C_q \ G_c \succeq G_q$ if $M_q \neq \varnothing$;*
*(b) $\exists c \in C$ s.t. $G_c \succeq G_q$ if $M_q = \varnothing$* [2]

In the following, we will often need to denote the md-elements to which query $q = \langle G_q, P_q, M_q \rangle$ *refers*; specifically,

- if $l \in G_q$, then $q$ refers to level $l$;
- if $m \in M_q$, then $q$ refers to measure $m$;
- if $(l = u) \in P_q$, then $q$ refers to member $u$ and to level $l$, being $l$ the level whose domain includes $u$.

The set of md-elements to which $q$ refers will be denoted with $ref(q)$.

**Example 2 (GPSJ query).** *With reference to the* Sales *cube, the query asking for quantity of sold products for each month of 2019 and each product type is $q = \langle G_q, P_q, M_q \rangle$, with*

$$\begin{aligned} G_q &= \{\mathsf{Month}, \mathsf{Type}\} \\ P_q &= \{(\mathsf{Year} = 2019)\} \\ M_q &= \{\mathsf{Quantity}\} \end{aligned}$$

[2] We assume that, when $M_q = \varnothing$, query $q$ asks for a count of the cardinality of $G_q$, so the existence of at least a cube at the required granularity ensures that $q$ is well-formed.

*The SQL formulation of $q$ on a star schema featuring fact table* `FT_Sales` *and dimension tables* `DT_Date` *and* `DT_Product` *would be*

```
SELECT DT_Date.Month, DT_Product.Type, sum(FT_Sales.Quantity)
FROM FT_Sales
JOIN DT_Date ON (FT_Sales.DateId = DT_Date.DateId)
JOIN DT_Product ON (FT_Sales.ProductId = DT_Product.ProductId)
WHERE DT_Date.Year = 2019
GROUP BY DT_Date.Month, DT_Product.Type
```

*For this query it is*

$$ref(q) = \{\mathsf{Month}, \mathsf{Type}, \mathsf{Year}, \mathsf{Quantity}, 2019\}$$

*By adding measure* AssembledItems*, from the* Assembly *cube, to $M_q$ we get an example of a drill-across query whose SQL formulation is*

```
SELECT DT_Date.Month, DT_Product.Type,
       sum(FT_Sales.Quantity), sum(FT_Assembly.AssembledItems)
FROM FT_Sales
JOIN DT_Date ON (FT_Sales.DateId = DT_Date.DateId)
JOIN DT_Product ON (FT_Sales.ProductId = DT_Product.ProductId)
JOIN FT_Assembly ON (FT_Assembly.DateId = DT_Date.DateId AND
       FT_Assembly.ProductId = DT_Product.ProductId)
WHERE DT_Date.Year = 2019
GROUP BY DT_Date.Month, DT_Product.Type
```

□

Enterprise cubes are the data source for the analytical reports to be returned to users according to the environment as perceived by the AR device. The set of data possibly perceived are listed in a dictionary that, intuitively, defines the device capabilities. These data are not limited to physical objects recognized in the environment through a pattern recognition process, but may include user-related information such as the user role as well as environmental properties such as the room temperature.

**Definition 5 (Dictionary).** *A* dictionary *$\mathcal{D}$ is a set of classes, each coupled with a domain of values. Each pair $d = \langle class, value \rangle$ such that value belongs to the domain of class is called an* entry *of $\mathcal{D}$.*

Note that the dictionary can describe the sensed environment at different levels of precision. For example, if the smart device that perceives the environment successfully identifies a bike, but is not capable of labeling the specific bike

model, it will return the $\langle \mathsf{Object}, \mathrm{Bike} \rangle$ entry. On the other hand, if the specific product BikeExcite is recognized, the smart device will return the entry $\langle \mathsf{Object}, \mathrm{BikeExcite} \rangle$.

The power of the A-BI$^+$ framework comes from the ability to bind the perceived entries to the cube md-elements. This capability is rooted in a-priori knowledge that specifies which md-elements can be interesting for the user when a given dictionary entry is perceived. This knowledge is defined through a dictionary-to-cube mapping established by a domain expert at setup time. To enhance the expressiveness of our framework, we consider that some md-elements may be interesting non *per se* but only when associated with other md-elements, so we map entries not on simple sets of md-elements but on sets of *fragments*, each fragment being a set of md-elements that should appear all together in queries.

**Definition 6 (Mapping).** *A* mapping *from dictionary $\mathcal{D}$ to data mart $\mathcal{M}$ is a multivalued function $\mu$ that maps an entry $d$ to a set of fragments, i.e., sets of md-elements of the cubes in $\mathcal{M}$. Each fragment $f \in \mu(d)$ has a mapping weight, $w_{map}(d, f) \in (0, 1]$.*

The mapping function is multivalued since many fragments of md-elements may be of interest for each dictionary entry; this typically happens for hierarchy levels, which can be all interesting —even if with different values of $w$. For example, when some device is perceived, besides showing data for that specific device, also showing aggregated data for the device type could be interesting. Through mapping weights, domain experts give an a-priori quantification of the interest of each fragment for analyses when a given entry is part of the context.

Although a discussion about how the dictionary is created and the mappings are established is out of the scope of this paper, we remark that this does not necessarily have to be done manually for all the cube members, which would be tedious for attributes with large domains, but it can be largely automated. For instance, to reduce the user's effort in populating the dictionary, an approach like the one proposed in [40] could be used. There, continual learning is applied to classify known objects and to learn objects of never-seen classes. Once sensed and marked as relevant, novel objects can be easily learned and added to the dictionary. Giving novel objects names equal or similar to names of md-elements ensures that a set of basic mappings from the dictionary to the data mart can be automatically created, to be possibly fine-tuned later by a domain expert. Alternatively, tentative mappings could be established by providing universally-quantified rules such as $\mu(\langle \mathsf{Object}, value \rangle) = \{\{value\}\}$ for each value that corresponds to a member of some level in a hierarchy.

**Example 3 (Dictionary and Mappings).** *The dictionary for our example includes, among the others, entries related to products (e.g., $\langle \mathsf{Object}, \mathrm{BikeExcite} \rangle$), product parts (e.g., $\langle \mathsf{Object}, \mathrm{Seat} \rangle$), and user roles (e.g., $\langle \mathsf{Role}, \mathrm{Controller} \rangle$). An excerpt of the mapping from this dictionary to the*

*cubes of Example 1 may look like this (Figure 1):*

$$\begin{aligned}\mu(\langle \mathsf{Object}, \mathrm{Seat}\rangle) &= \{\{\mathrm{Seat}\}, \{\mathsf{Device}\}\},\\ \mu(\langle \mathsf{Object}, \mathrm{BikeExcite}\rangle) &= \{\{\mathrm{BikeExcite}\}\},\\ \mu(\langle \mathsf{Role}, \mathrm{Controller}\rangle) &= \{\{\mathsf{AssembledItems}, \mathsf{Quantity}\}\},\\ \mu(\langle \mathsf{Date}, 16/10/2018\rangle) &= \{\{\mathsf{Date}\}\}\end{aligned}$$

*The first line returns two fragments including a member and a level respectively; it states that, when the user senses a seat, she may be interested in analyzing either the part to be assembled or the data concerning the assembly device. The second line returns one fragment including a member; it states that, when the user senses a product, it is normally interested in analyzing the data for that product. The third line returns one fragment including two measures; it states that controllers are interested in comparing the number of assembled items with the quantity sold. The fourth line returns one fragment including a level; it states that users are normally interested in daily data. Finally, examples of mapping weights are*

$$\begin{aligned}w_{map}(\langle \mathsf{Object}, \mathrm{BikeExcite}\rangle, \{\mathrm{BikeExcite}\}) &= 0.5\\ w_{map}(\langle \mathsf{Role}, \mathrm{Controller}\rangle, \{\mathsf{AssembledItems}, \mathsf{Quantity}\}) &= 1\end{aligned}$$

□

## 4. Context interpretation

In this section, we show how, given a set of perceived objects, A-BI$^+$ produces a set of *relevant queries*, i.e., queries whose results may be of interest to the user.

### *4.1. Take the context...*

The A-BI$^+$ starting point is the *context*, i.e., a set of dictionary entries corresponding to the currently perceived environment objects. More formally:

**Definition 7 (Context).** *A* context $T$ *over dictionary* $\mathcal{D}$ *is a set of entries of* $\mathcal{D}$*; each entry* $d \in T$ *is coupled with a* context weight $w_{cnt}(T, d) \in (0, 1]$.

The value of the weight for each entry may depend on different factors, depending on the application domain. Non-perceived entries (i.e., for which it would be $w_{cnt}(T, d) = 0$) are not included in the context. In our case study we assume that a subset of entries are *engaged*, meaning that they have explicitly been indicated by the user as being part of her current focus of interest [8]; for these entries, the weight is always 1. For the other entries, the weight is inversely proportional to the distance between the user and the specific object being observed.

Given a context, the mapping function identifies the relevant fragments of md-elements, i.e., those that will be involved in the queries to be issued against the cube.

**Definition 8 (Image).** *Given context $T$ over dictionary $\mathcal{D}$ and mapping $\mu$ from $\mathcal{D}$ to data mart $\mathcal{M}$, the* image *of $T$ through $\mu$ is the set of fragments that are mapped through $\mu$ from the entries in $T$:*

$$I_\mu(T) = \bigcup_{d \in T} \mu(d)$$

**Example 4 (Context).** *A possible context is the one depicted in Figure 1, where Bob is inspecting the assembly of fitness equipments in Room A.1 on October 16th 2018:*

$$\begin{aligned} T = \{&\langle \mathsf{Object}, \text{Seat} \rangle, \\ &\langle \mathsf{Object}, \text{BikeExcite} \rangle, \\ &\langle \mathsf{Role}, \text{Controller} \rangle, \\ &\langle \mathsf{Location}, \text{RoomA.1} \rangle, \\ &\langle \mathsf{Date}, 16/10/2018 \rangle\} \end{aligned}$$

*The BikeExcite product is engaged. Possible context weights are*

$$\begin{aligned} w_{cnt}(T, \langle \mathsf{Object}, \text{Seat} \rangle) &= 0.6, \\ w_{cnt}(T, \langle \mathsf{Object}, \text{BikeExcite} \rangle) &= 1, \\ w_{cnt}(T, \langle \mathsf{Role}, \text{Controller} \rangle) &= 1, \\ w_{cnt}(T, \langle \mathsf{Location}, \text{RoomA.1} \rangle) &= 0.6, \\ w_{cnt}(T, \langle \mathsf{Date}, 16/10/2018 \rangle) &= 0.6 \end{aligned}$$

*The image of $T$ through the mapping $\mu$ described in Example 3 is*

$$I_\mu(T) = \{\{\text{Seat}\}, \{\mathsf{Device}\}, \{\text{BikeExcite}\}, \{\mathsf{AssembledItems}, \mathsf{Quantity}\}, \{\mathsf{Date}\}\}$$

□

The image includes the set of fragments relevant to a context according to the mapping, but it does not specify how they will be used to generate the queries to be proposed to the user when that context is sensed. Indeed, given an image, several queries can be generated, each including a subset of the fragments in the image.

*4.2. ...add the log...*

The a-priori knowledge expressed through a mapping does not enable the system to learn by considering how user interests evolve, which instead could lead to picking different md-elements when proposing queries or to choosing one of them more/less frequently. To this end, A-BI$^+$ exploits the history of previous interactions, stored in the query log, by means of a collaborative filtering approach. The log stores, for each context, all the queries proposed to the user and the specific one chosen for execution.

**Definition 9 (Log).** *A log $L$ is a multiset of triples $\langle T, q, ok\rangle$ where $T$ is a context, $q$ is a query, and $ok$ (feedback) is 1 if the user accepted the query, $-1$ if she rejected the query.*

**Example 5 (Log).** *A possible log for our working example is $L = (\langle T, q_1, -1\rangle, \langle T, q_2, 1\rangle)$, where*

$$\begin{aligned} q_1 = \langle &\{\mathsf{Date}, \mathsf{Part}, \mathsf{Product}\},\\ &\{(\mathsf{Product} = \text{BikeExcite})\},\\ &\{\mathsf{AssembledItems}, \mathsf{AssemblyTime}\}\rangle\\ q_2 = \langle &\{\mathsf{Month}, \mathsf{Part}, \mathsf{Product}\},\\ &\{(\mathsf{Product} = \text{BikeExcite})\},\\ &\{\mathsf{AssembledItems}, \mathsf{AssemblyTime}\}\rangle \end{aligned}$$

*While $q_1$ has been rejected, $q_2$ (which is a roll-up of $q_1$ on the* Date *hierarchy) has been accepted.* □

A log entry related to context $T'$ should impact the recommendations related to the current context $T$ only if the two contexts are similar, since it is reasonable to assume that the user will have similar behaviors in similar contexts.

**Definition 10 (Context Similarity).** *Given two contexts $T$, $T'$ over dictionary $\mathcal{D}$, we define the* similarity *between $T$ and $T'$ as their Jaccard index:*

$$sim(T, T') = \frac{|T \cap T'|}{|T \cup T'|}$$

Given log $L$, the image $I_\mu(T)$ of context $T$ is extended to take previous user interactions into account as follows. Let $L_T \subseteq L$ be the subset of log triples whose context is similar to $T$:

$$L_T = \{\langle T', q, ok\rangle \in L \text{ s.t. } sim(T, T') \geq \epsilon\}$$

where $\epsilon$ is a similarity threshold. Then, $I_\mu(T)$ is extended by adding, for each query $q$ in $L_T$, one fragment corresponding to the set of md-elements referred to by $q$:

$$I^*_\mu(T) = I_\mu(T) \cup \{ref(q) : \exists\langle T', q, ok\rangle \in L_T\}$$

In this way, $I^*_\mu(T)$ includes all the fragments that are relevant to context $T$ *either according to the mapping or to the previous user experience.*

We now define the *log relevance* to $T$ of fragment $f \in I^*_\mu(T)$ as the weighted number of times $f$ has been accepted by the user (i.e., $ok = 1$) over the number of times it has been proposed (i.e., $ok = *$); weighting is based on the similarity between the current context $T$ and the considered log context $T'$:

$$\rho_T(L, f) = \frac{1 + \sum_{\langle T', q, 1\rangle \in L_T(f)} sim(T, T')}{2 + \sum_{\langle T', q, *\rangle \in L_T(f)} sim(T, T')}$$

where $L_T(f) = \{\langle T', q, ok\rangle \in L_T \text{ s.t. } f \sqsubseteq ref(q)\}$ and $\sqsubseteq$ is a hierarchy-aware containment relationship between sets of md-elements:

$$\begin{aligned} f \sqsubseteq ref(q) \Leftrightarrow\ & (\forall m \in f \text{ s.t. } m \text{ is a measure, } m \in ref(q)) \wedge \\ & (\forall u \in f \text{ s.t. } u \text{ is a member, } \exists u' \in ref(q) \text{ s.t. } u' \geq_h u) \wedge \\ & (\forall l \in f \text{ s.t. } l \text{ is a level, } \exists l' \in ref(q) \text{ s.t. } l' \succeq_h l) \end{aligned}$$

To avoid relevance to be 0 when $f$ has never been accepted, a Laplace smoothing is applied in the formula above. Noticeably, the impact of Laplace smoothing decreases as the cardinality of $L_T(f)$ increases, that is, the weight tends to 0 if several queries referring $f$ have been proposed but never accepted by the user. Conversely, it tends to $\frac{1}{2}$ if only a few queries referring $f$ have been proposed.

It is now possible to define the *relevance* to $T$ of each fragment $f \in I^*_\mu(T)$ by taking into account, for each context entry $d$ that maps to $f$, not only the entry weight $w_{cnt}(T, d)$ and the mapping weight $w_{map}(d, f)$, but also the log relevance $\rho_T(L, f)$:

$$rel_T(f) = \rho_T(L, f) \cdot \left( \sum_{d \in T \text{ s.t. } f \in \mu(d)} w_{cnt}(T, d) \cdot w_{map}(d, f) \right)$$

where $w_{cnt}(T, d) = w_{map}(d, f) = \frac{1}{2}$ for all $f \in I^*_\mu(T) \setminus I_\mu(T)$. Clearly, the reason for providing a default value for all fragments present in the extended image but not deriving from the context is to avoid the corresponding contribution to the relevance to be null; choosing the default value of $\frac{1}{2}$ is in line with the Laplace smoothing applied to the log relevance.

**Example 6 (Extended image).** *With reference to the image $I_\mu(T)$ from Example 4 and to the log entries in Example 5, the extended image is*

$$\begin{aligned} I^*_\mu(T) = \{&\{\text{Seat}\}, \{\mathsf{Device}\}, \{\text{BikeExcite}\}, \{\mathsf{AssembledItems}, \mathsf{Quantity}\}, \{\mathsf{Date}\}, \\ &\{\mathsf{Month}, \mathsf{Part}, \mathsf{Product}, \text{BikeExcite}, \mathsf{AssembledItems}, \mathsf{AssemblyTime}\}, \\ &\{\mathsf{Date}, \mathsf{Part}, \mathsf{Product}, \text{BikeExcite}, \mathsf{AssembledItems}, \mathsf{AssemblyTime}\}\} \end{aligned}$$

*The last two fragments have been added to $I_\mu(T)$ as they corresponds to queries drawn from contexts similar to $T$. As to $\rho_T(L, f)$ and $rel_T(f)$ it is*

$$\begin{aligned} &\rho_T(L, \{\mathsf{Date}, \mathsf{Part}, \mathsf{Product}, \text{BikeExcite}, \mathsf{AssembledItems}, \mathsf{AssemblyTime}\}) = 0.33 \\ &\rho_T(L, \{\mathsf{AssembledItems}, \mathsf{Quantity}\}) = 0.5 \\ &rel_T(\{\mathsf{Date}, \mathsf{Part}, \mathsf{Product}, \text{BikeExcite}, \mathsf{AssembledItems}, \mathsf{AssemblyTime}\}) = 0.16 \\ &rel_T(\{\mathsf{AssembledItems}, \mathsf{Quantity}\}) = 0.5 \end{aligned}$$

□

We close this section by observing that the log size will quickly increase with time and spending a few words about how the log can be curated. Indeed,

in [41] it is highlighted that many recommended queries can become irrelevant (e.g., in case of sensible context variations such as room refurnishment) or non-computable (e.g., due to changes in the multidimensional schema). A basic way to deal with memory limitations when storing large logs would be to provide a log cache, with size limits, where only the latest entries are cached per user. A more sophisticated way would be to adopt an *indicator of obsolescence* as in [41] to decide whether to prune obsolescent log entries.

*4.3. ...get the queries*

As already stated, the context interpretation component is in charge of generating a set $Q_T$ of queries relevant to context $T$. In principle, the query that includes *all* the md-elements in the fragments of the extended image of $T$ might be directly proposed to the user. Unfortunately, when several objects are sensed in the environment and the context includes a large number of entries, such queries would be *monster queries*, i.e., quite complex queries with very high cardinalities. Monster queries are particularly undesirable in AR applications since:

- High-cardinality queries take a long time to be computed, transferred to the user smart device, and visualized.
- While working on the field, users must be quick and reactive, while the results of monster queries are hardly interpretable.

In the A-BI$^+$ framework, monster queries are avoided in two ways: (i) by posing an upper bound $\gamma$ to the query cardinality, and (ii) by considering only the most relevant fragments in the image when generating the queries to be proposed to the user.

As to (i), given query $q = (G_q, P_q, M_q)$, the expected cardinality of its result, denoted $card(q)$, can be estimated as follows:

$$card(q) = card(G_q) \times \prod_{p \in P_q} sel(p)$$

where $card(G_q)$ is the expected cardinality of a query with group-by set $G_q$ and no selection predicates, and $sel(p)$ is the selectivity of each simple predicate $p$ belonging to $P_q$. Note that we can safely use this formula to estimate $card(q)$ because, as a consequence of the way we create queries in our approach, all predicates in $P_q$ are always *external*, i.e., they are expressed on levels that are less or equal to a level in $G_q$ [39] in the roll-up partial order. As to $card(G_q)$, it must be computed taking the cube sparsity into account, considering that the sparsity differs from cube to cube. In the simple case in which all measures in $M_q$ belong to a single cube $c_i$, it can be estimated for instance using the Cardenas formula as shown in [42, 43]:

$$card(G_q) = card_{c_i}(G_q) = \Phi(card_{max}(G_q), |c_i|)$$

where $|c_i|$ is the cardinality of $c_i$ and $card_{max}(G_q)$ is the maximum cardinality (i.e., if there were no sparsity) of group-by set $G_q$:

$$card_{max}(G_q) = \prod_{l \in G_q} |Dom(l)|$$

If $q$ is a drill-across query,[3] the measures in $M_q$ are scattered across two or more cubes $c_1, \ldots, c_r$; in this case the sparsities of these cubes can be assumed to be mutually independent:

$$card(G_q) = card_{max}(G_q) \cdot \prod_{i=1}^{r} card_{c_i}(G_q)/card_{max}(G_q)$$

where $card_{c_i}(G_q)/card_{max}(G_q)$ expresses the probability that a given tuple of members is present in $c_i$, thus the product expresses the probability of that tuple to be present in *all* the involved cubes. Note that other approaches have been devised for a more precise cardinality estimation in presence of selection predicates on multiple attributes, for instance [44], which uses singular value decomposition, and [45], based on histograms.

**Example 7 (Cardinality).** *Given cubes* Assembly *and* Sales*, let* $|\mathsf{Assembly}| = 200000$*,* $|\mathsf{Sales}| = 400000$*,* $|Dom(\mathsf{Product})| = 100$*, and* $|Dom(\mathsf{Date})| = 1000$*. Consider query*

$$\begin{aligned} q = \langle & \{\mathsf{Date}, \mathsf{Product}\}, \\ & \{(\mathsf{Year} = 2019)\}, \\ & \{\mathsf{AssembledItems}, \mathsf{Quantity}\}\rangle \end{aligned}$$

*By applying the formulas above we get*

$$\begin{aligned} card_{max}(\{\mathsf{Date}, \mathsf{Product}\}) &= 100000 \\ card_{\mathsf{Assembly}}(\{\mathsf{Date}, \mathsf{Product}\}) &= 86467 \\ card_{\mathsf{Sales}}(\{\mathsf{Date}, \mathsf{Product}\}) &= 98169 \\ card(\{\mathsf{Date}, \mathsf{Product}\}) &= 84884 \end{aligned}$$

*Assuming that* $sel(\mathsf{Year} = 2019) = 0.25$*, we get* $card(q) = 21221$*.* □

As to (ii), i.e., considering only the most relevant fragments in the image, before we proceed we remove from the extended image $I^*_\mu(T)$ all the fragments $f$ whose relevance $rel_T(f)$ is below a given threshold $\eta$, being the relevance defined as follows.

[3] We recall from Section 3 that drill-across queries can be formulated because all cubes in the data mart share a set of conformed dimensions.

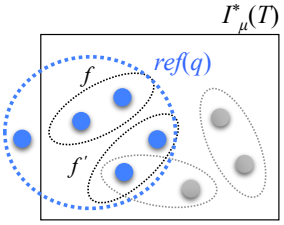


Figure 3: The relevance of $q$ (in blu its md-elements) is the sum of the relevances of fragments $f$ and $f'$

**Definition 11 (Relevant Queries and Query Relevance).** *Given context $T$, query $q$ is said to be* relevant to $T$ *if (i) it is $f \subseteq ref(q)$ for at least one fragment $f$ in the extended image of $T$, (ii) $q$ is well-formed, and (iii) $card(q) \leq \gamma$. The set of relevant queries to $T$ is denoted with $Q_T$. The* relevance *to $T$ of query $q \in Q_T$ in presence of log $L$ is defined as*

$$rel_T(q) = \sum_{f \in I^*_\mu(T) \ s.t. \ f \subseteq ref(q)} rel_T(f)$$

For instance, with reference to Figure 3, the relevance of $q$ is estimated by summing up the relevances of fragments $f$ and $f'$ as these are the only fragments of $I_\mu(T)$ completely contained in $ref(q)$.

**Example 8 (Query relevance).** *Given the mapping weights in Example 3, the context weights in Example 4, the log relevance in Example 5, and the extended image in Example 6, the relevance of*

$$\begin{aligned} q = \langle &\{\mathsf{Month}, \mathsf{Part}, \mathsf{Product}\}, \\ &\{(\mathsf{Product} = \text{BikeExcite})\}, \\ &\{\mathsf{AssembledItems}, \mathsf{AssemblyTime}\}\rangle \end{aligned}$$

*is $rel_T(q) = 0.4$. The fragments contributing to the query relevance (i.e., those contained in $ref(q)$) are* $\{\text{BikeExcite}\}$ *and* $\{\mathsf{Month}, \mathsf{Part}, \mathsf{Product}, \text{BikeExcite}, \mathsf{AssembledItems}, \mathsf{AssemblyTime}\}$. □

In the remainder of this section we explain how we create the set $Q_T$ of relevant queries, as introduced in Definition 11, to be handed to the query selection component. Basically, in Algorithm 1 we follow a depth-first enumeration approach [46] to generate all possible combinations of the fragments in $I^*_\mu(T)$. This is done by calling the recursive procedure *Expand*, whose pseudocode is shown in Algorithm 2, for each fragment available. Within *Expand*, function $Gen(f)$ (Line 1) returns a query $q$ using the md-elements in fragment $f$ as follows:

- $G_q$ includes all the levels in $f$ plus the levels of all the members in $f$;
- $P_q$ includes a selection predicate on each member in $f$;

**Algorithm 1** Generation of relevant queries

**Require:** $I^*_\mu(T)$: extended image, $\gamma$: cardinality threshold
**Ensure:** $Q_T$: set of relevant queries
1: $Q_T \leftarrow \varnothing$ ▷ Result set
2: $F \leftarrow I^*_\mu(T)$ ▷ Fragment set
3: **for each** $f \in F$ **do** ▷ For each fragment...
4: $F \leftarrow F \setminus \{f\}$
5: $Expand(F, f)$ ▷ ...generate relevant queries from $f$ and add them to $Q_T$
6: **return** $Q_T$

**Algorithm 2** Procedure $Expand(F, f)$

**Require:** $F$: set of fragments, $f$: fragment to be used for generating queries
1: $q \leftarrow Gen(f)$ ▷ Generate a query from $f$
2: **if** $wellFormed(q) \wedge q \notin Q_T$ **then**
▷ If $q$ is not well-formed and has already been generated, stop
3: **if** $card(q) \leq \gamma$ **then** ▷ If $q$ has low cardinality...
4: $Q_T \leftarrow Q_T \cup \{q\}$ ▷ ...add it to $Q_T$
5: **for each** $f' \in F$ **do** ▷ For each other fragment in $F$...
6: $F \leftarrow F \setminus \{f'\}$
7: $Expand(F, f \cup f')$ ▷ ...add it to $f$ and generate relevant queries

- $M_q$ includes all the measures in $f$.

To avoid redundancies in $G_q$ and $P_q$, only levels and members at the finest granularity are kept for each hierarchy. If the query returned by $Gen(f)$ is not already present in $Q_T$, has low cardinality, and is well-formed, it is added to the result (Line 4). Then, recursion is triggered by calling *Expand* with the union of $f$ and any other available fragment (Lines 5-7, parameter $F$ is passed by value).

Remarkably, if $q$ is not well-formed (Line 2), the current branch of recursion can safely be pruned. Indeed, a query is not well-formed when either (i) it has non-external predicates or (ii) it returns a measure that is not available at the required granularity (see Definition 4). As to (i), we note that $Gen(q)$ adds to the query group-by set the levels of all the members in $f$, so it cannot generate queries with non-external predicates. As to (ii), when proceeding with the recursion, new md-elements would be added to $q$; this may make the granularity of $q$ finer but it cannot make it coarser. Thus, the queries obtained by adding further md-elements to a query $q$ that is not well-formed can never be well-formed. The current branch can also be pruned if $q$ has already been added to $Q_T$; in this case, since we are adopting a depth-first approach, the extension of $q$ with further fragments has already been done as well. We finally note that cardinality cannot be used to prune the search space; indeed, by adding further md-elements to a high-cardinality query, we get a new query whose cardinality might be below the threshold $\gamma$.

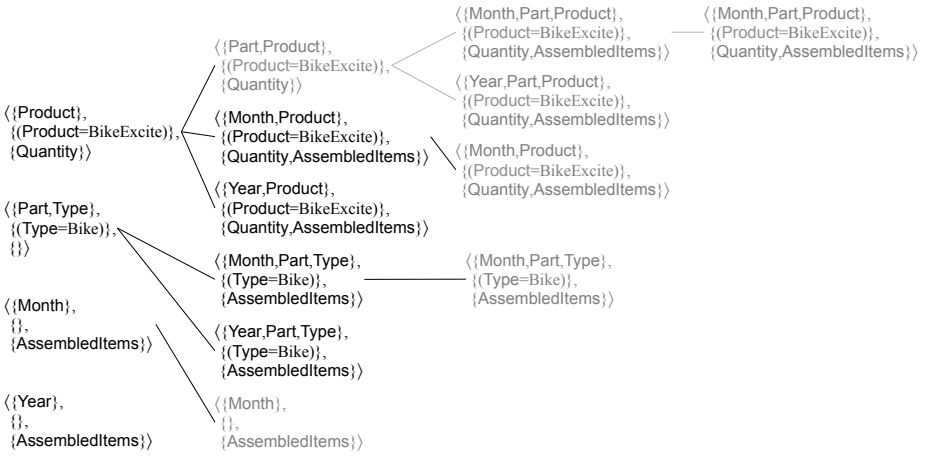


Figure 4: Depth-first query generation for Example 9 (gray arcs are safely pruned, so gray queries are not added to $Q_T$

**Example 9 (Query generation).** *Let*

$$I^*_\mu(T) = \{\{\text{BikeExcite}, \mathsf{Quantity}\}, \{\mathsf{Part}, \text{Bike}\}, \\ \{\mathsf{Month}, \mathsf{AssembledItems}\}, \{\mathsf{Year}, \mathsf{AssembledItems}\}\}$$

*As sketched in Figure 4, Algorithm 1 works as follows. The first fragment picked by the for cycle of Line 3 is* $\{\text{BikeExcite}, \mathsf{Quantity}\}$*, which corresponds to query* $q = \langle\{\mathsf{Product}\}, \{(\mathsf{Product} = \text{BikeExcite})\}, \{\mathsf{Quantity}\}\rangle$*. This query is well-formed and it has cardinality equal to 1, so it is added to* $Q_T$*. The depth-first exploration continues by picking fragment* $\{\mathsf{Part}, \text{Bike}\}$ *(Algorithm 2, Line 5) and calling Expand on the union fragment* $\{\text{BikeExcite}, \mathsf{Quantity}, \mathsf{Part}, \text{Bike}\}$*. Since measure Quantity is not defined at the part granularity, the query obtained is not well-formed and this branch of recursion is pruned (Algorithm 2, Line 2). The next fragments picked by Line 5 of Algorithm 2 are* $\{\mathsf{Month}, \mathsf{AssembledItems}\}$ *and* $\{\mathsf{Year}, \mathsf{AssembledItems}\}$*, which are expanded as shown in Figure 4.*

*Once the recursion started from* $\{\text{BikeExcite}, \mathsf{Quantity}\}$ *is terminated, the other three fragments are picked by Line 3 of Algorithm 1. In particular, when* $\{\mathsf{Month}, \mathsf{AssembledItems}\}$ *is picked, the corresponding query is added to the result (assuming it has low cardinality). Now, the fragment is expanded with* $\{\mathsf{Year}, \mathsf{AssembledItems}\}$ *(Algorithm 2, Line 5), producing the union fragment* $\{\mathsf{Month}, \mathsf{AssembledItems}, \mathsf{Year}\}$*; remarkably, since the corresponding query is already present in the result (*$\mathsf{Year}$ *is coarser than* $\mathsf{Month}$*, so it is removed by* $Gen(f)$*), this branch is pruned.* □

## 5. Query selection

Context interpretation returns the set $Q_T$ of relevant queries, whose results may be of interest to the user. This set is potentially exponential in the number

of fragments; as we will show in Figure 12, discarding ill-formed and high-cardinality queries does not drastically reduce the cardinality of $Q_T$, which may easily turn out to be several thousands. Clearly, some selection has to be done to avoid flooding the user with tons of (probably very similar to each other) queries. Thus, the goal of the step discussed in this section is to select from $Q_T$ a fixed number $rq$ of queries to be recommended to the user. The guiding criterion is to select the subset of queries that are both maximally relevant to the context and diverse; this is done by defining an ad-hoc measure of *query set relevance* that takes both relevance and diversity into account.

To this end, we start by generalizing to sets of queries the definition of similarity given for query pairs in [47]. This definition combines three components: one related to group-by sets, one to selection predicates, and one to measure sets. Let $levBelow(G_q)$ denote the set of levels that are below a level of $G_q$ in the roll-up order:

$$levBelow(G_q) = \{l' : l \succeq_h l',\ \text{for } l \in G_q\}$$

and $memBelow(P_q)$ denote the set of members that are below a member of $P_q$ in the part-of order:

$$memBelow(P_q) = \{u' : u \geq_h u',\ \text{for } u \in P_q\}$$

**Definition 12 (Query Similarity).** *Given context $T$, let $Q \subseteq Q_T$. The* similarity *of the queries in $Q$ is defined as*

$$sim(Q) = \alpha \cdot \sigma_{gbs}(Q) + \beta \cdot \sigma_{sel}(Q) + \gamma \cdot \sigma_{meas}(Q)$$

*where $\alpha$, $\beta$, and $\gamma$ are normalized to 1 and*

$$\sigma_{gbs}(Q) = \frac{|\bigcap_{q \in Q} levBelow(G_q)|}{|\bigcup_{q \in Q} levBelow(G_q)|}$$

$$\sigma_{sel}(Q) = \frac{|\bigcap_{q \in Q} memBelow(P_q)|}{|\bigcup_{q \in Q} memBelow(P_q)|}$$

$$\sigma_{meas}(Q) = \frac{|\bigcap_{q \in Q} M_q|}{|\bigcup_{q \in Q} M_q|}$$

Like in [47], we choose $\alpha = 0.35$, $\beta = 0.5$, and $\gamma = 0.15$.

**Example 10 (Query similarity).** *Let $Q = \{q', q'', q'''\}$, with*

$q' = \langle \{\mathsf{Product}, \mathsf{Month}\}, \{(\mathsf{Type} = \text{Bike})\}, \{\mathsf{AssembledItems}, \mathsf{Quantity}\} \rangle$

$q'' = \langle \{\mathsf{Product}, \mathsf{Month}\}, \{(\mathsf{Category} = \text{Equipment})\}, \{\mathsf{AssembledItems}, \mathsf{AssemblyTime}\} \rangle$

$q''' = \langle \{\mathsf{Type}\}, \{(\mathsf{Category} = \text{Equipment})\}, \{\mathsf{AssembledItems}\} \rangle$

*Considering the roll-up and part-of orders in Figure 2 and the involved md-elements (see Table 2), it is $sim(Q) = 0.35 \cdot \frac{2}{5} + 0.5 \cdot \frac{1}{2} + 0.15 \cdot \frac{1}{3} = 0.44$.* □

Table 2: Md-elements for computing quest similarity in Example 10; intersecting md-elements are in bold

| | **md-element** | $q'$ | $q''$ | $q'''$ |
|---|---|---|---|---|
| $\bigcup_{q \in Q} levBelow(G_q)$ | Product | ✓ | ✓ | |
| | **Type** | ✓ | ✓ | ✓ |
| | **Category** | ✓ | ✓ | ✓ |
| | Month | ✓ | ✓ | |
| | Year | ✓ | ✓ | |
| $\bigcup_{q \in Q} memBelow(P_q)$ | Bike | ✓ | | |
| | **Equipment** | ✓ | ✓ | ✓ |
| $\bigcup_{q \in Q} M_q$ | **AssembledItems** | ✓ | ✓ | ✓ |
| | AssemblyTime | | ✓ | |
| | Quantity | ✓ | | |

Based on the definition of query similarity, we can now define the relevance to the context of any set of queries. The global relevance of a set of queries cannot be properly computed as the sum of their relevances due to the intersections between their md-elements, thus we have to apply the well-known inclusion-exclusion principle [48]. The inclusion-exclusion principle counts the number of distinct elements in the union of finite sets by summing up the cardinalities of the individual sets, subtracting the number of elements that appear in at least two sets, adding back the number of elements that appear in at least three sets, and so on. Similarly, to estimate the global relevance of a set of queries we sum the relevances of individual queries, subtract the average query relevance weighted by their similarity for any pair of queries, add back the average query relevance weighted by their similarity for any triple of queries, and so on.

**Definition 13 (Query Set Relevance).** *Given context $T$, let $Q \subseteq Q_T$. The* relevance *to $T$ of $Q$ is defined as*

$$rel_T(Q) = \sum_{\varnothing \neq Q' \subseteq Q} sim(Q') \cdot \frac{\sum_{q \in Q'} rel_T(q)}{|Q'|} \cdot (-1)^{|Q'|+1}$$

**Example 11 (Query set relevance).** *With reference to Figure 5, given $Q = \{q, q'\}$ such that $rel(q) = 0.8$, $rel(q') = 0.7$, and $sim(Q) = 0.2$, it is $rel_T(Q) = rel_T(q) + rel_T(q') - sim(q, q') \cdot (rel_T(q) + rel_T(q'))/2 = 1.35$.* □

Applying the inclusion-exclusion principle in Definition 13 ensures that, at a parity of relevances of the single queries, the more diverse these queries are, the higher the query set relevance. Thus, selecting from $Q_T$ the subset $R$ of $rq$ queries with the maximum query set relevance implicitly allows A-BI$^+$ to recommend queries that are both relevant to the context and diverse. More formally:

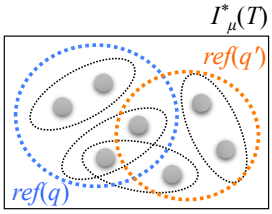


Figure 5: Estimation of query set relevance in Example 11

**Algorithm 3** Query selection

**Require:** $Q_T$: set of relevant queries, $rq$: number of queries to be recommended
**Ensure:** $R$: recommended query set

1: $R \leftarrow \varnothing$ ▷ Result set
2: $Q \leftarrow Q_T$ ▷ Search space
3: **while** $(|R| < rq) \wedge (Q \neq \varnothing)$ **do**
▷ Still room in $R$ and search space not empty
4: $q \leftarrow argmax_Q(rel_T(R \cup \{q\}))$ ▷ Pick the most promising query...
5: $Q \leftarrow Q \setminus \{q\}$ ▷ ... remove it from the search space
6: $R \leftarrow R \cup \{q\}$ ▷ ... and add it to the result
7: **return** $R$

**Problem 1 (Query Selection).** *Given context $T$, select the subset $R$, $\varnothing \neq R \subseteq Q_T$, such that $rel_T(R) \geq rel_T(R')$ for each $\varnothing \neq R' \subseteq Q_T$.*

The query selection problem can be mapped to a *Weighted Maximum Coverage Problem* (WMCP) where each $q \in Q_T$ corresponds to a set of elements (i.e., fragments), each with its own weight (i.e., relevance). We want to find out the subset $R \subseteq Q_T$ with maximal weight and such that $|R| < rq$. The reduction in relevance due to query similarity is taken into account in the WMCP, which adds only once the weight of elements appearing in more than one $q \in R$. It is easy to verify that also every WMCP can be mapped to a query selection problem where (i) for each element a new fragment $f$ is created, (ii) for each set of elements a new query $q \in Q_T$ is created, and (iii) $rel_T(f)$ is set to the weight of the element corresponding to $f$.

Since the two problems can be mapped to each other, they must have the same complexity. In [49] it is shown that the WMCP is NP-hard and that it can be faced with polynomial complexity by adopting a greedy algorithm that, at each iteration, picks the most promising element; so we adopt the greedy approach whose pseudocode is shown in Algorithm 3. Basically, at each iteration we pick from $Q_T$ the query that, if added to the result $R$, maximizes the query set relevance $rel_T$ of $R$ (at the first iteration, this equals to picking the most relevant query). The algorithm is incremental, so queries can be recommended as soon as they are picked —without having to wait for the algorithm to terminate.

**Example 12 (Query selection).** *Let $Q_T = \{q', q'', q'''\}$ such that $rel(q') =$*

0.7, $rel(q'') = 0.6$, $rel(q''') = 0.5$, $sim(q', q'') = 0.6$, $sim(q', q''') = 0.1$, *and* $sim(q'', q''') = 0.2$. *After initialization, Algorithm 3 picks* $q'$ *from* $Q_T$ *(Line 4) as it has top query relevance so it also maximizes query set relevance. At the second iteration,* $q'''$ *is picked at Line 4: although* $rel(q'') > rel(q''')$, $q''$ *is more similar to* $q'$ *than* $q'''$, *thus the query set relevance if* $q''$ *is added to* $R$ *is lower (*$rel_T(q', q'') = 0.91$, $rel_T(q', q''') = 1.14$*). Assuming* $rq = 2$, *query selection stops here.* □

## 6. Experimental tests

In this section, we evaluate the A-BI$^+$ framework in terms of (1) effectiveness, (2) efficiency, and (3) user satisfaction (i.e., to what extend the recommended queries meet the users' desiderata).

As to (1) and (2), we compare A-BI$^+$ to our previous implementation A-BI [15]. Tests are carried out against a synthetic benchmark since, in this work, we assume the problem of context generation to be addressed by the smart device and, to the best of our knowledge, no AR open dataset exists. The user-system interaction works as follows: a session simulates a user walking through a factory of 10 rooms. While moving, she collects one view of each room (in a session each room is visited once). From each view, the smart device recognizes a set of objects belonging to the dictionary and lists them into a context. For each context, A-BI$^+$ recommends a set $R$ of queries to the user; she either chooses one of these queries (i.e., she gives a positive feedback for one of the recommended queries) or formulates an additional query that is slightly different from the ones proposed (i.e., she gives a negative feedback for all the queries and adds a new one). After some time, the user ends her exploration of the factory (i.e., her session). When a new user enters the factory (i.e., a new session begins), A-BI$^+$ relies on the query log to recommend a new set of queries that better suit her interests. Since each session covers 10 rooms, after each session 10 contexts are added to the log together with the corresponding user feedback. The contexts related to each room may be slightly different, since the user could perceive the room from a different point of view, or the smart device could fail to recognize some of the objects.

This interaction is simulated by randomly generating 10 seed contexts, each corresponding to a different room. Seed contexts differ significantly from each other. Then, to simulate multiple visits of each room, small context variations are generated starting from each seed (i.e., different room perspectives). The number of objects recognized in each room (i.e., the context cardinality) ranges between 10 and 16. The test is repeated 10 times and the average behavior is considered.

We denote with $s$ the number of sessions, i.e., the number of times each room has already been sensed. Besides, for each context, we call:

- $q_{max}$ the query with maximal relevance in $R$. We recall that Algorithm 3 always recommends the set of $rq$ well-formed queries with the highest relevance to the context.

Table 3: Notation summary

| Notation | Meaning |
|---|---|
| $T$ | Context (corresponds to a view of a factory room) |
| $R$ | Set of recommended queries |
| $rq = \|R\| \in [1, 4]$ | Number of recommended queries |
| $q_{max}$ | Query with maximal relevance to the context |
| $q_u$ | User query |
| $q_{div}$ | Query most similar to $q_u$ |
| $s \in [0, 8]$ | Number of times the user has already sensed a context |
| $\delta \in [0.5, 1]$ | Similarity between $q_u$ and $q_{max}$ |
| $\gamma = 1000$ | Query cardinality threshold |
| $\epsilon = 0.8$ | Context similarity threshold |
| $\eta = 0.2$ | Fragment relevance threshold |
| $\theta \in [0.05, 0.25]$ | Diversification threshold for A-BI |

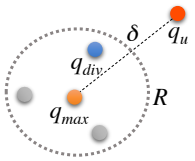


Figure 6: The user query $q_u$, the maximal query $q_{max}$, and the set $R$ of recommended queries; among them, $q_{div}$ is the one most similar to $q_u$

- $q_u$ the query formulated by the user. We denote with $\delta$ the similarity between the user query and the maximal query ($\delta = sim(\{q_u, q_{max}\})$, as of Definition 12). The lower the value of $\delta$, the higher the difference between the user and maximal queries; if $\delta = 1$, the user exactly chooses the maximal query.

- $q_{div}$ the query most similar to $q_u$ among those in $R$.

A notation summary is provided in Table 3.

Queries $q_u$ and $q_{max}$ can be different since the relevance initially estimated by A-BI$^+$ might not be aligned with the user's perceived one. This gap, evaluated in Section 6.3, decreases as the user returns in the same rooms since A-BI$^+$ can exploit the log to align its estimation of relevance to the user's one. An intuitive representation of $q_u$, $q_{max}$, and $q_{div}$ is shown in Figure 6, where the query space is represented as a Cartesian plane with Euclidean distances.

We executed our tests against a cube including 5 linear hierarchies with 5 levels each. Each dimension has 64 members, determining a maximum cube cardinality of about $10^9$. The dictionary includes one entry for each md-element (i.e., we assume the smart device can recognize every single element of the cube); each dictionary entry $d$ is mapped to a fragment $f$ containing at most 3 md-elements. Mapping weights $w_{map}(d, e)$ randomly range in $[0.2, 1]$. Note that A-BI$^+$ entails mappings with higher expressiveness than A-BI, where dictionary

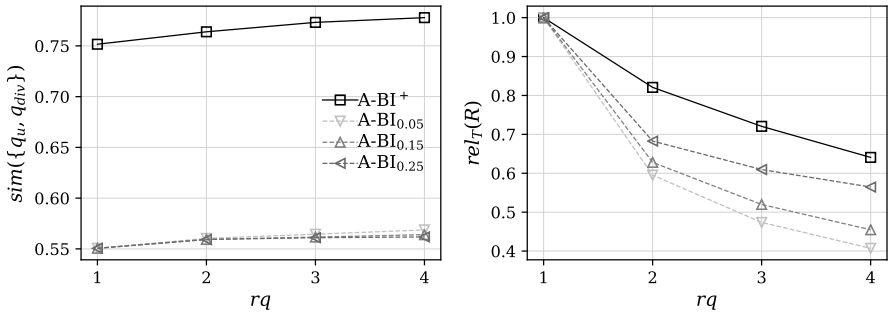


Figure 7: Average similarity between $q_{div}$ and $q_u$ (left) and average relevance of the recommended query set (right) for increasing values of $rq$ ($s = 0$, $\delta = 0.7$, $|T| = 12$)

entries were mapped to single md-elements. Also, while in A-BI$^+$ diversification is inherently tied to the maximization of the relevance of the returned queries, in A-BI it is ruled by a specific parameter, $\theta$. We will compare the two approaches using different thresholds of diversification $\theta \in [0.05, 0.25]$; in the figures, with A-BI$_{0.15}$ we denote a run of A-BI with a diversification threshold set to 0.15. Values of $\theta$ higher than 0.25 deviate too much from the queries related to the context and are not considered [15].

### *6.1. Effectiveness*

A-BI$^+$ can recommend a variable number of queries, $rq$. The higher $rq$, the larger the user effort in choosing the best query out of the recommended ones. In an AR context, due to real-time and visualization constraints, this aspect becomes even more critical; thus, we limit the maximum number of retrieved queries to $rq = 4$.

Figure 7 characterizes $R$ when different numbers of relevant queries are recommended to the user. The left part of the figure shows that the recommendation effectiveness, measured as the similarity between the user's query $q_u$ and $q_{div}$, improves as $rq$ increases. Remarkably, the similarity between $q_{div}$ and $q_u$ is always higher for A-BI$^+$ than for A-BI, independently of the diversification strength $\theta$ used by A-BI. A-BI$^+$ overcomes A-BI due to (1) its enriched mapping expressiveness; (2) the improved algorithm for generating relevant queries (Algorithm 2); and (3) the implicit diversification process. As to (1), given two md-elements that are only relevant if picked together, in A-BI$^+$ they can only appear together in a query, while in A-BI they may be added to the query individually. As to (3), the diversification effectiveness in A-BI$^+$ is better understood from Figure 7 (right), which shows that the relevance of the recommended query set is always superior to A-BI, independently of the diversification strength $\theta$.

Although Figure 7 (left) shows that the similarity between $q_{div}$ and $q_u$ slightly increases with $rq$, the actual impact of diversification will be better appreciated in Section 6.3, where we will see that the users preferred a recommended query different from the most relevant query $q_{max}$ in 15% of the times.

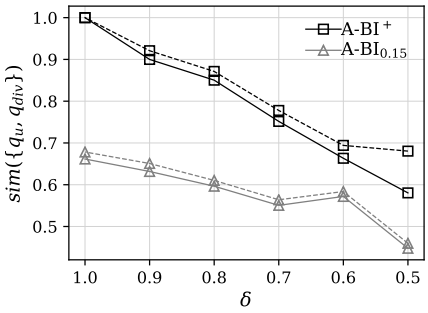


Figure 8: Average similarity between $q_{div}$ and $q_u$ as $q_u$ diverges from the context; solid and dashed lines refer to $rq = 1$ (no diversification) and $rq = 4$ (diversification), respectively ($s = 0$, $|T| = 12$)

This confirms the benefit of diversification in offering users different query flavors among which to choose [50]). Clearly, when there is a low correlation between the user's interest and the context, it becomes hard for a recommender to return useful answers. In our tests this divergence between the context and the user's query is simulated by increasing $\delta$; Figure 8 shows that even in this case A-BI$^+$ improves over A-BI, and that diversification helps in mitigating the correlation gap.

As rooms are repeatedly visited, collaborative filtering comes into play and the effectiveness of A-BI$^+$ improves. Figure 9 depicts to what extent the query log helps in making $q_{div}$ closer to $q_u$. In a real scenario, both the context and the user query $q_u$ could slightly change in different visits. Figure 9 compares the recommendation effectiveness when the user query is fixed (left) and when the context is fixed (right). It is apparent that, when $q_u$ is fixed, $q_{div}$ quickly converges to $q_u$. Convergence is not complete due to context variations: like hybrid recommendation approaches [32], A-BI$^+$ merges the user's interests stored in the log with the a-priori knowledge stored in the enriched image. If, for the very same context, the user requires slightly different queries across different visits, convergence is limited to the query fragments that are permanently required. In all cases, A-BI$^+$ performs better than A-BI.

To better emphasize how the hybrid nature of A-BI$^+$ impacts effectiveness, in Figure 10 we compare it to the two baselines given by its pure collaborative filtering behavior on the one hand, by its pure context-based behavior on the other. The first baseline, named *Coll*, returns the query that was chosen in the past from the most similar context; the second one, *Ctx*, returns the maximal query. Overall, *Coll* achieves worse performances than A-BI$^+$ since (i) no query is returned for $s = 0$ (i.e., $sim() = 0$), (ii) it completely ignores the currently sensed entities as it only contains entities sensed in the past, and (iii) if the user picks different queries in similar contexts, collaborative filtering initially oscillates between different queries. Conversely, A-BI$^+$ outperforms *Ctx* since the latter cannot keep into account the fragments that are actually chosen by

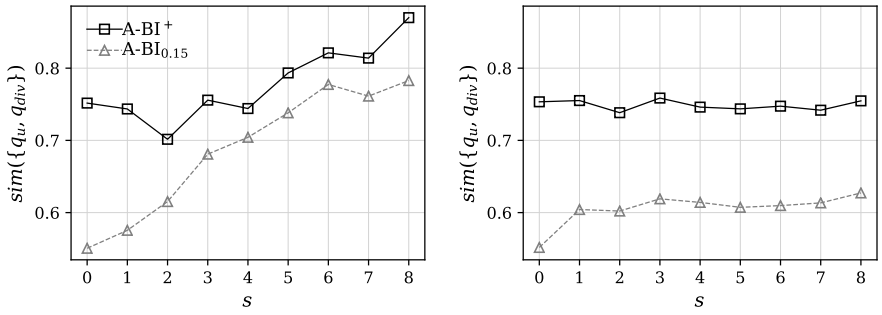


Figure 9: Average similarity between $q_{div}$ and $q_u$ for repeated visits when the context slightly changes (left) and $q_u$ slightly changes (right) ($\delta = 0.7$, $rq = 1$, $|T| = 12$)

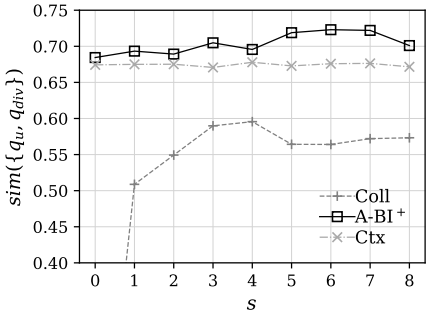


Figure 10: Average similarity between $q_{div}$ and $q_u$ for repeated visits when both the context and $q_u$ slightly changes ($rq = 2$, $|T| = 12$)

the user even if they are not coded by mappings. Finally, *Coll* is worse than *Ctx* since its recommendations sum up two errors: the exclusion of currently sensed entities, and the inclusion of entities sensed in the past that are included in the target query.

### *6.2. Efficiency*

We ran the tests on a machine equipped with Intel(R) Core(TM) i7-6700 CPU @ 3.40GHz CPU and 8GB RAM, with the A-BI$^+$ framework implemented in Scala; the log is stored in main memory. Figure 11 (left) shows the *total* time required to recommend increasing numbers of queries. Remarkably, the order of magnitude is $10^{-1}$ seconds. Besides, although the execution time slightly increases with $rq$, the time needed to return the first recommendation is fixed as Algorithm 3 never drops a query once it has been added to the result set $R$. Figure 11 (right) shows the increase in execution time for larger contexts (the higher the number of context entries, the higher the number of mappings). Query generation (which encompasses Algorithms 1 and 2) accounts for most of

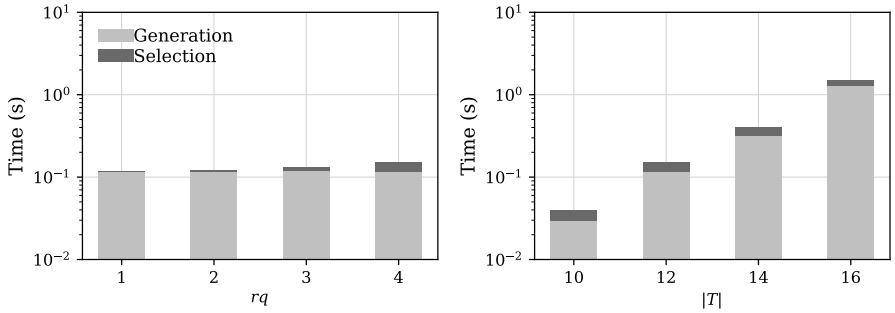


Figure 11: Efficiency of A-BI$^+$ for increasing values of $rq$ and $|T|$ ($s = 0$, $\delta = 0.7$; where not specified, $rq = 4$ and $|T| = 12$)

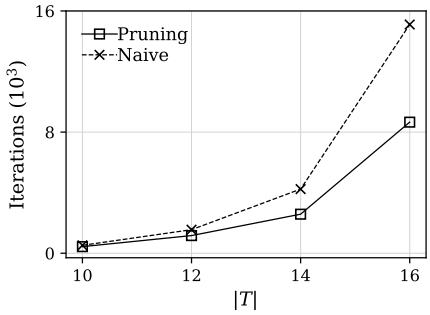


Figure 12: Effects of pruning in Algorithm 2 (Line 2) as the number of algorithm iterations

the time; indeed, the generative approach is computationally heavy, requiring to find, inside the fragment space, a potentially exponential number of relevant queries. Pruning in Algorithm 2 (Line 2) helps in constraining the generation search space. Figure 12 shows how pruning decreases the (exponential) number of generated queries for different context cardinalities. Remarkably, when large contexts are considered it is possible to cut down generation times by constraining the number of generated queries.

We finally emphasize that the execution time corresponds to the time necessary to generate the recommended queries, and not to the time to actually execute them. Queries are executed against the enterprise data mart and their performance clearly depends on the underlying multidimensional engine.

### *6.3. User evaluation*

There is no point in recommending a set of queries if the relevance estimated by the recommender significantly differs from the user's one. To assess how close the user's relevance is to the one of A-BI$^+$ or, in other words, to assess the recommendation quality, we conducted a set of tests with 30 users, mainly

master students in data science with basic or advanced knowledge of business intelligence and data warehousing. The evaluation is based on a real-world factory environment provided by Technogym, a large Italian company producing gym equipment. After a 10 minute introduction to A-BI$^+$, we simulated two user sessions in which the user enters two rooms for the first time (i.e., the log is not considered). For each room, the user was asked to impersonate a production controller and to suggest a GPSJ query $q_u$ that could help her in carrying out an assigned task based on a given context. To avoid biases, the assigned tasks were generic, meaning that there is not a single query that obviously fulfills the task, so the suggested queries depend on the personal interpretation of the task. In each room, once the user has provided her query, three queries recommended by A-BI$^+$ were presented to her. Finally, the user was asked to provide (1) the *perceived* similarity of each recommended query to the query she suggested, and (2) a score (on a scale from 1 to 10) indicating how the recommended queries are deemed to be relevant to the context and to the proposed task. The first question enables the evaluation of how the similarity adopted in A-BI$^+$ is perceived by the users independently of the relevance of the recommended queries to the context. Conversely, the second question is aimed at understanding the perceived relevance of the recommended queries to the context/task.

**Example 13 (Room visit).** *With reference to the* Assembly *cube and to the context represented in Figure 1*

$$T = \{\langle\{\mathsf{Object}, \mathrm{BikeExcite}\}\rangle, \langle\{\mathsf{Object}, \mathrm{Seat}\}\rangle, \langle\{\mathsf{Date}, 20/05/2019\}\rangle, \langle\{\mathsf{Role}, \mathrm{Controller}\}\rangle\}$$

*the assigned task is: "Analyze the assembly speed with reference to the context". Examples of queries recommended by A-BI$^+$ are*

$$\begin{aligned}
q &= \langle\{\mathsf{Year}, \mathsf{Part}, \mathsf{Product}\},\\
&\quad \{(\mathsf{Year} = 2019)\ AND\ (\mathsf{Part} = \mathrm{Seat})\ AND\ (\mathsf{Product} = \mathrm{BikeExcite})\},\\
&\quad \{\mathsf{AssembledItems}, \mathsf{AssemblyTime}\}\rangle\\
q' &= \langle\{\mathsf{Year}, \mathsf{Part}, \mathsf{Category}\}\\
&\quad \{(\mathsf{Year} = 2019)\ AND\ (\mathsf{Part} = \mathrm{Seat})\ AND\ (\mathsf{Category} = \mathrm{Sport})\},\\
&\quad \{\mathsf{AssembledItems}, \mathsf{AssemblyTime}\}\rangle\\
q'' &= \langle\{\mathsf{Date}, \mathsf{Part}, \mathsf{Category}\},\\
&\quad \{(\mathsf{Date} = 20/05/2019)\ AND\ (\mathsf{Part} = \mathrm{Seat})\ AND\ (\mathsf{Category} = \mathrm{Sport})\},\\
&\quad \{\mathsf{AssembledItems}, \mathsf{AssemblyTime}\}\rangle
\end{aligned}$$

□

The results are summarized in Table 4. The average perceived similarity between the user query $q_u$ and the three queries recommended by A-BI$^+$ is $0.58 \pm 0.20$ for Room 1 and $0.57 \pm 0.20$ for Room 2, which is very close to the one computed through $sim()$ ($0.59 \pm 0.15$ and $0.61 \pm 0.15$, respectively). Having

| | **Room 1** | **Room 2** |
|---|---|---|
| Avg. $sim()$ | $0.59 \pm 0.15$ | $0.61 \pm 0.15$ |
| Avg. perceived similarity | $0.58 \pm 0.2$ | $0.57 \pm 0.2$ |
| Pearson correlation | 0.41 | 0.41 |
| $q_{max}$ matches | 62% | 69% |
| $q_{div}$ matches | 77% | 88% |

Table 4: Results of user evaluation

near values is not enough to certify the coherence between the two similarities, thus we also computed their Pearson correlation coefficient obtaining an overall correlation of 0.41, which further supports this coherence.

As to the perceived relevance of recommended queries to the assigned tasks, the users evaluated the relevance of the recommended query set as 7.85 for Room 1 and 7.62 for Room 2, proving their satisfaction towards the recommendation. We finally emphasize that, without diversification ($rq = 1$), A-BI$^+$ would return only the most relevant query, $q_{max}$, which turned out to be the most similar one to $q_u$ in 62% of cases for Room 1 and 69% of cases for Room 2. When diversification is taken into account ($rq = 3$), these percentages increase to 77% for Room 1 and 88% for Room 2.

## 7. Conclusion

The A-BI$^+$ framework is a first result in the direction of establishing a tight connection between analytical reporting and AR applications. Besides proposing a reference functional architecture and an interaction process, in this paper we have shown that query recommendations can be given in real-time, highlighting the role of diversification and collaborative filtering in improving their effectiveness. Noticeably, our framework could be easily generalized to operate in other contexts, e.g., to recommend analytical queries concerning nearby objects based on the recognition of RFID tags.

A-BI$^+$ can be improved along different directions. First of all, it would be interesting to investigate how A-BI$^+$ could be turned into a purely statistical framework where all weights are expressed in terms of probabilities and reasoning is probabilistic as well; in this case, the log could be used to directly update mapping weights. Another possibility is to extend our model of context to a graph, so as to base recommendations on separate groups of entries (e.g., to distinguish foreground from background objects and to make mappings role-aware); this could be particularly relevant to take egocentric computer vision and engagement into account [8]. Also the execution performances of recommended queries deserve further attention. Some possible enhancements here would be (i) to add a criterion for query selection that also considers an estimate of the query performance and (ii) to give users, for each recommended query, an estimate of its execution time plus a quick preview of its its results; note that the latter point would raise some interesting possibilities for multiquery optimiza-

tion and caching. Finally, recommendation could also be extended from plain OLAP queries to complex analytics, e.g., anomaly detection: some event that is not in line with historical trends is going on, so it should be singled out.

In a broader perspective, it would be interesting to correlate context-awareness to data quality issues. In fact, it has been recognized that contextual assessments can be as important as objective quality indicators because they affect which information gets used for decision making tasks [51]. Specifically, the data quality dimensions impacted by contextual data are relevancy, value-added, timeliness, completeness, and amount of data [52].

## References


[1] T. Dwyer, N. H. Riche, K. Klein, W. Stuerzlinger, B. H. Thomas, Immersive analytics, Dagstuhl Reports 6 (2016) 1–9.

[2] R. Azuma, A survey of augmented reality, Presence 6 (1997) 355–385.

[3] A. Croatti, A. Ricci, Towards the web of augmented things, in: Proceedings of ICSA, Gothenburg, Sweden, 2017, pp. 80–87.

[4] G. Adomavicius, R. Sankaranarayanan, S. Sen, A. Tuzhilin, Incorporating contextual information in recommender systems using a multidimensional approach, ACM Trans. Inf. Syst. 23 (2005) 103–145.

[5] K. Stefanidis, E. Pitoura, P. Vassiliadis, A context-aware preference database system, Int. J. Pervasive Computing and Communications 3 (2007) 439–460.

[6] I. A. Ibrahim, A. M. Albarrak, X. Li, Constrained recommendations for query visualizations, Knowl. Inf. Syst. 51 (2017) 499–529.

[7] A. Bulling, O. Cakmakci, K. Kunze, J. M. Rehg, Eyewear computing - augmenting the human with head-mounted wearable assistants, Dagstuhl Reports 6 (2016) 160–206.

[8] Y. Su, K. Grauman, Detecting engagement in egocentric video, in: Proceedings of ECCV, 2016, pp. 454–471.

[9] N. A. M. El-Sayed, B. H. Thomas, K. Marriott, J. Piantadosi, R. T. Smith, Situated analytics: Demonstrating immersive analytical tools with augmented reality, J. Vis. Lang. Comput. 36 (2016) 13–23.

[10] W. Büschel, A. Mitschick, R. Dachselt, Here and now: Reality-based information retrieval: Perspective paper, in: Proceedings of CHIIR, New Brunswick, USA, 2018, pp. 171–180.

[11] N. Hube, M. Müller, The data in your hands: Exploring novel interaction techniques and data visualization approaches for immersive data analytics, in: Proceedings of VisBIA, Castiglione della Pescaia, Italy, 2018, pp. 12–21.

[12] T. Czauderna, J. Haga, J. Kim, M. Klapperstück, K. Klein, T. W. Kuhlen, S. Oeltze-Jafra, B. Sommer, F. Schreiber, Immersive analytics applications in life and health sciences, in: K. Marriott, et al. (Eds.), Immersive Analytics, volume 11190 of *LNCS*, Springer, 2018, pp. 289–330.

[13] F. Tao, J. Cheng, Q. Qi, M. Zhang, H. Zhang, F. Sui, Digital twin-driven product design, manufacturing and service with big data, The International Journal of Advanced Manufacturing Technology 94 (2018) 3563–3576.

[14] T. Chandler, T. Morgan, T. W. Kuhlen, Exploring immersive analytics for built environments, in: K. Marriott, et al. (Eds.), Immersive Analytics, volume 11190 of *LNCS*, Springer, 2018, pp. 331–357.

[15] M. Francia, M. Golfarelli, S. Rizzi, Augmented business intelligence, in: Proceedings of International Workshop on Design, Optimization, Languages and Analytical Processing of Big Data, Lisbon, Portugal, 2019.

[16] H. Jerbi, F. Ravat, O. Teste, G. Zurfluh, Preference-based recommendations for OLAP analysis, in: Proceedings of DaWaK, Linz, Austria, 2009, pp. 467–478.

[17] J. Aligon, E. Gallinucci, M. Golfarelli, P. Marcel, S. Rizzi, A collaborative filtering approach for recommending OLAP sessions, Decision Support Systems 69 (2015) 20–30.

[18] N. Khoussainova, Y. Kwon, M. Balazinska, D. Suciu, Snipsuggest: Context-aware autocompletion for SQL, PVLDB 4 (2010) 22–33.

[19] R. Thollot, N. Kuchmann-Beauger, M. Aufaure, Semantics and usage statistics for multi-dimensional query expansion, in: Proceedings of DASFAA, Busan, South Korea, 2012, pp. 250–260.

[20] R. B. Llavori, V. Nebot, Context-Aware Business Intelligence, 2015, pp. 87–110.

[21] M. Vartak, S. Rahman, S. Madden, A. G. Parameswaran, N. Polyzotis, SEEDB: efficient data-driven visualization recommendations to support visual analytics, PVLDB 8 (2015) 2182–2193.

[22] K. Dimitriadou, O. Papaemmanouil, Y. Diao, AIDE: an active learning-based approach for interactive data exploration, IEEE Trans. Knowl. Data Eng. 28 (2016) 2842–2856.

[23] M. L. Guilly, J. Petit, V. Scuturici, SQL query completion for data exploration, CoRR abs/1802.02872 (2018).

[24] J. Cumin, J. Petit, V. Scuturici, S. Surdu, Data exploration with SQL using machine learning techniques, in: Proce. EDBT, Venice, Italy, 2017, pp. 96–107.

[25] X. Yan, J. Guo, X. Cheng, Context-aware query recommendation by learning high-order relation in query logs, in: Proceedings of CIKM, Glasgow, United Kingdom, 2011, pp. 2073–2076.

[26] D. Jiang, K. W. Leung, J. Vosecky, W. Ng, Personalized query suggestion with diversity awareness, in: Proceedings of ICDE, Chicago, USA, 2014, pp. 400–411.

[27] S. Qi, D. Wu, N. Mamoulis, Location aware keyword query suggestion based on document proximity, IEEE Trans. Knowl. Data Eng. 28 (2016) 82–97.

[28] M. Chen, J. Sun, X. Ni, Y. Chen, Improving context-aware query classification via adaptive self-training, in: Proce. CIKM, Glasgow, United Kingdom, 2011, pp. 115–124.

[29] A. Giacometti, P. Marcel, E. Negre, A. Soulet, Query recommendations for OLAP discovery-driven analysis, IJDWM 7 (2011) 1–25.

[30] K. Xu, M. Zhu, D. Zhang, T. Gu, Context-aware content filtering & presentation for pervasive & mobile information systems, in: Proceedings of ICST, Quebec, Canada, 2008, p. 20.

[31] W. Xue, H. K. Pung, P. P. Palmes, T. Gu, Schema matching for context-aware computing, in: Proceedings of UbiComp, Seoul, Korea, 2008, pp. 292–301.

[32] P. Marcel, E. Negre, A survey of query recommendation techniques for data warehouse exploration, in: Proceedings of EDA, Clermont-Ferrand, France, 2011, pp. 119–134.

[33] K. Zheng, H. Wang, Z. Qi, J. Li, H. Gao, A survey of query result diversification, Knowl. Inf. Syst. 51 (2017) 1–36.

[34] E. Negre, F. Ravat, O. Teste, R. Tournier, Cold-start recommender system problem within a multidimensional data warehouse, in: Proceedings of RCIS, 2013, pp. 1–8.

[35] A. Ajanki, M. Billinghurst, T. Järvenpää, M. Kandemir, S. Kaski, M. Koskela, M. Kurimo, J. Laaksonen, K. Puolamäki, T. Ruokolainen, T. Tossavainen, Contextual information access with augmented reality, in: Proceedings of Int. Work. on Machine Learning for Signal Processing, 2010, pp. 95–100.

[36] R. Sicat, J. Li, J. Choi, M. Cordeil, W. Jeong, B. Bach, H. Pfister, DXR: a toolkit for building immersive data visualizations, IEEE Trans. Vis. Comput. Graph. 25 (2019) 715–725.

[37] B. Shneiderman, The eyes have it: A task by data type taxonomy for information visualizations, in: Proceedings of IEEE Symposium on Visual Languages, Boulder, USA, 1996, pp. 336–343.

[38] A. Gupta, V. Harinarayan, D. Quass, Aggregate-query processing in data warehousing environments, in: Proceedings of VLDB, 1995, pp. 358–369.

[39] S. Rizzi, E. Saltarelli, View materialization vs. indexing: Balancing space constraints in data warehouse design, in: Proceedings of CAiSE, 2003, pp. 502–519.

[40] V. Lomonaco, D. Maltoni, L. Pellegrini, Fine-grained continual learning, CoRR abs/1907.03799 (2019).

[41] J. Aligon, P. Marcel, E. Negre, Summarizing and querying logs of OLAP queries, in: F. Guillet, B. Pinaud, G. Venturini, D. A. Zighed (Eds.), Advances in Knowledge Discovery and Management, volume 3, 2011, pp. 99–124.

[42] P. G. Selinger, M. M. Astrahan, D. D. Chamberlin, R. A. Lorie, T. G. Price, Access path selection in a relational database management system, in: Proceedings of SIGMOD, Boston, USA, 1979, pp. 23–34.

[43] M. Golfarelli, E. Saltarelli, The workload you have, the workload you would like, in: Proceedings of DOLAP, New Orleans, USA, 2003, pp. 79–85.

[44] V. Poosala, Y. E. Ioannidis, Selectivity estimation without the attribute value independence assumption, in: Proceedings of VLDB, Athens, Greece, 1997, pp. 486–495.

[45] D. Gunopulos, G. Kollios, V. J. Tsotras, C. Domeniconi, Selectivity estimators for multidimensional range queries over real attributes, VLDB J. 14 (2005) 137–154.

[46] F. Pan, G. Cong, A. K. H. Tung, J. Yang, M. J. Zaki, Carpenter: finding closed patterns in long biological datasets, in: Proceedings of SIGKDD, Washington DC, USA, 2003, pp. 637–642.

[47] J. Aligon, M. Golfarelli, P. Marcel, S. Rizzi, E. Turricchia, Similarity measures for OLAP sessions, Knowl. Inf. Syst. 39 (2014) 463–489.

[48] F. Roberts, B. Tesman, Applied combinatorics, Chapman and Hall/CRC, 2009.

[49] D. S. Hochbaum, Approximating covering and packing problems: Set cover, vertex cover, independent set, and related problems, in: D. S. Hochbaum (Ed.), Approximation Algorithms for NP-hard Problems, PWS Publishing Co., Boston, MA, USA, 1997, pp. 94–143.

[50] M. R. Vieira, H. L. Razente, M. C. N. Barioni, M. Hadjieleftheriou, D. Srivastava, C. T. Jr., V. J. Tsotras, On query result diversification, in: Proceedings of ICDE, Hannover, Germany, 2011, pp. 1163–1174.

[51] S. Watts, G. Shankaranarayanan, A. Even, Data quality assessment in context: A cognitive perspective, Decision Support Systems 48 (2009) 202–211.

[52] D. M. Strong, Y. W. Lee, R. Y. Wang, Data quality in context, Commun. ACM 40 (1997) 103–110.