# Fuzzy Query By Example

Aurélien Moreau, Olivier Pivert, Grégory Smits

# Fuzzy Query By Example

Aurélien Moreau, Olivier Pivert and Grégory Smits
Univ Rennes, CNRS, IRISA – UMR 6074
22305 Lannion Cedex France
{aurelien.moreau | olivier.pivert | gregory.smits}@irisa.fr

## ABSTRACT

This paper describes Fuzzy Query By Example, an approach helping users retrieve data without any prior knowledge of the database schema or any formal querying language. The user is solicited to evaluate, in a binary way, pre-selected items of the database. We provide a characterization-based strategy that identifies the properties shared by the examples (resp. counter-examples) positively (resp. negatively) evaluated by the user. These properties are expressed using linguistic terms from a fuzzy vocabulary to ensure that the user has a good understanding of the inferred query.

## CCS Concepts

**•Information systems → Users and interactive retrieval;**

## Keywords

Query by Example; Fuzzy logic; Databases;

## 1. INTRODUCTION

Storing and structuring information efficiently are some of the keys to facilitating data browsing and retrieval. Commercial websites provide easy-to-use interfaces to navigate around their data, *e.g.* with keyword search. Other search paradigms — such as faceted search or attribute filtering — were also introduced to ease date browsing. In this paper we consider the Query By Example (QBE) paradigm, in which users are asked to evaluate samples from the database to find items similar to the ones they evaluate positively — *positive examples* — while dismissing those evaluated negatively — *counter-examples*. QBE implies two main challenges: (i) the selection of examples to submit for user evaluation, and (ii) the interpretation of these user evaluations to infer a query retrieving relevant results.

We present Fuzzy Query By Example (FQBE), a QBE approach based on fuzzy set theory. Fuzzy logic provides tools to express and infer preferences in a flexible way. Our objective is to help users obtain answers with a simple binary evaluation of examples. As in [13], we consider a fuzzy vocabulary for each attribute domain in the database. This vocabulary enables us to formulate, in a linguistic way, descriptions of the attribute values shared by positive examples that are not shared by counter-examples (later defined as *characterizations*).

Let us consider that the user is given to evaluate the examples in Table 1. Based on the obtained evaluations, with FQBE we infer that the user is interested in cars that have: *small* engine size (eng.), *medium* price, and *very low, low or medium* fuel consumption (mostly low) (FC). In addition to returning the items that best match the examples evaluated by the user, we also provide the user with an interpretable linguistic explanation of the fuzzy query inferred by the system. In other words, the user *knows* what the system *believes* about his/her preferences. We implemented FQBE and conducted experiments to demonstrate the interest of our approach. In this paper we show how to:

1. Select which examples in a dataset should be submitted to the user for evaluation, leveraging a vocabulary;
2. Use the evaluations to deduce the data properties the user is interested in;
3. Translate these properties into a query;
4. Provide the user with understandable explanations.

The remainder of the paper is structured as follows. Related work is discussed in Section 2. Section 3 provides a refresher on fuzzy set theory. In Section 4, we describe the principle of the approach and present experimental results in Section 5. Finally, Section 6 recalls the main contributions and outlines perspectives for future work.

## 2. RELATED WORK

The elicitation of user preferences stems from the domain of *preference learning* [3]. Preference learning techniques aim at inferring a preference model (utility function or binary relation) from examples or user feedbacks. In this approach we do not aim at learning a preference model but descriptions of a set of examples. However we do rank the examples



*SAC 2018,April 9–13, 2018, Pau, France*

ACM 978-1-4503-5191-1/18/04…$15.00

DOI: https://doi.org/10.1145/3167132.3167208

**Table 1: Some attribute values of the example used in this paper**

| year | FC | mileage | eng. | HP | price | make | eval |
|---|---|---|---|---|---|---|---|
| 2009 | 6 | 32500 | 1.4 | 85 | 9900 | seat | + |
| 2009 | 6 | 59000 | 1.4 | 85 | 8900 | seat | + |
| 2008 | 10 | 25000 | 2 | 136 | 20900 | volvo | - |
| 2008 | 10 | 50000 | 2 | 136 | 20900 | volvo | - |
| 2009 | 8 | 45000 | 1.4 | 85 | 9900 | seat | + |
| 2008 | 5 | 32500 | 1.4 | 70 | 10000 | mini | + |
| 2008 | 9 | 35000 | 2 | 140 | 27000 | ford | - |
| 2009 | 6 | 12000 | 1.6 | 115 | 16000 | chevrolet | - |
| 2009 | 8 | 6000 | 1.6 | 115 | 16000 | chevrolet | - |
| 2005 | 12 | 39000 | 2 | 140 | 15500 | audi | - |
| 2010 | 9 | 4000 | 1.9 | 105 | 20500 | seat | - |
| 2008 | 9 | 60000 | 2 | 140 | 23000 | audi | - |
| 2009 | 7 | 17000 | 2 | 136 | 31500 | peugeot | - |

provided to the user to prepare the preference learning step.

The selection of examples representative of a dataset can be viewed as a machine learning task similar to *boosting* algorithms [6]. Starting from one "good" example, the search for new examples is a compromise between *exploitation* — ensuring obtaining other "good" examples with similar values — and *exploration* — aiming at obtaining more diverse examples, albeit not so "good." When selecting examples we aim at balancing exploitation and exploration.

Query by Example is a paradigm in information retrieval to acquire results based either on:

- one (or several) input tuple(s) provided by the user;
- or the evaluation of prototypical examples (positively, negatively, ...) reflecting the content of the database.

The expected output contains elements that are similar to the input tuple(s) provided as example(s), or that reflect the choices of the user if prototypical examples were evaluated. For example, if a user browses houses and positively evaluates houses with 3 bedrooms and negatively evaluates houses with a small garden, then the results will include houses with 3 bedrooms and a "not small" garden.

Some user-example-based approaches have been proposed for graph databases, where users can either submit a graph [9] or simply tuples [5]. Only providing tuples enables the user to not have to specify the relation between the instances in the tuples. For example, if a user inputs the tuple ⟨Jerry Yang, Yahoo!⟩, then answers such as ⟨David Filo, Yahoo!⟩ and ⟨Sergei Brin, Google⟩ are returned, without ever providing the relation (in this case, company founder) between the elements in the tuple given as input.

Unlike most papers cited above, our work focuses on an evaluation-based QBE approach. We do not need users to have in mind the exact idea of what they are looking for; instead we ask them to evaluate (positively or negatively) examples of our choosing. We highlight the differences between the works of [1], [13] and this paper on three points: the selection of examples, their evaluation, and the use of these evaluations.

## 2.1 Selection of Prototypical Examples

The authors of [1] suggest using: (i) actual items that are representative of categories, or (ii) items that feature diversified values for the attributes the user is interested in, or (iii) items generated by the user. The authors of [13] propose to resort to a random or partially random selection of examples, either using some information on the user if available, or representing predefined categories of data. Also, they take into account previous evaluations to select new examples with a partitioning method. In both [1] and [13] the authors do not specify how to obtain examples representative of (categories of) the dataset. In this work we propose to use the vocabulary describing the data as well as the representativity of the data to select examples to evaluate.

## 2.2 Evaluation of Prototypical Examples

The authors of [1] propose to assign an importance to attributes, and to evaluate the representativity of an example on a scale. The authors of [13] propose to evaluate each example value for each attribute on a scale, as well as to give a global evaluation to each example. In this work we ask users to simply give a binary global evaluation to each example: no attribute preferences, nor any single attribute value preferences. Thus we do not require users to have an understanding of the structure of the database, and try to keep the number of interactions with the system reasonable.

## 2.3 Evaluations, Results, and User Preferences

The authors of [1] check whether items in the database are "similar to at least one example (w.r.t. all the attributes) and which are dissimilar to all counter-examples (each time w.r.t. at least one attribute)." They do not infer user preferences — with linguistic labels — but provide items that match the user's preferences, according to a measure of their own. For each database item $x$, this measure is computed with (i) the similarity degree between positive examples and $x$, and (ii) the dissimilarity degree between counter-examples and $x$, on all attribute domains. The authors of [13] provide items to evaluate until the user is satisfied, by selecting new items to evaluate with the k-NN algorithm. The discovery of user preferences — to obtain new examples — is done in an "extensional way" through the selection of desirable items. The reconstruction of user preferences — to keep the user aware of the preference elicitation process — however is done in an "intensional way" by describing the liked items. Global positive and negative evaluations are used to find association rules determining whether some linguistic terms are relevant to the user. Then for each attribute, at best one linguistic term is selected to express the user preferences for this attribute, only if the support, confidence and lift for the association rule that found it are "high enough." They generate a fuzzy query based on user preferences reconstructed but never run it: they only use it for explanation purposes. In this work we look for common *properties* between positive examples on the one hand, and common *properties* between counter-examples on the other hand to infer user preferences and generate a query matching those. Unlike the approach described in [1], we do not use similarity relations to infer user preferences. Also, we use the inferred preferences to compute a fuzzy query and present its results, unlike the approach in [13] which resorts to an iterative evaluation of examples until the user is satisfied (without ever evaluating a fuzzy query).

# 3. LINGUISTIC VOCABULARY AND SET OF REPRESENTATIVE ITEMS

Before detailing FQBE we first introduce some notions such as fuzzy vocabularies, fuzzy queries, and also we provide a method to build a set of diverse items representing the diversity of the terms involved in the vocabulary.

## 3.1 Fuzzy Vocabulary and Data Rewriting

Fuzzy set theory was introduced by Zadeh [12] for modeling classes or sets whose boundaries are not clear-cut. For such objects, the transition between full membership and full mismatch is gradual rather than crisp. Typical examples of such fuzzy classes are those described using adjectives of the natural language, such as *young, cheap, fast*, etc. Formally, a fuzzy set $F$ on a referential $U$ is characterized by a membership function $\mu_F : U \to [0, 1]$ where $\mu_F(u)$ denotes the grade of membership of $u$ in $F$. In particular, $\mu_F(u) = 1$ reflects full membership of $u$ in $F$, while $\mu_F(u) = 0$ expresses absolute non-membership. When $0 < \mu_F(u) < 1$, one speaks of partial membership.

Let $R$ be a relation defined on a set $\mathcal{A}$ of $q$ categorical or numerical attributes $\{A_1, A_2, \ldots, A_q\}$. A fuzzy vocabulary, denoted by $\mathcal{V}$, on $R$ is defined by means of fuzzy partitions of the $q$ domains. A fuzzy partition $\mathcal{P}_i$ associated with the domain $\mathcal{D}_i$ of attribute $A_i$ is composed of $m_i$ fuzzy sets $\{P_{i,1}, P_{i,2}, \ldots, P_{i,m_i}\}$, such that for all $x \in \mathcal{D}_i$:

$$\sum_{j=1}^{m_i} \mu_{P_{i,j}}(x) = 1,$$

where $\mu_{P_{i,j}}(x)$ denotes the degree of membership of $x$ to the fuzzy set $P_{i,j}$. Each fuzzy partition $\mathcal{P}_i$ is associated with a set of linguistic labels $\{L_1^i, L_2^i, \ldots, L_{m_i}^i\}$.

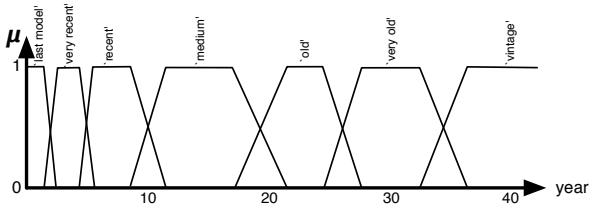


**Figure 1: A partition over the domain of the attribute *year***

For the case of numerical attributes (Fig. 1 illustrates a possible definition of a partition over the attribute *year* of second-hand cars), it is also imposed, for the sake of interpretability, that a fuzzy set $P_i$ in a partition $\mathcal{P}_i$ can only overlap with its predecessor $P_{i-1}$ or/and its successor $P_{i+1}$ (when they exist). In this paper we only consider numerical attributes. However FQBE is also applicable to categorical attributes: we only impose that for each attribute domain the sum of the membership degrees on all modalities of a partition be 1.

In the approach we propose, it is assumed that a vocabulary has already been manually defined on the data considered by a domain expert using a dedicated graphical interface such as ReqFlex [11], or using an automatic method of vocabulary elicitation from data [7, 4, 10].

Definition 1. *An item $x$ may be rewritten in terms of a vocabulary $\mathcal{V}$ as a vector of $\sum_{k=1}^{q} m_k$ membership degrees. This vector, denoted by $RV_x$, is of the form $\langle \mu_{P_{1,1}}(x.A_1), \ldots, \mu_{P_{1,m_1}}(x.A_1), \ldots, \mu_{P_{q,1}}(x.A_q), \ldots, \mu_{P_{q,m_q}}(x.A_q) \rangle$.*

Example 1. *The rewriting of a car from 2007 is rewritten according to the* year *partition illustrated by the Figure 1 into the following vector: $\langle 0, 0, 0.5, 0.5, 0, 0, 0 \rangle$, or, only using non-zero values: $\langle 0.5/recent, 0.5/medium \rangle$.*$\diamond$

## 3.2 Fuzzy Query

Terms from a user vocabulary may be used to form fuzzy predicates that can then take part in the selection statement of a query. Let $P_{i,j}$ be a fuzzy set from the partition $\mathcal{P}_i$ defined on the domain of attribute $A_i$ and $L_j^i$ be its associated linguistic label. Then, a fuzzy predicate based on $P_{i,j}$ is of the form $A_i$ *is* $L_j^i$. Such fuzzy predicates, as well as Boolean predicates, may be combined in a conjunctive or disjunctive way using respectively the t-norm ($\top$) and t-conorm ($\bot$) operators. Let $P_a$ and $P_b$ be two fuzzy sets, then their conjunctive (resp. disjunctive) combination is denoted by $P_a \wedge P_b$ (resp. $P_a \vee P_b$) and processed in the following way:

$$\mu_{P_a \wedge P_b}(x) = \top(\mu_{P_a}(x), \mu_{P_b}(x)) \quad (1)$$

(resp. $\mu_{P_a \vee P_b}(x) = \bot(\mu_{P_a}(x), \mu_{P_b}(x))$), where the min (resp. max) operator is generally used for $\top$ (resp. $\bot$).

In FQBE, we generate a conjunctive fuzzy query $Q'$ reflecting the properties that the desirable items should satisfy. Less restrictive aggregation functions, such as quantifiers or weighted sums, may be used instead of the conjunction. However, despite the fact that they may capture and discriminate more items, their meaning is more difficult to explain to the user (Sec. 4.4).

## 3.3 Diverse and Representative Items

The generated query $Q'$ is inferred from the binary evaluations of items provided by the user. The first question to address concerns the selection of items to submit to the user such that their evaluation, whether positive or negative, will make it possible to determine if a term of the vocabulary should appear in the inferred query.

As said in Sec. 2, we share the point of view of several existing approaches to QBE that it is more meaningful to evaluate real items from the DB instead of artificial ones. An optimal but unrealistic solution would be to find for each term of the vocabulary a set of items among which only one item fully characterizes the concerned term, all other values (i.e. rewritings) being equal. As it appears unconceivable to find such sets of items and to task the user to evaluate too many items, we propose a strategy that builds offline a reduced set $\mathcal{S}$ of $k$ items from the DB. These items are selected in such a way that they are representative of the vocabulary, of the dataset, and that they are as mutually diverse as possible.

The representativity of an item $x$ w.r.t. the vocabulary quantifies the extent to which the item may be precisely

described by the different linguistic terms. This representativity degree w.r.t. the vocabulary, denoted by $RepV(x)$ is computed from the rewriting vector $RV_x$ as follows:

$$RepV(x) = \sum_{A_i, i=1..q} \max_{P_{i,j}, j=1..m_i} \mu_{P_{i,j}}(x), \quad (2)$$

where $q$ is the number of dimensions over which each item is described. This representativity degree is thus maximal if $x$ fully satisfies one fuzzy set on each dimension (as opposed to somewhat satisfying more than one fuzzy set).

An item $x$ also has to be representative of a sufficiently large data subset. This second representativity degree, this time w.r.t. the dataset $\mathcal{D}$, is denoted by $RepD(x)$ and defined by:

$$RepD(x) = \frac{1}{|\mathcal{D}|} \times \sum_{x' \in \mathcal{D}} 1 - d(RV_{x'}, RV_x), \quad (3)$$

where $d(RV_{x'}, RV_x)$ is a distance measure between two rewriting vectors computed in the following way:

$$d(RV_{x'}, RV_x) = \frac{\sum_{\mathcal{P}_i \in \mathcal{V}} \sum_{P_{i,j} \in \mathcal{P}_i} |\mu_{P_{i,j}}(x) - \mu_{P_{i,j}}(x')|}{\sum_{k=1}^{q} m_k}, \quad (4)$$

where $\mathcal{P}_i$ is a partition of the vocabulary $\mathcal{V}$, and that $P_{i,j}$ is a modality of the partition $\mathcal{P}_i$. The first element of $\mathcal{S}$ is computed with:

$$Sc(x, \emptyset) = \top(RepV(x), RepD(x)). \quad (5)$$

Then, the set $\mathcal{S}$ to build has to be as diverse as possible so as to capture the different term combinations that may be used to retrieve items. The diversity of an item $x$, or more precisely of its rewriting vector $RV_x$, w.r.t. to the set $\mathcal{S}$ of previously selected items, denoted by $Div_{\mathcal{S}}(x)$, represents the extent to which $RV_x$ is disjoint from the rewriting vectors of the items in $\mathcal{S}$.

$$Div_{\mathcal{S}}(x) = \frac{1}{|\mathcal{S}|} \sum_{x' \in \mathcal{S}} \mu_{disjoint}(RV_x, RV_{x'}), \quad (6)$$

where $\mu_{disjoint}(RV_x, RV_{x'})$ quantifies how much the two rewriting vectors are disjoint:

$$\mu_{disjoint}(RV_x, RV_{x'}) = \frac{1}{q} \sum_{A_i} [1 - \perp_{P_{i,j}} \top(\mu_{P_{i,j}}(x), \mu_{P_{i,j}}(x'))]. \quad (7)$$

W.r.t. the current content of the set $\mathcal{S}$, a global score may be computed for each item $x \in \mathcal{D} \backslash \mathcal{S}$ based on the three previously defined notions:

$$Sc(x, \mathcal{S}) = \top(RepV(x), RepD(x), Div_{\mathcal{S}}(x)), \quad (8)$$

where the Lukasiewicz t-norm and t-conorm ($\top_{Luk}(x, y) = \max(0, x + y - 1)$ and $\perp_{Luk}(x, y) = \min(x + y, 1)$) have been used in our case to allow for some compensation between the combined degrees.

The algorithm (Algo. 1) used to build $\mathcal{S}$ consists in first identifying the item $x^1$ from $\mathcal{D}$ maximizing the score $Sc(x, \emptyset)$, an item is arbitrarily picked in case of tie, and then to incrementally complete this set with the next best item, and so on until the cardinality of $\mathcal{S}$ is equal to $k$. It is worth recalling that this costly algorithm, $k \times |\mathcal{D}|$ steps, that computes a locally optimal diversified set (depending on the choice of the first $x^1$), is performed offline and only once, unless significant changes on $\mathcal{D}$ have been done. The second question concerns the choice of the value for the parameter $k$. The main goal of the approach is to display a small set of items that can be quickly evaluated by the user. Thus, as no more than 20 or 30 items can be displayed simultaneously, we consider that setting $k$ to 100 is generally enough. Obviously, additional interesting items may be identified on-the-fly if $k$ items are not enough, at the cost of new scans of the data.

**Input**: data $\mathcal{D}$; $k$ the number of expected examples in $\mathcal{S}$
**Output**: set of diversified examples $\mathcal{S}$
**begin**
  $\mathcal{S} \leftarrow \emptyset$;
  $x^1 \leftarrow \arg\max_{x \in \mathcal{D}}(Sc(x, \emptyset))$;
  $\mathcal{S} \leftarrow \{x^1\}$;
  **while** $|\mathcal{S}| < k$ **do**
    $maxSc \leftarrow 0$;
    $x^1 \leftarrow NULL$;
    **foreach** $x \in \mathcal{D} \backslash \mathcal{S}$ **do**
      $tmpSc \leftarrow Sc(x, \mathcal{S})$;
      **if** $tmpSc > maxSc$ **then**
        $maxSc \leftarrow tmpSc$;
        $x^1 \leftarrow x$;
      **end**
    **end**
    $\mathcal{S} \leftarrow \{x^1\} \cup \mathcal{S}$;
  **end**
  **return** $\mathcal{S}$;
**end**

**Algorithm 1:** Construction of $\mathcal{S}$

# 4. FQBE

FQBE consists of the following steps:

- The user chooses which examples are "good" or "bad" according to his/her expectations. They are put in a positive and a negative set respectively.
- The sets are characterized as in [8] so as to discover the modalities that represent them best.
- The positive set provides a conjunctive selection statement reflecting fuzzy properties desired by the user.
- The answers to this query are submitted to the user, along with explanations of the user preferences in a linguistic interpretable form.

After each example evaluation, the inferred query is updated and displayed. If they are satisfactory, the generated query is executed. Otherwise, the user can evaluate more examples to modify the inferred characterizations.

## 4.1 Evaluating Examples

In the following we use a running example to illustrate the last three steps mentioned at the beginning of the section. For the sake of clarity we select our own examples and the associated evaluations. Table 1 contains the data elements used in this example with the associated (positive or negative) evaluations. Positive examples are put in the set $\mathcal{E}_+$ and counter-examples are put in the set $\mathcal{E}_-$. Items in this example are described according to the attributes: year, fuel

consumption, mileage, option level, comfort level, security level, engine size, horse power, price and make.

## 4.2 Characterization of the Sets of Examples

The objective is to deduce from the evaluations of the suggested examples the user's expectations. We aim at finding which properties are satisfied by items of $\mathcal{E}_+$ and not by those of $\mathcal{E}_-$ (and vice-versa). These properties are called *characterizations*.

DEFINITION 2. *A characterization $E_{\mathcal{E}}$ attached to a set $\mathcal{E}$ is a conjunction of couples (attribute, fuzzy set of labels) of the form*

$$E_{\mathcal{E}} = \{(A_i,\ F_i) \mid A_i \in \mathcal{A} \text{ and } F_i \text{ is a fuzzy set of linguistic labels from the partition of the domain of } A_i\}.$$

Items from the sets $\mathcal{E}_+$ and $\mathcal{E}_-$ are projected on the vocabulary in order to obtain a characterization of the sets using terms of the natural language, i.e. terms from the vocabulary. The projection of a set $\mathcal{E}$ on the partition of an attribute $A_i \in \mathcal{A}$ is represented by a fuzzy set of labels $F_i = \{L_j^i/\mu_{L_j^i}(\mathcal{E}) \mid L_j^i \in \mathcal{P}_i\}$ where

$$\mu_{L_j^i}(\mathcal{E}) = \frac{\sum_{x\in\mathcal{E}} \mu_{L_j^i}(x)}{|\mathcal{E}|}, \tag{9}$$

and $\mu_{L_j^i}(x)$ is the degree of membership of $x$ to $L_j^i$. It is assumed that the only labels that appear in $F_i$ are such that $\mu_{L_j^i}(\mathcal{E}) > 0$. The degree associated with each label is related to the number of points verifying it and to their membership degrees, hence making characterizations representative of each set. The projection of the sets $\mathcal{E}_+$ and $\mathcal{E}_-$ on the modalities of the vocabulary leads to a table as the one illustrated in Table 2.

**Table 2: Projection of the sets on the vocabulary**

| Set | year | mileage | price |
|---|---|---|---|
| $\mathcal{E}_+$ | v. recent (1) | low (1) | medium (1) |
| $\mathcal{E}_-$ | recent (0.11)<br>v. recent (0.78)<br>recent (0.11) | v. low (0.44)<br>low (0.56) | expensive (0.54)<br>v. expensive (0.46) |

Each combination of attributes is a candidate characterization. All such combinations are reviewed. In [8] the authors coined two properties that characterizations of interest must have: specificity and minimality. A high specificity degree ensures that the characterization indeed characterizes one set in particular and not the other. Minimality ensures that the characterization contains as little attributes as possible while having a high specificity degree. Let us consider a candidate characterization $CE_{\mathcal{E}+}$ for $\mathcal{E}_+$. To check whether it is specific, we consider its mirror $CE_{\mathcal{E}-}$ for $\mathcal{E}_-$ with the same attributes (we project the elements of $\mathcal{E}_-$ on the attributes of $CE_{\mathcal{E}+}$). The specificity degree $sp(CE_{\mathcal{E}+}, CE_{\mathcal{E}-})$ is computed by the mutually exclusive disjunction between the fuzzy sets:

$$sp(CE_{\mathcal{E}+}, CE_{\mathcal{E}-}) = \max_{A_i\in\mathcal{A}} (1 - \max_{L_j^i\in\mathcal{P}_i} \min(\mu_{L_j^i}(\mathcal{E}_+),\ \mu_{L_j^i}(\mathcal{E}_-))), \tag{10}$$

where $\mathcal{A}$ is the set of attributes in the candidate characterizations $CE_{\mathcal{E}+}$ and $CE_{\mathcal{E}-}$, $\mathcal{P}_i$ the partition of the attribute $A_i \in \mathcal{A}$ on the vocabulary, and $L_j^i \in \mathcal{P}_i$ the modalities covered by the characterizations.

REMARK 1. *If either $\mathcal{E}_+$ or $\mathcal{E}_-$ is empty then all candidate characterizations for the non-empty set are fully specific.*

The number of possible characterizations is exponential in the number of attributes. In order to avoid reviewing all of them, the authors of [8] found that the characterization containing all attributes has the maximum specificity degree possible, a degree in $[0, 1]$ not necessarily equal to 1. Our Apriori-like algorithm to find characterizations reviews them by size, from one-attribute characterizations to the $q$-attributes characterization. As we want to find the minimal and specific characterizations, we stop looking for longer characterizations as soon as a characterization with the maximum specificity degree is found. It means that the attributes not yet explored do not increase the overall specificity. It means that these attributes do not make it possible to discriminate between items from $\mathcal{E}_+$ and $\mathcal{E}_-$. The characterizations found are then sorted by decreasing specificity degree.

EXAMPLE 2. *Using the data in Table 2, we present some of the possible candidate characterizations for $\mathcal{E}_+$:*

- $CE_+^1$ ="year is very recent (1)"
- $CE_+^2$ ="mileage is low (1)"
- $CE_+^3$ ="price is medium (1)."

*Candidate characterizations for $\mathcal{E}_-$ include:*

- $CE_-^1$ ="year is recent (0.11) or very recent (0.78) or last model (0.11)"
- $CE_-^2$ ="mileage is very low (0.44) or low (0.56)"
- $CE_-^3$ ="price is expensive (0.54) or very exp. (0.46)."

*In this running example we consider candidate characterizations with only one attribute, and get:*

$$sp(CE_+^1,\ CE_-^1) = 1 - \max(\min(0, 0.11), \min(1, 0.78), \min(0, 0.11)) = 0.22$$

*We obtain $sp(CE_+^2,\ CE_-^2) = 0.44$ and $sp(CE_+^3,\ CE_-^3) = 1$. As the label* very recent *is present in both $CE_+^1$ and $CE_-^1$ with a high degree, their specificity degree is low. Among these candidate characterizations, $CE^3$ is the best choice according to the specificity degree. Since $CE^3$ has the maximum specificity degree, there is no need to look for "longer" characterizations with the attribute* price*: they will not be minimal and cannot be more specific.*

*After reviewing all possible candidate characterizations and checking whether they verify the minimality property, we order them by decreasing specificity degree and find the following characterizations for $\mathcal{E}_+$ (top-3):*

- $E_+^1$ ="engine size is small" *(specificity 1);*
- $E_+^2$ ="price is medium" *(specificity 1);*
- $E_+^3$ ="consumption is very low (0.13) or low (0.62) or medium (0.25)" *(specificity 0.78).*

*We find the following characterizations for $\mathcal{E}_-$ (top-3):*

- $E_-^1$ ="engine size is medium (0.28) or big (0.72)" *(specificity 1);*
- $E_-^2$ ="price is expensive (0.54) or very expensive (0.46)" *(specificity 1);*

- $E^3_-$ ="consumption is low (0.22) or medium (0.11) or high (0.67)" *(specificity 0.78).*$\diamond$

Only the characterizations with a high enough specificity degree (above a threshold $\lambda$) become a selection condition.

## 4.3 Generation of the Fuzzy Query

From the characterizations of $\mathcal{E}_+$ and those of $\mathcal{E}_-$ we seek to build a query that will look for elements similar (in terms of properties) to the positive examples. The characterizations are first "refined," so as to render them coherent. A label present in the positive and negative characterizations may or may not be removed depending on its membership degree. We propose to apply the following rule: *if the positive and the negative degrees are both superior or equal to 0.5, then they are both removed from the positive and negative characterizations. Otherwise, the label with the highest degree is kept.*

Characterizations with a specificity degree equal to 1 are not affected by this rule. Indeed, a specificity degree equal to 1 means that there are no labels shared between the positive and negative characterizations: their sets are fully disjoint. In our running example, $E^3_-$ is concerned, the *low* modality is removed (0.62 is higher than 0.22) and we get: $E^3_-$ ="*consumption is medium (0.11) or high (0.67)*". The *medium* modality is also removed (0.25 is higher than 0.11), and as a result: $E^3_-$ ="*consumption is high (0.67)*".

Finally we translate the characterizations into a conjunctive query, each characterization becoming a selection condition. The selection conditions are in Listing 1.

```
engine size is small
and price is medium
and consumption is (very low (0.13) or low (0.62)
    or medium (0.25))
```

**Listing 1: Conjunction of selection conditions**

For some attributes one label does not fully cover the sets of examples or counter-examples: this leads to a disjunction of labels (such as *consumption is very low or low or medium*). Each of these labels does not carry the same weight: for instance here the label *low* has the highest membership degree (0.62). We treat these degrees as importances that must be taken into account in the query. To differentiate labels we propose to use the weighted disjunction as proposed in [2]. This weighted disjunction enables us to take into account the weights (here the membership degrees) assigned to each modality in the characterization, so that we obtain:

$$\mu_{A_i}(t) = \max_{P_{i,j} \in \mathcal{P}_i} \min(w_j, \mu_{P_{i,j}}(t)) \tag{11}$$

where $w_j$ denotes the weight associated with the modality $P_{i,j}$ related to attribute $A_i$ in the characterization used to generate the query.

Example 3. *Let us consider the selection condition* "consumption is very low (0.13) or low (0.62) or medium (0.25)". *In order to use the weighted disjunction, the weights must first be normalized so that the maximum weight is equal to 1. The selection condition thus becomes* "consumption is very low (0.2) or low (1) or medium (0.4)". *When reviewing a tuple $t$ with a* fuel consumption *that is fully "low," we get:*

$$\begin{aligned}\mu_{\textit{fuel con.}}(t) &= \max(\min(w_{\textit{very low}}, \mu_{\textit{very low}}(t)), \min(w_{\textit{low}}, \\ &\qquad \mu_{\textit{low}}(t)), \min(w_{\textit{medium}}, \mu_{\textit{medium}}(t))) \\ &= \max(\min(0.2, 0), \min(1, 1), \min(0.4, 0)) = 1,\end{aligned}$$

*which is the maximum possible obtainable value.*$\diamond$

## 4.4 Query Results

Query results are presented to the user, along with characterizations. They enable the user to know what preferences were inferred from his/her evaluations. Negative explanations can greatly improve readability if for any given attribute the positive explanation is a disjunction of most modalities. In this case only providing one or two modalities as the negative explanation is more cooperative than providing a disjunction covering almost the entire partition. For instance, the explanation "*not high* consumption" is more intelligible than the disjunction in $E^3_+$. In our running example, the user gets the following explanations:
*We believe you have an interest in:*

- *small* engine size;
- *medium* price.
- *not high* consumption.

Several objectives must be met when formulating explanations. Contradictions must be avoided: an attribute's label should not appear in positive and negative characterizations. Translation rules handle this issue. Characterizations must be easy to read: negligible labels (with a very low membership degree) may be omitted, and important labels should be emphasized.

# 5. EXPERIMENTS AND DISCUSSION

We study the effectiveness of FQBE using a real dataset of second-hand cars scraped from LeBonCoin.fr of cardinality 10k+, with 9 different numerical attributes that each have a predefined vocabulary. We set two objectives: (i) use *precision* to compare our example selection scoring method $Sc$ to a random selection, and (ii) determine how well we infer user preferences, and the impact of the specificity threshold $\lambda$ on *precision* and *recall*.

To evaluate the proposed approach, we consider the set of queries in Listing 2. These queries have been selected for the diversity of the vocabulary elements involved in their selection statement. For each such query $Q$ that we try to infer, we evaluate examples based on their belonging to the result set $R$ of $Q$.

Each tested fuzzy query $Q$ yields a fuzzy result set $R$. The *core* of $R$ is denoted by $R_c$ and contains only the fully satisfactory results of $R$. Its support, denoted by $R_s$ contains all elements from $R$ with a non-zero degree. With FQBE, by evaluating examples we generate a query $Q'$ that yields a result set $R'$. To evaluate the interest of the generated query $Q'$, we compare its result set $R'$ to the original result set $R$. Obtaining the fuzzy set equality $R = R'$ is not our primary objective here. Our first objective is precision: finding only "good" results. Only afterwards we shall focus on recall: finding all "good" results. We compute the degree

of inclusion of $R'$ in $R$ with $\mu_{\subseteq}(R', R)$, in order to know how "good" the generated results are. This is a fuzzy interpretation of the *precision* in IR. Similarly, we compute the inclusion of $R$ in $R'$ with $\mu_{\subseteq}(R, R')$ to obtain the *recall*.

$$\text{Precision} = \mu_{\subseteq}(R', R) = \frac{\sum_{x \in R'} \min(\mu_R(x), \mu_{R'}(x))}{\sum_{x \in R'} \mu_{R'}(x)}. \quad (12)$$

```
Q1: price is medium and mileage is low
Q2: (price is expensive or v. expensive)
    and year is last model
Q3: (consumption is low or v. low) and mileage
    is medium and (year is recent or v. recent)
Q4: price is medium and consumption is low
    and mileage is v. low and year is recent
Q5: price is cheap and mileage is low
Q6: consumption is low and year is v. recent
Q7: mileage is v. low and horsepower is v. high
Q8: price is cheap and horsepower is v. high
Q9: mileage is medium and consumption is v. low
Q10: year is medium and price is cheap
    and mileage is medium
```

**Listing 2: Conjunctions of selection conditions**

## 5.1 Example Selection Methods Comparison

We compare two sets of examples: a random selection, and $\mathcal{S}$, the set of examples *selected* with *Sc*. In both cases we consider a set of 150 examples to evaluate (let us recall that $\mathcal{S}$ is ordered). For each query $Q$ considered, we positively evaluate the examples that match it (that are in $R$), and negatively evaluate the others. Characterizations found such that $sp \geqslant 0.7$ are kept. In Table 3 we specify the size of the result set $R$ (the cardinality $|R_s|$ of its support) of each query $Q$ considered, as well as the number of results $|R_c|$ fully satisfying $Q$. We present the number of consecutive examples to evaluate to attain certain precision values (0.5 and 0.9) between the set of generated results and that of original results. We only focus on precision ($R' \subseteq R$) as we aim to find original results only, not all original results.

In Table 3 there is only one line for *Sc* as the 0.9 precision value is always attained with the same number of examples necessary to attain the 0.5 precision value. Random scores are obtained by averaging their results over 10 runs, and using the number 151 when no minimal number of examples has been found to attain the expected precision values (*e.g.* if for $Q1$ the precision 0.9 is attained by evaluating 49 examples on the first run, and is never attained by evaluating all examples on the second run, the average over these two runs is $(49 + 151)/2 = 100$). Nevertheless, the average numbers over 10 selections of randomly-chosen examples are far greater than those of *Sc*. In Table 3 we also show the ratio between the minimal number of random examples and the minimal number of examples from $\mathcal{S}$ to attain a precision of 0.9. On average, this ratio is of 15.2, showing that $\mathcal{S}$ is more efficient at building $Q'$. Only $Q4$ is not inferred: R4 may be too small compared to the dataset.

## 5.2 Impact of specificity threshold $\lambda$

We now evaluate the precision and recall over examples selected with *Sc* only. In Table 4 we present the cardinalities of the generated sets of answers for the 10 inferred queries considered (which can be compared to the cardinalities of the original sets of answers from Table 3), as well as the precision and recall for three different specificity threshold values $\lambda$. We positively evaluate examples from $\mathcal{S}$ that are in $R$, and negatively evaluate the others. $\mathcal{S}(R)$ is the set of examples from $\mathcal{S}$ that are also in $R$.

With a specificity threshold of 0.7, the generated queries do not yield many fully satisfactory results. While the precision is excellent (maximal for all queries but $Q4$ and $Q8$), the recall is very low, as are the numbers of generated results. The generated queries all have a very high number of selection conditions (see Table 5), almost one for every attribute (out of 9).

With a specificity threshold of 0.75, we obtain the maximal precision for queries $Q1, Q2, Q3, Q5, Q6, Q9$, and $Q10$. In other words, all fully satisfactory results from $R1'$ also fully satisfy $R1$, and the top-42 (unordered) results between these two sets are the same. However we are still far from covering the whole $R1$, although it is already a good improvement from finding only 4 elements with $\lambda = 0.7$. On average, the generated queries cover half the attributes of the dataset.

With the specificity threshold $\lambda = 0.8$, we obtain the maximal precision for queries $Q1, Q2, Q3, Q5, Q9$, and $Q10$. In other words, all fully satisfactory results from $R1'$ also fully satisfy $R1$, and the top-685 (unordered) results between these two sets are the same. Furthermore, we obtain $R1_s = R1'_s$. Furthermore, the recall for $R1$ is also maximal: we have $R1 = R1'$. This is a major improvement in terms of result set size, although the equality was not found in as many cases as before. The queries obtained with $\lambda = 0.8$ are described in Listing 3. On average the queries only cover a few attributes of the dataset (between 2 and 4).

```
Q1‘ price is medium (1) and mileage is low (1)
Q2‘ year is last model (1) and (price is
    expensive (0.462) or v. expensive (1))
Q3‘ year is recent (1) and (price is medium (1)
    or expensive (0.25)) and mileage is medium (1)
    and (consumption is v. low (0.5) or low (1))
Q4‘ true
Q5‘ price is cheap (1) and mileage is low (1)
    and (year is medium (1) or recent (1))
Q6‘ (price is v. cheap (0.5) or medium (1)
    or expensive (0.5)) and consumption is low (1)
Q7‘ (price is v. expensive (1) or excessively
    expensive (1)) and (consumption is high (0.66)
    or v. high (1)) and mileage is v. low (1)
Q8‘ price is cheap (1) and (year is v. old (0.5)
    or old (0.5) or medium (1))
Q9‘ consumption is v. low (1) and mileage is
    medium (1) and (price is v. cheap (0.43)
    or cheap (1) or medium (0.48)) and
    (year is medium (1) or recent (0.4))
Q10‘ consumption is v. low (1) and price is
    cheap (1) and mileage is medium (1)
    and year is medium (1)
```

**Listing 3: Conjunctions of selection conditions obtained**

The differences between recall values lie on the specificity threshold. We propose to check how many characterizations are kept for each query and for different specificity thresholds in Table 5. With 0.7, a query is generated over almost all attributes, with 0.75 over half the attributes and with 0.8 over one third of the attributes. This may explain the earlier gaps between result set sizes in Table 4. A high specificity

**Table 3: Result sets sizes, and inclusion degrees with Random and Sc selection methods**

| | Q1 | Q2 | Q3 | Q4 | Q5 |
|---|---|---|---|---|---|
| $\|R_c\|$ | 685 | 156 | 536 | 3 | 29 |
| $\|R_s\|$ | 1312 | 157 | 714 | 47 | 72 |
| Random | | | | | |
| Precision $> 0.5$ | 27.7 | 71.6 | 53.1 | +150 | 139.2 |
| Precision $> 0.9$ | 86.5 | 84.5 | 121.6 | +150 | 139.2 |
| $Sc$ | | | | | |
| Precision $> 0.9$ | 22 | 2 | 14 | +150 | 3 |
| Ratio @ 0.9 | 3.93 | 42.25 | 8.69 | 1 | 46.4 |

| | Q6 | Q7 | Q8 | Q9 | Q10 |
|---|---|---|---|---|---|
| $\|R_c\|$ | 462 | 458 | 175 | 116 | 385 |
| $\|R_s\|$ | 654 | 912 | 382 | 459 | 477 |
| Random | | | | | |
| Precision $> 0.5$ | 27.5 | 15.2 | 51.7 | 121.1 | 57.9 |
| Precision $> 0.9$ | 51.4 | 34.5 | 59 | 121.4 | 84.1 |
| $Sc$ | | | | | |
| Precision $> 0.9$ | 3 | 2 | 17 | 14 | 26 |
| Ratio @ 0.9 | 17.13 | 17,25 | 3.47 | 8.67 | 3.23 |

**Table 4: Number of elements from $S$ in $R$, and cardinalities of result sets**

| Q | Q1 | Q2 | Q3 | Q4 | Q5 | Q6 | Q7 | Q8 | Q9 | Q10 |
|---|---|---|---|---|---|---|---|---|---|---|
| $\|S(R_c)\|$ | 5 | 19 | 5 | 0 | 6 | 4 | 19 | 4 | 7 | 6 |
| $\|S(R)\|$ | 6 | 19 | 6 | 0 | 6 | 4 | 21 | 4 | 8 | 6 |
| $\lambda = 0.7$ | | | | | | | | | | |
| $\|R'_c\|$ | 4 | 3 | 74 | 3575 | 3 | 1 | 1 | 26 | 2 | 1 |
| $\|R'_s\|$ | 21 | 38 | 225 | 10479 | 4 | 8 | 43 | 119 | 27 | 12 |
| Precision | 1 | 1 | 1 | | 1 | 1 | 1 | 0,192 | 1 | 1 |
| Recall | 0,01 | 0,05 | 0,22 | | 0,09 | 0,01 | 0,02 | 0,03 | 0,05 | 0,02 |
| $\lambda = 0.75$ | | | | | | | | | | |
| $\|R'_c\|$ | 42 | 36 | 357 | 3575 | 8 | 2 | 211 | 605 | 20 | 5 |
| $\|R'_s\|$ | 167 | 98 | 644 | 10479 | 27 | 32 | 523 | 931 | 116 | 29 |
| Precision | 1 | 1 | 1 | | 1 | 1 | 0,88 | 0,18 | 1 | 1 |
| Recall | 0,07 | 0,47 | 0,79 | | 0,33 | 0,02 | 0,5 | 0,49 | 0,25 | 0,04 |
| $\lambda = 0.8$ | | | | | | | | | | |
| $\|R'_c\|$ | 685 | 124 | 357 | 3575 | 24 | 1232 | 287 | 1458 | 97 | 22 |
| $\|R'_s\|$ | 1312 | 157 | 644 | 10479 | 64 | 1952 | 588 | 1729 | 407 | 91 |
| Precision | 1 | 1 | 1 | | 1 | 0,31 | 0,83 | 0,17 | 1 | 1 |
| Recall | 1 | 0,96 | 0,79 | | 0,88 | 0,87 | 0,55 | 0,92 | 0,9 | 0,12 |

threshold will limit the number of characterizations considered to generate the query, and in turn less selection conditions will increase the size of the result set. Let us note that is it crucial to find *some* characterizations for the approach to work, and that should there be no characterization with a specificity degree above a given $\lambda$, then the top-k characterizations should be considered instead (with $k = 3$ for instance). $\lambda$ values higher than 0.8 become too restrictive.

**Table 5: Number of characterizations obtained for the positive set of examples with different specificity thresholds**

| $\lambda$ | Q1 | Q2 | Q3 | Q4 | Q5 | Q6 | Q7 | Q8 | Q9 | Q10 |
|---|---|---|---|---|---|---|---|---|---|---|
| 0.7 | 8 | 9 | 8 | 0 | 9 | 8 | 8 | 8 | 8 | 8 |
| 0.75 | 5 | 5 | 4 | 0 | 5 | 4 | 5 | 4 | 5 | 5 |
| 0.8 | 2 | 2 | 4 | 0 | 3 | 2 | 3 | 2 | 4 | 4 |

# 6. CONCLUSION

In this paper we presented FQBE, a Query by Example approach to help users browse databases by simply evaluating examples. We proposed a method to select which examples to submit for user evaluation, which provides better results than a random selection of examples. We also showed that for most tested queries, FQBE is capable of inferring the user preferences and returning only original satisfactory results.

Future work include conducting experiments on other real-world datasets, as well as taking into account user feedback to alter the inferred user preferences. We also intend to conduct a user study so as to evaluate the benefits of FQBE.

**Acknowledgments** This work has been partially funded by the French DGE (Direction Générale des Entreprises) under the project ODIN (Open Data INtelligence).

# 7. REFERENCES

[1] M. De Calmès, D. Dubois, E. Hullermeier, H. Prade, and F. Sedes. Flexibility and fuzzy case-based evaluation in querying: An illustration in an experimental setting. *International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems*, 11(01):43–66, sep 2003.

[2] D. Dubois and H. Prade. Weighted minimum and maximum operations in fuzzy set theory. *Information Sciences*, 39(2):205–210, sep 1986.

[3] J. Fürnkranz and E. Hüllermeier. *Preference Learning*. Springer Berlin Heidelberg, Berlin, Heidelberg, 2011.

[4] S. Guillaume and B. Charnomordic. Generating an interpretable family of fuzzy partitions from data. *IEEE Transactions on Fuzzy Systems*, 12(3):324–335, 2004.

[5] N. Jayaram, A. Khan, C. Li, X. Yan, and R. Elmasri. Querying knowledge Graphs By Example entity tuples. In *ICDE*, volume 27, pages 1494–1495. IEEE, may 2016.

[6] M. Kearns and L. Valiant. Cryptographic Limitations on Learning Boolean Formulae and Finite Automata. *Journal of the ACM*, 41(1):67–95, 1994.

[7] C. Marsala and B. Bouchon-Meunier. Fuzzy partitioning using mathematical morphology in a learning scheme. In *FUZZ'IEEE*, volume 2, pages 1512–1517. IEEE, 1996.

[8] A. Moreau, O. Pivert, and G. Smits. A Fuzzy Approach to the Characterization of Database Query Answers. In *IPMU*, Eindhoven, Netherlands, 2016.

[9] D. Mottin, M. Lissandrini, Y. Velegrakis, and T. Palpanas. Exemplar Queries: Give me an Example of What You Need. *VLDB*, pages 365–376, 2014.

[10] G. Smits, M.-J. Lesot, and O. Pivert. Vocabulary Elicitation for Informative Descriptions of Classes. In *IFSA-SCSI*, Otsu, Japan, 2017.

[11] G. Smits, O. Pivert, and T. Girault. ReqFlex: Fuzzy Queries for Everyone. *VLDB*, 6(12):1206–1209, aug 2013.

[12] L. A. Zadeh. Fuzzy Sets. *Information and Control*, 8(3):338–353, 1965.

[13] S. Zadrozny, J. Kacprzyk, and M. Wysocki. On a novice-user-focused approach to flexible querying: The case of initially unavailable explicit user preferences. *ISDA*, pages 696–701, 2010.