A Tour from Regularities to Exceptions

Fabrizio Angiulli, Fabio Fassetti, Luigi Palopoli and Domenico Ursino

1 Overview

The enormous growth of information available in database systems has led to a significant development of techniques for knowledge discovery. At the heart of the knowledge discovery process is the application of data mining algorithms in charge of extracting hidden relationships among pieces of stored information. Information thus extracted from databases have widespread use in great many application domains and contexts.

In this paper, we focus on traditional, structured databases as data sources. In this realm information is available at two different levels, that of the database instance (known as *extensional level*) and that of the database schema (often called *intensional level*). For both of them, it is interesting to devise methods that automatically extract useful information to be used, for instance, in re-engineering applications. Also, it is sometimes useful to search databases for hidden regularities or, viceversa, to look for exceptional data bunches. These two dichotomies sketch, in a nutshell, the structure of this paper in which we summarize our long-term contributions in the area of data mining and data integration.

F. Angiulli (✉) · F. Fassetti · L. Palopoli
DIMES, University of Calabria, Calabria, Italy
e-mail: fabrizio.angiulli@unical.it

F. Fassetti
e-mail: fabio.fassetti@unical.it

L. Palopoli
e-mail: luigi.palopoli@unical.it

D. Ursino
DICEAM, University "Mediterranea" of Reggio Calabria, Calabria, Italy
e-mail: ursino@unirc.it

S. Flesca et al. (eds.), *A Comprehensive Guide Through the Italian Database Research Over the Last 25 Years*, Studies in Big Data 31, DOI 10.1007/978-3-319-61893-7_18

To illustrate, in the first part of this paper, we deal with discovering regularities and exceptions in data. Searching for regularities in data instances is, actually, at the heart of many data mining tasks, including classification, regression, clustering and rule discovery. This latter is the specific subject we shall focus on first, by considering two forms of it, namely, associative rules induction and metaquerying. After that, we shall discuss the problem of looking for exceptions, aka outliers in database instances. In this respect, it must be noted that searching regularities and exceptions does not complement one another, and specific and often unrelated techniques are needed in order to look for them. We shall discuss about outlier detection in ordinary data instances and in data streams and about the related problems of constructing explanations justifying for data bunches to be considered as exceptional.

Finally, we tackle the context of extracting regularities and exceptions in data schemas and we illustrate their exploitation to the problem of data integration and, ultimately, of information system cooperation.

2 Extensional Data Mining

2.1 Dependency Discovery

This task aims at revealing types of correlations holding among pieces of information stored in a database, which can be somehow expressed in the form of logical-like rules. For instance, with boolean association rules, one describes co-occurrences of items within databases of itemsets.

Not all rules induced from a database are, however, interesting: this will be the case only if it describes a relationship that is "mostly valid". To state such a validity, *indices* are used, that is, functions with values usually in [0, 1]. Support and confidence are classical indices employed in the data mining field. Intuitively, when a rule scores a high *support*, it is worth to further consider it, since there exist a significant fraction of the database tuples that satisfy the conjunction of the atoms in the rule. *Confidence* shows to what extent a given rule is true within the database at hand. Just for the way of example, a *confidence* value of 0.7, associated to the boolean association rule of the form $I_1, I_2, \ldots, I_n \rightarrow I$, tells that 70 percent of the stored itemsets where $I_1, I_2, \ldots, I_n$ are true (that is, occur) also have I true.

2.1.1 Association Rules Induction

Above we have made reference to the simplest form of an association rule, that is, the boolean one. In several application domains, though, they are not enough to encode interesting data dependencies.

Quantitative association rules use conditions of the form: $(i)\ A = c$; $(ii)\ A \neq c$; $(iii)\ A \in [l, u]$; $(iv)\ A \notin [l, u]$; where A is an attribute, l and u are values and $[l, u]$

denotes the set of numbers x s.t. $l \leq x \leq u$. In either forms, inducing association rules is a quite widely used data mining technique. Despite their widespread utilization, a thorough analysis of the complexity of the associated tasks was not been developed when we begun working on the subject.

We define a form of association rules that generalizes over the quantitative, categorical and the boolean attributes. We allow the null values (indicated by ϵ) to occur in the database, denoting the absence of information, for which it is forbidden to specify conditions. We capture [13, 15] the formal framework of boolean rules by calling *boolean* a database defined on a set of attributes taking value over $\{c, \epsilon\}$, where c is an arbitrary constant. In this setting, an association rule $(B_1 = c) \wedge \ldots \wedge (B_p = c) \Rightarrow (H_1 = c) \wedge \ldots \wedge (H_q = c)$ will encode the boolean association rule $B_1, \ldots, B_p \Rightarrow H_1, \ldots, H_q$.

Given a database T, support and confidence for association rule $B \Rightarrow H$ are:

- the *support* of $B \Rightarrow H$ in T, written $sup(B \Rightarrow H, T)$, is $|T_{B \wedge H}|/|T|$, and
- the *confidence* of $B \Rightarrow H$ in T, written $cnf(B \Rightarrow H, T)$, is $|T_{B \wedge H}|/|T_B|$,

where T_C denotes the set of tuples of T which satisfy the condition C. In our investigation, we considered two additional indexes, namely *gain* and *laplace*, that we shall not discuss further here.

Association rule induction problem: *Let I be a set of attributes, let T be a database on I, let k, $1 \leq k \leq |I|$, be a natural number, and let s, $0 < s \leq 1$, be a rational number. Furthermore, let ρ be an index. The association rule induction problem $\langle I, T, \rho, k, s \rangle$ is as follows: Is there a non-trivial association rule R such that $|R| \geq k$ and $\rho(R, T) \geq s$?*

Complexity results are summarized in Table 1. Before commenting on those, we recall that problems belonging to classes as AC^0, TC^0, L, and LOGCFL are very efficiently parallelizable (indeed $AC^0 \subseteq TC^0 \subseteq NC_1 \subseteq L \subseteq LOGCFL \subseteq NC_2 \subseteq P$; the reader is referred to [24] for basics on complexity classes), so that the algorithm design effort could be addressed accordingly:

Table 1 Complexity results for the *Association rule induction problem*

	Index	Database type	Constraints	Complexity
1	*sup*, *gain*, *laplace*	No nulls	No	NP-complete
2	*cnf*	No nulls	No	TC^0
3	All	With nulls	No	NP-complete
4	All	Sparse	No	L
5	All	Any	$\|I\|$ fixed	L
6	*sup*	With nulls	s fixed	NP-complete
7	*sup*	Boolean	$s\|T\|$ or k fixed	TC^0
8	*sup*	Boolean	$s\|T\|$ and k fixed	AC^0_2

- All the problems are NP-complete in the presence of null values (row 3 of Table 1); therefore, dealing with nulls makes "per se" the task of rule induction very demanding;
- The problem $\langle I, T, cnf, k, s\rangle$ becomes tractable when databases without nulls are considered (row 2), while the other problems remain intractable (row 1); this means that generating rules with high-confidence is easier than the case of other indices;
- In the presence of a bound on the length of the tuples (row 4) or of a bound on the number of attributes (row 5), then all the problems become highly parallelizable; this result can be understood if one considers that, when such bounds are imposed, the number of candidate rules is polynomial and different rules can be generated independently from one another.

2.1.2 Metaquerying

Metaquerying is a task aiming at extracting first order logical rules from a relational and a deductive database. This is a semi-supervised task in that the metaquery serve as the description of a class of patterns that the user is willing to discover. Differently from canonic association rules induction, patterns discovered using metaqueries can link information from several tables in databases. To illustrate, a metaquery **MQ** has the form $T \leftarrow L_1, \ldots, L_m$ where T and L_i are literal schemes $Q(Y_1, \ldots, Y_n)$, and Q is either an ordinary predicate name or a predicate variable. In the latter case, $Q(Y_1, \ldots, Y_n)$ can be instantiated to an atom with a predicate name denoting a relation in the database.

Answering a metaquery **MQ** on a database instance **DB** amounts to finding all substitutions σ, also called *instantiations*, of relational patterns appearing in **MQ** by atoms having as predicate names relations in **DB**, such that the Horn rule σ(**MQ**) holds in **DB** with a certain degree of plausibility, defined in terms of indices such as *support* and *confidence*.

We worked in the context outlined above with the two-sided aim of defining a clear and well-defined semantics and of studying the complexity of the underlying computational problems [11, 12, 14].

Metaquery semantics: *Let* **MQ** *be a metaquery,* **DB** *a database, and* σ *an instantiation. According to the type of the instantiation* (*namely, 0, 1, or 2*), *for any relational pattern* L *and atom* A, $\sigma(L) = A$ *implies the following:*

- ***type–0****: L and A have the same list of arguments;*
- ***type–1****: the arguments of A are obtained from those of L by permutation.*
- ***type–2****: the arguments of A are a superset of those of L, and the ones not appearing in L do not occur elsewhere in the instantiated rule.*

Metaquery induction problem: Let $T \in \{0, 1, 2\}$ *be an instantiation type and let* I *be a plausibility index. Let* **DB** *denote a database instance,* **MQ** *a metaquery, and* k *a threshold,* $0 \leq k < 1$. *Then, the combined* (*data, resp.*) *complexity of* $\langle$**DB**, **MQ**, $I, k, T\rangle$ *is the complexity, measured in the size of* **DB**, **MQ** *and* k (**DB**,

Table 2 Complexity results for the *Metaquery induction problem*

	Complexity measure	Problem type	Instantiation type	Indices	Threshold	Complexity
1	Comb. Compl.	General	0, 1, 2	**I**	$k = 0$	NP-complete
2	Comb. Compl.	General	0, 1, 2	cvr, sup	$0 \leq k < 1$	NP-complete
3	Comb. Compl.	General	0, 1, 2	cnf	$0 \leq k < 1$	NP^{PP}-complete
4	Comb. Compl.	Acyclic	0	**I**	$k = 0$	LOGCFL-complete
5	Comb. Compl.	Acyclic	1, 2	**I**	$k = 0$	NP-complete
6	Comb. Compl.	Acyclic	1, 2	cvr, sup	$0 \leq k < 1$	NP-complete
7	Comb. Compl.	Semi-acyclic	0, 1, 2	**I**	$k = 0$	NP-complete
8	Data Compl.	General	0, 1, 2	**I**	$k = 0$	AC_0
9	Data Compl.	General	0, 1, 2	**I**	$0 \leq k < 1$	TC_0

resp.), *of deciding if there exists a type-*T *instantiation* σ *such that* $I(\sigma(\mathbf{MQ})) > k$, *where* **DB** *has variable schema* (*fixed schema, resp.*).

Complexity results are summarized in Table 2, where it can be read that:

- The combined complexity of: (1) $\langle \mathbf{DB}, \mathbf{MQ}, I, 0, T \rangle$ is NP-complete for any instantiation type T; (2) $\langle \mathbf{DB}, \mathbf{MQ}, I, k, T \rangle$ is NP-hard for any index I; (3) $\langle \mathbf{DB}, \mathbf{MQ}, I, k, T \rangle$ is in NP for $0 \leq k < 1$, any T, and $I \in \{cvr, sup\}$.
- Instantiating metaqueries complying with a given bound on the confidence value turns out to be more complex than for other indices. This is due to the need of computing the *exact* count of tuples satisfying the body of an instantiation, whereas for other indices this is not required. In fact, this problem is related to the $\#P$-complete problem #SAT, where the question concerns *counting* the exact number of solutions of a given boolean formula. The class $\#P$ employed as an oracle is equivalent to another class related to counting, namely PP. Specifically, the combined complexity of $\langle \mathbf{DB}, \mathbf{MQ}, cnf, k, T \rangle$ turns out to be NP^{PP}-complete, a class "close" to PSPACE. Interestingly enough, the above result states the first natural problem known for the complexity class NP^{PP}.
- In order to single out tractable cases in the context of the combined complexity analysis, we individuate the classes of (*semi–*)*acyclic* metaqueries on the basis of the acyclicity of the (*semi–*)*hypergraph* associated with a metaquery: nodes are associated to both predicate and ordinary (only ordinary, in the case of the semi-hypergraph) variables occurring in the metaquery, while edges ecompass all variables appearing together in a certain relational pattern. Interestingly, the combined complexity of $\langle \mathbf{DB}, \mathbf{MQ}, I, 0, 0 \rangle$ for acyclic metaqueries is LOGCFL-complete, hence, highly parallelizable. However, acyclicity is not sufficient to guarantee tractability in general.
- As for the data complexity, depending on the threshold k, its membership ranges from AC_0 ($k = 0$) to TC_0 ($0 \leq k < 1$), hence, highly parallelizable.

2.2 Mining Exceptions

Outlier mining approaches tackle the knowledge discovery problem from a perspective which is reversed with respect to that of regularity mining ones. Specifically, the goal of outlier detection techniques is to isolate a few individuals deemed as exceptional, also referred to as *outliers*.

While in many contexts outliers are considered as noise that must be eliminated, as pointed out elsewhere "one persons noise could be another persons signal", and thus outliers themselves can represent the knowledge of interest in many applications, as in medical analysis, intrusion detection, surveillance systems, data cleaning, to cite a few.

Actually, exception mining techniques can be grouped in two main categories, that are *outlier detection* and *outlier explanation*.

2.2.1 Outlier Detection

Outlier identification has its roots in statistics: "An outlier is an observation that deviates so much from other observations as to arouse suspicions that it was generated by a different mechanism". According to most statistical approaches, outliers are those points that satisfy a discordancy test, that is, that are significantly far from what would be their expected position given the hypothesized distribution [1].

Approaches to outlier detection pertaining to the data mining field can be classified in supervised, semi-supervised, and unsupervised.

Unsupervised outlier detection. Unsupervised methods search for outliers in an unlabelled data set by assigning to each object a score which reflects its degree of abnormality. Most of the unsupervised approaches proposed in the data mining literature can be classified as deviation-based, distance-based, density-based, isolation-based, and others [1].

Among the unsupervised outlier detection methods, distance-based outlier ones occupy a prominent position. Distance-based outlier detection has been introduced by [26] to overcome the limitations of model-based statistical methods, that are the methods requiring that the data fits an hypothesized distribution. According to the original definition, an object obj is a distance-based outlier in a dataset with respect to parameters k and R if less than k objects in the data set lie within distance R from obj. Subsequently, some variants of the basic distance-definition have been introduced in the literature that have become popular during the years. Specifically, in order to provide a ranking of the outliers and with the aim of taking into account the whole neighborhood of the objects, [9] proposed to rank objects on the basis of the sum of the distances $\omega_k(\cdot)$ (or, equivalently, the average distance) from their k nearest neighbors, a measure called *weight* and also referred to as aKNN score (for average KNN).

The first two algorithms for mining distance-based outliers in large data sets were presented in [26]. However, none of these methods scales well for both large and

high-dimensional data, and this originates efforts for developing scalable algorithms that are scalable in both the size and the dimensionality of the data.

HilOut algorithm. Within this scenario [9, 10] proposed an algorithm, called *HilOut*, for detecting the top-n distance-based outliers according to the weight score. The major contributions of this research have been a novel distance-based outlier definition and the first distance-based outlier algorithm guaranteeing an approximate solution within a deterministic factor in time linear with respect to the dataset size.

The algorithm relies on the definition of approximate set $\{a_1, \ldots, a_n\}$ of outliers: elements a_i of this set have a weight greater than the weight of the true outliers within a pre-defined factor $\epsilon \geq 1$ (that is $\epsilon\omega_k(a_i) \geq \omega_k(o_i)$, where o_i denotes the true i-th top outlier, $i = 1, \ldots, n$).

Let n and d be the dataset size and dimensionality, respectively. The algorithm consists of two phases. The first provides an approximate solution, within a rough deterministic factor, after executing at most $d + 1$ sorts and scans of the data set, with temporal cost $O(d^2nk)$ and spatial cost $O(nd)$,

Specifically, the algorithm avoids the distance computation between each pair of points by making use of the space-filling curves (and, specifically, the Hilbert curve), that are mappings of an hypercube $D = [0, 1]^d$ into the interval $I = [0, 1]$, to linearize the dataset. The mapping assures that if two points are close in I, they are close in D too, although the reverse in not always true.

During the first phase the algorithm calculates a lower and an upper bound to the weight of each point by exploiting Hilbert curves, and determines the points candidate to belong to the solution set, or candidate outliers. If the number of candidates n^* is n, then the algorithm stops reporting the exact solution. Otherwise, the second phase is needed, which calculates the exact solution with a final scan of temporal cost $\mathcal{O}(n^*nd)$. Experimental results highlighted that in practice the algorithm always stops, reporting the exact solution, during the first phase after much less than $d + 1$ steps.

DOLPHIN algorithm. The DOLPHIN algorithm [6, 7] detects distance-based outliers according to the definition of [26]. The algorithm, designed to work on disk-resident datasets, maintains an in-memory data structure, called INDEX. DOLPHIN performs two sequential scans of the dataset file. During the first scan INDEX is employed to maintain a summary of the portion of the dataset already examined. In particular, for each incoming dataset object obj, the objects stored in INDEX are exploited in order to determine if obj is an inlier. The object obj will be inserted into INDEX if it is not recognized as an inlier. By adopting this strategy it is guaranteed that INDEX contains all the outliers occurring in the portion of the dataset already scanned. Moreover, some of the objects stored in INDEX, called *proved inliers*, can be recognized as inliers on the basis of the objects read after them. When the first dataset scan finishes, INDEX contains a superset of the dataset outliers. During the second scan, the candidate outliers stored in INDEX are compared with all the dataset objects and at the end of the scan, INDEX contains all and only the outliers of the dataset.

It is proved that the size of INDEX is $O(k/p)$, where p denotes the probability that two randomly picked objects, one from INDEX and the other from the dataset, are neighbors. As for the value of k/p, for meaningful combinations of the parameters

k and R, that are those associated with a pre-determined fraction α (usually of the order of 1%) of objects to be recognized as outliers, provably corresponds to a small fraction (of the order of 1%) of the dataset size. Importantly, probably approximately correct values for k and R associated with a given α can be quickly determined by means of a fixed-size sampling procedure.

Summarizing, the spatial cost is $O(k/p)$, the temporal cost is $O(nk/p)$, which for k fixed is linear in the dataset size, and the I/O cost is linear, since it corresponds to the cost of sequentially reading the input dataset file twice. DOLPHIN has been compared with state of the art distance-based outlier detection algorithms showing that it is much more efficient.

Outliers in data streams. In many emerging applications, such as fraud detection, network flow monitoring, telecommunications, data management and others, data arrive continuously and it is either unnecessary or impractical to store all incoming objects. In this context, a challenge is to find the most exceptional objects among the flow of incoming data, also called a *data stream*. Data mining on data streams is often performed based on certain time intervals, called windows.

Finding outliers in data streams is a relatively novel and challenging research area [25]. The method proposed in [8], called STORM (for STream OutlieR Miner), introduces a novel concept of querying for outliers. Specifically, previous work deals with continuous queries, that are evaluated as objects arrive. Conversely, one-time queries are evaluated once over a point-in-time. The underlying intuition is that, due to evolution, stream characteristics can change over time and, hence, by classifying single objects when a data analysis is required, the concept drift, a challenging characteristics of data streams, can be captured.

Semi-supervised outlier detection. Semi-supervised methods assume that only normal examples are given. The goal is to find a description of the data, that is a rule partitioning the object space into an accepting region, containing the normal objects, and a rejecting region, containing all the other objects. These methods are also called one-class classifiers or domain description techniques.

In [16] the concept of *outlier detection solving set* is defined, a subset of the input data set representing a model that can be used to predict distance-based outliers according to the weight score. The computational complexity of computing a minimum cardinality solving set is analyzed, showing that it is in general an intractable problem. An algorithm, called *Solving Set*, that computes with sub-quadratic time requirements a solving set and the top-n outliers is described, and experimental evidence that the solving set is a fraction of the overall data set and that the false positive rate obtained using it is negligible is given.

Parallel/distributed and GPU based strategies for solving set computation are described in [19, 21]. Other compressed representation for novelty detection based on distance-based definitions are introduced in [4, 5] and compared with well-established one-class classification methods.

Outlier Explanation. In many real situations, one is given a data population characterized by a certain number of attributes, and information are provided that one of the individuals in that data population is abnormal, but no reason whatsoever is

given as to why this particular individual is to be considered abnormal. The problem we deal with next is precisely to single out such reasons.

This problem has many practical applications. For instance, in analyzing health parameters of a sick patient it is relevant to single out parameters mostly differentiating them from those of the healthy population. As another example, a data history associated with an athlete that has established an exceptional performance can be analyzed to detect characteristics determining that performance.

Despite its wider applicability, a limited attention has been paid to the subject at the time we started working on it. We tackled the problem under several directions, since, due to its peculiarities, it needed the designing of specific techniques on the basis of one or more outliers in input and on the basis of the presence of numerical or categorical attributes. The following table reports the techniques we develop for the different contexts, together with the referred work.

	categorical data	numerical data
one outlier	FOP [17, 18]	OPD [23]
more outlier	EXPREX [20]	EXPREX [20]

In order to illustrate the scenario, we refer to the first case, namely a categorical dataset and one outlier provided in input. Nevertheless, the underlying ideas are shared by the different scenarios.

Consider the example consisting in a portion of the Zoo dataset (Fig. 1a). This database consists of 15 boolean-valued and two numerical attributes representing animal features. The table reports some of the attributes collected for some animals. It is known that the platypus is an exceptional animal being it a mammal, but laying eggs. This intuitive notion can be formally illustrated on the database by noticing that among dataset objects having value "y" for the attribute *eggs*, the platypus is the only animal having value "y" for the attribute *milk*. Obviously the value "y" for the

name	eggs	milk	legs
⋮	⋮	⋮	⋮
flea	y	n	6
gnat	y	n	6
gull	y	n	2
octopus	y	n	8
ostrich	y	n	2
parakeet	y	n	2
penguin	y	n	2
platypus	y	y	4
polecat	n	y	4
pony	n	y	4
wasp	y	n	6
⋮	⋮	⋮	⋮

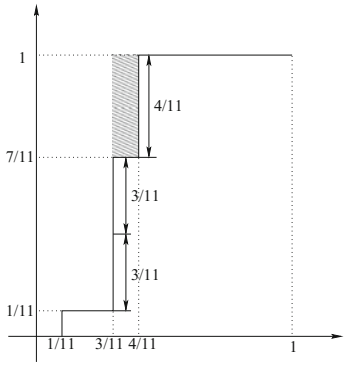
(a)

legs	freq f
⟨2⟩	4
⟨4⟩	3
⟨6⟩	3
⟨8⟩	1

(b)

f	ff	*cum* ff
1	1	1
3	6	7
4	4	11

(c)



(d)

Fig. 1 Illustration of the outlier explanation problem

attribute *milk* is not an exceptional feature "per se", but it is surprising when attention is restricted to the animals which lay eggs. This is a case where a *local* property is individuated, where the attribute *eggs* plays the role of *explanation* for the *outlying property*, *milk*, of the platypus.

The intuition underlying our definition of exceptionality is that a set of attributes, or *property*, makes an object exceptional if the frequency of the combination of values assumed by that object on those attributes is rare if compared to the frequencies associated with the other combinations of values assumed on the same attributes by the other objects of the database.

Indeed, considering frequency values may lead to incorrect conclusions. As an example, consider a key-attribute. Obviously, the value assumed by the outlier object on that attribute occurs just once on the dataset, but this cannot be considered exceptional since all the values occur once.

Our approaches, then, try to capture this intuition in the various scenarios where we search for *property-explanation pairs*.

One outlier – Categorical data. The technique we propose to deal with the case of one user-provided outlier in a categorical dataset is based on (1) the construction of the histogram of frequency values (Fig. 1b), (2) the construction of the cumulated histogram of the frequency of frequencies (Fig. 1b) and, (3) the quantification of the "degree of unbalanceness" between the frequency of the object o under consideration and the frequencies of the rest of the database (Fig. 1b). This latter step is performed by measuring the area above the cumulated frequency of frequencies histogram of the database w.r.t. the set of attributes of interest, starting from the frequency of o. Indeed, the larger this area is, the smaller the frequency of o is w.r.t. the frequencies associated with the other data set objects. To illustrate, Fig. 1 reports the computation of the outlierness of the property *legs* for the platypus.

One outlier – Numerical data. When dealing with numerical attributes, in order to extend the previously stated intuition, a key aspect is being able to efficiently estimate both the *cumulated density frequency* (*cdf*) and the related *probability density frequency* (*pdf*), as well as to exploit them to measure the associated imbalance. This is in fact our main contribution in this scenario.

The *Outlying Property Factor* (OPF) of a numerical attribute a in o w.r.t. the dataset is defined as follows:

$$OPF_a(o) = \Omega\left(\int_{f_a(o[a])}^{\sup(f_a)} (1 - G_a(f))\mathrm{d}f - \int_0^{f_a(o[a])} G_a(f)\mathrm{d}f\right). \quad (1)$$

where, Ω denotes a function from the set of real numbers to [0, 1], and

$$G_a(\varphi) = Pr(X_a^f \leq \varphi) = \int_0^{\varphi} Pr(X_a^f = v)\mathrm{d}v, \quad (2)$$

with X_a^f denoting the random variable whose pdf represents the relative likelihood for the pdf f_a, representing the density of the active domain of a, to assume a certain value. Thus, $G_a(\varphi)$ measures the probability to observe in a a density value not

exceeding φ. To maintain the analogy with the previous definition, the first integral in OPF measures the *area above* the cdf $G_a(f)$ for $f > f_a(o[a])$, while the second integral in OPF measures the *area below* the cdf G_a for $f \leq f_a(o[a])$.

Two problems are related to the applicability above definition: (1) to estimate the empirical pdfs from dataset values, and (2) to determine the "natural" intervals of homogeneous values to be employed to form explanations. The strategies we have designed to solve these two problems exploit a common framework, which is based on Kernel Density Estimation (KDE).

More outliers. The work [20] extends the perspective of previously described approaches in order to be able to deal with *groups*, or *sub-populations*, of anomalous individuals. As an example, consider a restrict group of longevous individuals; it would be very useful to single out properties, namely genetic traits, differentiating them as a whole from normal individuals.

We designed exceptionality scores well-tailored for comparing a rare population with a large one, which exploit the notion of *randomization test* based on the *Pearson chi-square* criterion, for categorical properties, and on the *Cramér-von-Mises* criterion, for numerical properties. These criteria evaluate the badness of fit of a probability distribution F compared to a sample set. In particular, we employ as reference distribution F the empirical distribution function associated with the population of inliers and, as the sample set, the population of outliers.

3 Intensional Data Mining

Analogously to what happens for the extensional level [36], also for the intensional one it is possible to define regularities and exceptions. In particular, both of them, as a whole, form the so-called *interschema properties* [29, 32].

These can be partitioned into *nominal properties* [30], which involve lexicon, and *structural properties* [37], which involve the structure of sources to integrate.

Nominal properties, in turn, can be divided in synonymies, homonymies and hyponymies. A *synonymy* between two concepts C_1, belonging to a schema S_1, and C_2, belonging to a schema S_2, indicates that they have the same meaning but different names. A *homonymy* between $C_1 \in S_1$ and $C_2 \in S_2$ denotes that they have the same name but different meaning [29, 32]. A concept $C_1 \in S_1$ is a hyponym of a concept $C_2 \in S_2$ (which, in turn, is the hypernym of C_1) if C_1 has a more specific meaning than C_2 [31]. Synonymies and homonymies are examples of regularities, whereas homonymies represent a case of exceptions.

The main *structural properties* [40] are type conflicts, subschema similarities and complex knowledge patterns. A *type conflict* denotes that the same concept is represented by means of different schema structures in the involved databases (for instance, by means of an attribute in S_1 and an entity in S_2). A *sub-schema similarity* indicates that a subschema of S_1 and another of S_2 could represent the same concept although they seem to have a different structure. A *complex knowledge pattern* represents a (generally complex) relationship involving several concepts possibly

belonging to several different schemas [28, 35]. Also for structural properties we can recognize regularities (in this case, subschema similarities) and exceptions (in this case, type conflicts). Interestingly, complex knowledge patterns can encompass both regularities and exceptions. In the literature, different techniques for the extraction of interschema properties have been proposed. The interested reader can find a survey in [38].

In this chapter, we shall illustrate one of them based on the assumption that two concepts are similar if the concepts belonging to their neighborhoods are similar. Now, the question is: given a concept C, which are its neighbors? To answer this question, it is necessary to define a metric aiming at detecting the semantic relationship degree between two concepts. Thanks to this metric, it is possible first to detect the closest neighbors of C, then to determine the neighbors of the closest neighbors, and so forth, until to a certain neighbor distance has been covered. Clearly, in the definition of the semantics of C, the contribution of its closest neighbors must be higher than the one of its farthest neighbors. In other words, as this process moves away from C, the weight of neighbors in determining the semantics of C decreases.

To define the metric of our interest, some support information is needed. To determine this information, it is necessary to associate some suitable graphs (called Semantic Distance Graphs - SD Graphs) with the involved E/R schemas. Specifically, given an E/R schema S, the corresponding SD-Graph can be represented as $G(S) = \langle N(S), D(S)\rangle$, where $N(S)$ indicates the set of the nodes of $G(S)$, whereas $A(S)$ denotes the set of its arcs. There is a node in $G(S)$ for each entity, relationship or attribute of S. $A(S)$ consists of two subsets, namely: (i) $SA(S)$, i.e., the set of *solid arcs*, denoting a strong relationship between two concepts; (ii) $DA(S)$, i.e., the set of *dashed arcs*, indicating a weak relationship between two concepts.

There is a solid arc: (i) from an entity or a relationship to each of its attributes; (ii) from a relationship to each of the entities linked by it; (iii) from a key attribute to the corresponding entity; (iv) from the "child" entity to the "father" one of an *isa* relationship.

There is a dashed arc: (i) from a non-key attribute to the corresponding entity or relationship; (ii) from an entity to each of the relationships it participates to; (iii) from a "father" entity to each of its "children entities" of an *isa* relationship.

The more the shortest path connecting two nodes encompasses dashed arcs, the more they must be considered semantically distinct. We define $D\text{-}path_n$ in $G(S)$ a path with n dashed arcs and any number of solid arcs. Given two nodes x and y in $G(S)$, the *D-shortest* path from x to y, denoted by $\langle x, y\rangle$, is the path from x to y characterized by the minimum number of D-arcs.

Given a node x, the neighborhood, or *context* of level $i \geq 0$ of x is defined as: $cnt(x, i) = \{y \mid y \in N(S), y \neq x, \langle x, y\rangle \text{is a } D - path_i \text{ in G(S)}\}$.

The A_cnt of level i $(i \geq 0)$ of a node x is defined as as the set of the attributes belonging to $cnt(x, i)$. The E_cnt of level i $(i \geq 0)$ of a node x encompasses all the entities belonging to all the contexts of level j $(0 \leq j \leq i)$ of x.

In order to determine if two concepts, belonging to two different schemas, are similar, it is necessary to first examine their contexts of level 0, then the ones of level 1, and so forth.

Let $x \in G(S_1)$ and $y \in G(S_2)$ be two nodes. In order to compute the similarity degree of their A_cnts, a full bipartite graph is constructed. In this graph, there is a node for each attribute of x and y. Each edge has associated a weight on the basis of the similarity degree of the corresponding attributes. Given two attributes A_1 of x and A_2 of y, the weight of the edges linking them is computed by means of the following formula: $\gamma(A_1, A_2) = w_l \times L(A_1, A_2) + w_d \times D(A_1, A_2) + w_k \times K(A_1, A_2)$, where w_n, w_d and w_k are weighting factors whose sum is equal to 1, and $L(A_1, A_2)$, $D(A_1, A_2)$ and $K(A_1, A_2)$ consider the lexicographic, domain and "key characterization" similarities of A_1 and A_2, respectively. After the bipartite graph has been constructed, the computation of the similarity degree of the two A_cnts is performed by computing a suitable objective function associated with the maximum weight matching related to this bipartite graph. This objective function is given by the sum of the weights of the selected arcs multiplied by 2 and divided by the number of the nodes of the bipartite graph. The corresponding value belongs to the real interval [0, 1]. In an analogous way, it is possible to compute the similarity degree of two E_cnts. Finally, the similarity degree $c - sim(x, y, i)$ of two contexts $cnt(x, i)$ and $cnt(y, i)$ is a weighted mean of the similarity degree of their A_cnts and their E_cnts.

To compute the similarity $sim(x, y)$ between x and y, it is necessary to apply an iterative procedure. First $c - sim(x, y, 0)$ is computed. Then, $c - sim(x, y, 1)$ is determined and exploited to refine the previously computed value. This way of proceeding is performed until to a user predefined level n is reached, or until to the value of $sim(x, y)$ reaches a fixed point, with farthest levels influencing the computed value less than closer ones. For this latter purpose, we exploit a quadratic decrease function.

Once the similarity coefficients of all the objects of two schemas S_1 and S_2 have been determined, it is possible to compute interschema properties:

- There exists a *synonymy* between two objects $C_1 \in S_1$ and $C_2 \in S_2$ if they have the same type (i.e., both of them are entities or relationships or attributes) and their similarity coefficient $sim(C_1, C_2)$ is higher than a certain threshold th.
- There exists a *homonymy* between two objects $C_1 \in S_1$ and $C_2 \in S_2$ if they have the same name and the same type but their similarity coefficient is lower than (1-th).
- There exists a *type conflict* between two objects $C_1 \in S_1$ and $C_2 \in S_2$ if they have different types but their coefficient is higher than a threshold th.

The interested reader can find all details about this approach in [32].

In order to compute *subschema similarities*, a very similar approach can be adopted. In fact, the neighborhood of a subschema is nothing more than the union of the neighborhoods of the objects forming the subschema [40]. The approach to computing *hyponymies* is very similar to the one described above [31]. Finally, in order to determine complex knowledge patterns, a variant of description logics can be exploited [28]. As previously pointed out, interschema properties play a key role in schema integration and, ultimately, in the construction of Cooperative Information Systems and Data Warehouses [39].

The interested reader can find the description of a system performing interschema property computation by means of the approach described above, as well as schema integration for the construction of a Cooperative Information Systems or a Data Warehouse in [33, 34].

4 Future Trends

As current trends, a renewed interest is witnessed for the field of metaquerying due to the success of the semantic web and to its strict relationship with ontological, that carries in the spotlight the need of querying classes and concepts [27]. Furthermore, notable research directions pertaining outliers are devoted to the needs of certain applications, e.g., in terms of interpretability or capability to describe the reasons underlying unusual behaviours of users, to provide user interfaces to navigate within data and to visualize outliers, and towards tailoring the discovery process on users' goals rather than on some pre-defined notion of abnormality [3].

As another interesting challenge for researchers, it is worth to emphasize the relevance of designing techniques able to work in domain so complex to necessarily require new and specialized methodologies. Among these, relevant contributions could be provided to fields like physics or biology because of the huge amount of data already produced in these areas characterized by high variability and uncertainty.

Moreover, of particular interest is the analysis of networks and of their dynamics related to the formidable popularity of online social networks and availability of huge amount of user-generated content. This has lead to a significant increase in the number of studies about social network in the area of the social sciences network modeling and analysis in the area of machine learning and data mining. All that is aimed to extract knowledge about relations and diffusion processes in order to shed lights on social behaviors and interactions among individuals or discover anomalous, malicious individuals who attempt to perform illegal activities and cause harm to other users [2, 22].

References

1. C. Aggarwal, *Outlier Analysis* (Springer, Berlin, 2013)
2. L. Akoglu, H. Tong, D. Koutra, Graph based anomaly detection and description: a survey. Data Min. Knowl. Discov. **29**(3), 626–688 (2015)
3. L. Akoglu, F. Bell, E. Müller, T.E. Senator (eds.), *ACM KDD Workshop on Outlier Definition, Detection and Description on Demand, San Francisco, USA* (2016)
4. F. Angiulli, Condensed nearest neighbor data domain description. IEEE Trans. Pattern Anal. Mach. Intell. **29**(10), 1746–1758 (2007)
5. F. Angiulli, Prototype-based domain description for one-class classification. IEEE Trans. Pattern Anal. Mach. Intell. **34**(6), 1131–1144 (2012)
6. F. Angiulli, F. Fassetti, An efficient method for outlier detection, in *SEBD* (2008), pp. 326–333

7. F. Angiulli, F. Fassetti, DOLPHIN: an efficient algorithm for mining distance-based outliers in very large datasets. ACM Trans. Know. Discov. Data **3**(1), 4:1–4:57 (2009)
8. F. Angiulli, F. Fassetti, Distance-based outlier queries in data streams: the novel task and algorithms. Data Min. Knowl. Discov. **20**(2), 290–324 (2010)
9. F. Angiulli, C. Pizzuti, Fast outlier detection in high dimensional spaces, in *PKDD* (2002), pp. 15–26
10. F. Angiulli, C. Pizzuti, Outlier mining in large high-dimensional data sets. IEEE Trans. Knowl. Data Eng. **17**(2), 203–215 (2005)
11. F. Angiulli, R. Ben-Eliyahu-Zohary, G. Ianni, L. Palopoli, Computational properties of metaquerying problems, in *PODS* (2000), pp. 237–244
12. F. Angiulli, G. Ianni, L. Palopoli, Metaquerying: proprietà e tecniche di implementazione, in *SEBD* (2000), pp. 317–330
13. F. Angiulli, G. Ianni, L. Palopoli, On the complexity of mining association rules, in *SEBD* (2001), pp. 177–184
14. F. Angiulli, R. Ben-Eliyahu-Zohary, G. Ianni, L. Palopoli, Computational properties of metaquerying problems. ACM Trans. Comput. Log. **4**(2), 149–180 (2003)
15. F. Angiulli, G. Ianni, L. Palopoli, On the complexity of inducing categorical and quantitative association rules. Theor. Comput. Sci. **314**(1–2), 217–249 (2004)
16. F. Angiulli, S. Basta, C. Pizzuti, Distance-based detection and prediction of outliers. IEEE Trans. Knowl. Data Eng. **18**(2), 145–160 (2006)
17. F. Angiulli, F. Fassetti, L. Palopoli, Un metodo per la scoperta di proprietà inattese, in *SEBD* (2006), pp. 321–328
18. F. Angiulli, F. Fassetti, L. Palopoli, Detecting outlying properties of exceptional objects. ACM Trans. Database Syst. **34**(1), 1–62 (2009)
19. F. Angiulli, S. Basta, S. Lodi, C. Sartori, Distributed strategies for mining outliers in large data sets. IEEE Trans. Knowl. Data Eng. **25**(7), 1520–1532 (2013)
20. F. Angiulli, F. Fassetti, L. Palopoli, Discovering characterizations of the behavior of anomalous subpopulations. IEEE Trans. Knowl. Data Eng. **25**(6), 1280–1292 (2013)
21. F. Angiulli, S. Basta, S. Lodi, C. Sartori, GPU strategies for distance-based outlier detection. IEEE Trans. Parallel Distrib. Syst. **27**(11), 3256–3268 (2016)
22. F. Angiulli, F. Fassetti, E. Narvaez, Anomaly detection in networks with temporal information, in *Discovery Science* (Italy, Bari, 2016), pp. 359–375
23. F. Angiulli, F. Fassetti, G. Manco, L. Palopoli, Outlying property detection with numerical attributes. Data Min. Knowl. Discov. **31**(1), 134–163 (2017)
24. M.R. Garey, D.S. Johnson, *Computers and Intractability: A Guide to the Theory of NP-Completeness* (W.H. Freeman, New York, 1979)
25. M. Gupta, J. Gao, C. Aggarwal, J. Han, Outlier detection for temporal data: a survey. IEEE Trans. Knowl. Data Eng. **26**(9), 2250–2267 (2014)
26. E. Knorr, R. Ng, A unified notion of outliers: properties and computation, in *KDD* (1997), pp. 219–222
27. M. Lenzerini, L. Lepore, A. Poggi, Answering metaqueries over hi (OWL 2 QL) ontologies, in *IJCAI, New York, USA* (2016), pp. 1174–1180
28. L. Palopoli, D. Saccà, D. Ursino, DL_P: a description logic for extracting and managing complex terminological and structural properties from database schemes. Inf. Syst. **24**(5), 410–424 (1999)
29. L. Palopoli, D. Saccà, D. Ursino, Semi-automatic techniques for deriving interscheme properties from database schemes. Data Knowl. Eng. **30**(4), 239–273 (1999)
30. L. Palopoli, G. Terracina, D. Ursino, A graph-based approach for extracting terminological properties of elements of XML documents, in *ICDE* (IEEE Computer Society, Heidelberg, Germany, 2001), pp. 330–337
31. L. Palopoli, D. Saccà, G. Terracina, D. Ursino, A technique for deriving hyponymies and overlappings from database schemes. Data Knowl. Eng. **40**(3), 285–314 (2002)
32. L. Palopoli, D. Saccà, G. Terracina, D. Ursino, Uniform techniques for deriving similarities of objects and subschemes in heterogeneous databases. IEEE Trans. Knowl. Data Eng. **15**(2), 271–294 (2003)

33. L. Palopoli, G. Terracina, D. Ursino, Dike: a system supporting the semi-automatic construction of cooperative information systems from heterogeneous databases. Softw. Pract. Exp. **33**(9), 847–884 (2003)
34. L. Palopoli, G. Terracina, D. Ursino, Experiences using DIKE, a system for supporting cooperative information system and data warehouse design. Inf. Syst. **28**(7), 835–865 (2003)
35. L. Palopoli, G. Terracina, D. Ursino, A plausibility description logic for handling information sources with heterogeneous data representation formats. Ann. Math. Artif. Intell. **39**(4), 385–430 (2003)
36. L. Pontieri, D. Ursino, E. Zumpano, An approach for the extensional integration of data sources with heterogeneous representation formats. Data Knowl. Eng. **45**(3), 291–331 (2003)
37. L. Palopoli, D. Rosaci, G. Terracina, D. Ursino, A graph-based approach for extracting terminological properties from information sources with heterogeneous formats. Knowl. Inf. Syst. **8**(4), 462–497 (2005)
38. E. Rahm, P. Bernstein, A survey of approaches to automatic schema matching. VLDB J. **10**(4), 334–350 (2001)
39. D. Rosaci, G. Terracina, D. Ursino, An approach for deriving a global representation of data sources having different formats and structures. Knowl. Inf. Syst. **6**(1), 42–82 (2004)
40. G. Terracina, D. Ursino, A uniform methodology for extracting type conflicts and subscheme similarities from heterogeneous databases. Inf. Syst. **25**(8), 527–552 (2000)