D53
Analysis Modules Selection and Implementation

VV. AA.

This deliverable consists of a large body of data analysis methods, in line with the initial classification described by D51, which have been prototyped and applied to use cases.

The deliverable is organized in four sections: the first section concerns the analysis of genomic data. It presents the main contributions of the GenData project to the analysis and annotation of the raw sequences produced by Next-Generation Sequencing experiments. The second section addresses the analysis of genomic metadata, which consist of sequence annotations and related information coming from external sources, including for instance clinical data.

The index of the deliverable sections is as follows:

1. **Analysis of genomic data**

   1.1 **Peak Shape analysis**.

   Abstract: *In recent years many techniques have been developed to study genetic and epigenetic processes. In particular, since 2005 Next Generation Sequencing (NGS) methods have revolutionized the genomic field by allowing fast and not very expensive sequencing. Among NGS method, chromatin immunoprecipitation followed by sequencing (ChIP-seq) permits to investigate protein-DNA interactions, e.g. the direct interaction between transcription factors, histones and DNA. At present, in the relevant literature, the analysis of ChIP-seq data is mainly restricted to the detection and the investigation of enriched regions (peaks) in the genome, considering only signal intensity. Motivated by the fact that these peaks have very different shapes, we propose to take into account*

also other features of peak shape, with the idea that statistically significant shape differences are associated with a functional role and a biological meaning.

1.2 **Discovering new gene functionalities from random perturbations of known gene ontological annotations**.
Abstract: *Computational analyses for biomedical knowledge discovery greatly benefit from the availability of the description of gene and protein functional features expressed through controlled terminologies and ontologies, i.e. of their controlled annotations. In the last years, several databases of such annotations have become available; yet, these annotations are incomplete and only some of them represent highly reliable human curated information. To predict and discover unknown or missing annotations existing approaches use unsupervised learning algorithms. We propose a new learning method that allows applying supervised algorithms to unsupervised problems, achieving much better annotation predictions. This method, which we also extend from our preceding work with data weighting techniques, is based on the generation of artificial labeled training sets through random perturbations of original data. We tested it on nine Gene Ontology annotation datasets; obtained results demonstrate that our approach achieves good effectiveness in novel annotation prediction, outperforming state of the art unsupervised methods.*

2. **Analysis of genomic metadata**

2.1 **An approach to on-demand ETL for the GOLAM framework**.
Abstract: *In traditional OLAP systems, the ETL process loads all available data in the data warehouse before users start querying them. In some cases, this may be either inconvenient (because data are supplied from a provider for a fee) or unfeasible (because of their size); on the other hand, directly launching each analysis query on source data would not enable data reuse, leading to poor performance and high costs. The alternative investigated in this paper is that of fetching and storing data on-demand, i.e., as they are needed during the analysis process. In this direction we propose the Query-Extract-Transform-Load (QETL) paradigm to feed a ROLAP cube; the idea is to fetch facts from the source data provider, load them into the cube only when they are needed to answer some OLAP query, and drop them when some free space is needed to load other facts. Remarkably, QETL includes an optimization step to cheaply extract the required data based on the specific features of the data provider.*

2.2 **Discovering frequent correlations from genomic metadata**.
Abstract: *Gene Expression Datasets (GEDs) usually consist of the expression values of thousands of genes within hundreds of samples. Frequent itemset and association rule mining algorithms have been applied to discover significant co-expressions among multiple genes from GEDs. To perform these data analyses, gene expression values are commonly discretized into a predefined number of bins. Such an expert-driven and not trivial preprocessing step could bias the quality of the mining result. This deliverable presents a novel approach to discovering gene correlations from GEDs which does not require data discretization. By representing per-sample gene expression values as item weights, frequent weighted itemsets can be extracted. The discovery of weighted itemsets instead of traditional (not weighted) ones prevents experts from discretizing GEDs before analyzing them and thus improves the effectiveness of the knowledge discovery process. Experiments performed on real GEDs demonstrate the effectiveness of the proposed approach.*

2.3 **Discovering frequent correlations from medical data**.
Abstract: *Physicians and healthcare organizations always collect large amounts of data during patient care. These large and high-dimensional datasets are usually characterized by an inherent sparseness. Hence, the analysis of these datasets to figure out interesting and hidden knowledge is a challenging task. This deliverable proposes a new data mining framework based on generalized association rules to discover multiple-level correlations among patient data. Specifically, correlations among prescribed examinations, drugs, and patient profiles are discovered and analyzed at different abstraction levels. The rule extraction process is driven by a taxonomy to generalize examinations and drugs into their corresponding categories. To ease the manual inspection of the result, a worthwhile subset of rules, i.e., the non-redundant generalized rules, is considered. Furthermore, rules are classified according to the involved data features (medical treatments or patient profiles) and then explored in a top-down fashion, i.e., from the small subset of high-level rules a drill-down is performed to target more specific rules. The experiments, performed on a real diabetic patient dataset, demonstrate the effectiveness of the proposed approach in discovering interesting rule groups at different abstraction levels.*

2.4 **Towards a statistical framework for attribute comparison in very large relational databases**.
Abstract: *The technological evolution and the multiplication of information sources has brought about an ever-increasing need of techniques for the analysis of large-scale databases. The recent re-*

search, generally collected under the umbrella of "Big Data Analytics", attempts to solve this many-sided problem. We describe a general methodology for the statistical analysis of large-scale databases with the aim to extract relevant, often implicit or unexpected, information about the distribution of the attribute values in two (large) tuple sets resulting from different queries on a large database. This analysis aims at helping users to gain knowledge about the datasets they are exploring. While a relatively large literature addressing a similar problem exists under the name of subgroup discovery, our framework presents the following distinctive features: 1) it manages both categorical and numerical attributes; 2) it represents subgroups as SQL queries; 3) the classification of attributes into unusualness or interest comprises statistical hypothesis tests and the Hellinger distance; 4) the search of relevant attributes relies on the joint use of sampling and incremental mechanisms for statistical hypothesis tests.

Section 1.1 — Peak Shape analysis

Marzia A. Cremona, Alice Parodi, Piercesare Secchi

MOX - Dipartimento di Matematica, Politecnico di Milano

1 Introduction

In recent years many techniques have been developed to study genetic and epigenetic processes. In particular, since 2005 Next Generation Sequencing (NGS) methods have revolutionized the genomic field by allowing fast and not very expensive sequencing. Among NGS method, chromatin immunoprecipitation followed by sequencing (ChIP-seq) permits to investigate protein-DNA interactions, e.g. the direct interaction between transcription factors, histones and DNA [11]. At present, in the relevant literature, the analysis of ChIP-seq data is mainly restricted to the detection and the investigation of enriched regions (peaks) in the genome, considering only signal intensity. Motivated by the fact that these peaks have very different shapes, as shown in Figure 1, we propose to take into account also other features of peak shape, with the idea that statistically significant shape differences are associated with a functional role and a biological meaning.

2 State of the art

A large number of algorithms and methodologies for the analysis of ChIP-seq data are currently available [12, 16]. However, almost all these techniques concentrate on the intensity of ChIP-seq signal. The shape of the peaks varies a lot among the experiments for different proteins as well as among different areas

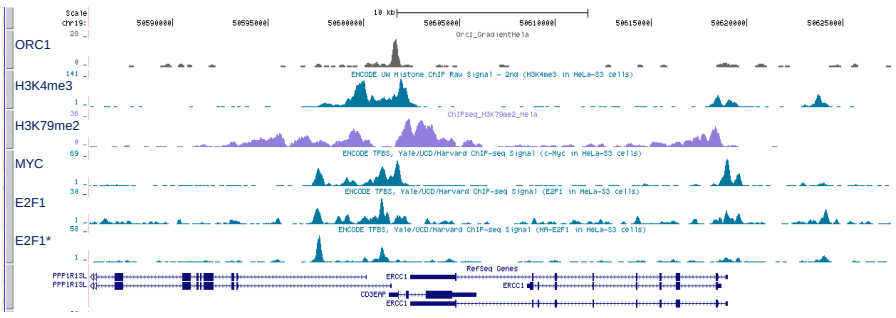


Figure 1: Genome browser show different peak shapes

in a single ChIP-seq. Recently, other aspects of shape besides intensity have been used in peak calling [6, 10, 8], peak ranking [17] and ChIP-seq differential analysis [15]. All these novel techniques show that peak shape information can improve peak detection, but they do not answer the question of whether peak shape includes additional biological properties.

3 Activity description

In the context of GenData, our research focus on applying different techniques for clustering ChIP-seq peaks, from multivariate analysis on shape indices, to functional data analysis treating peaks as curves. Afterward, the obtained clusters are biologically characterized using several statistical and bioinformatics methodologies to relate the different shapes to other genomic datasets, including ChIP-seq experiments for other binding proteins, RNA-seq experiments for differential expression and DNase-seq experiments for the structure of the chromatin.

3.1 Multivariate approach: Shape Index Clustering for ChIP-seq peaks

In the first part of the work, we study ChIP-seq peaks through multivariate statistical techniques, by selecting five indices that summarize peak shape [4, 3, 5]:

1. The maximum height of the peak;
2. The area under the step function that defines the peak;
3. The full width at half maximum, that is the width of the peak at half of maximum height;
4. The number of local peaks, namely the local maxima of the peak after smoothing;
5. The shape index M, divided by the maximum height of the peak.

The shape index M is a measure of the complexity of the peak, robust to noise. In particular, the shape index M is the number of edges in a maximal matching for the tree constructed starting from the peak, as suggested in [6]. We then use a clustering algorithm on these indices in order to assess whether the peaks can be divided into groups according to their shapes. This central part of our analysis pipeline is available online as a command line R script [2]. The resulting clusters are then characterized in terms of gene ontology enrichment analysis, motif analysis and in terms of co-occurrences with other binding proteins, as well as histone modifications and open chromatin maps. Finally, we relate the clusters to the differential expression results obtained by RNA-seq experiments. In this steps we employ bioinformatics tools such as GREAT [9] and MEME-ChIP [7], in addition to a range of statistical methodologies, from hypothesis test to random forests and multiple correspondence analysis.

The application of this analysis pipeline to publicly available ChIP-seq for the erythroid transcription factor GATA-1 in K562 cells leads to the identification of three clusters, suggesting the existence of statistically significant differences in peak shape inside a single ChIP-seq. Such differences are related to motif occurrences and to peculiar locations (for instance, one cluster is significantly associated to promoter regions). Moreover, peak shape appears to be associated with the presence of a putative protein complex and with different types of gene regulations [5]. Applying the shape index clustering to GATA-1 in other cell types, as well as to other binding proteins in K562, we discover that peak shape can vary depending on the different binding proteins under investigation. These results suggest that ChIP-seq peak shape carries much information and its deep study can provide many biological insights.

3.2 Functional approach

A second approach we deal with is the analysis of the peaks as curves: the shape problem is embedded in the functional data analysis framework. In this way peak shape is studied in a more natural and direct way, without introducing any summarization, so that the method becomes general enough to be applied to different pipelines associated to various transcription factor; no prior information on the general shape of the peaks is needed [1].

We adapt the k-mean alignment algorithm [14] to the specific case of ChIP-seq data. In particular some preprocessing steps are required to deal with the ChIP-seq curves: a removal of the background and a smoothing to guarantee a sufficient regularity and to compute derivatives.

Once the data are regularized, the algorithm can be applied and two more assumptions are required

- comparing two curves means computing their distance. We assume that a proper measure of the distance of two ChIP-seq peaks is the L^2 norm on the derivatives.
$$\rho(x_1, x_2) = ||\nabla x_1 - \nabla x_2||_{L^2}$$
- computing the difference in shapes means compute the distance with the proper metric on the data once they are aligned [13]. If we deal with functions we have to consider that transformations on the abscissa (like shifts) can vary the distance between two peaks. We want to remove as much as possible the alignment problem, finding the transformation of the abscissa of the data which minimizes the distance between the peaks. Now this remaining distance is the true amplitude distance.

Once we have defined the proper distance and abscissa transformation, we apply the k-medoid alignment algorithm, an iterative procedure to cluster and align the peaks. We implement in C++ the algorithm to make the computation efficient also for huge datasets, as ChIP-seq datasets are. Then we apply this algorithm to different sets of data concerning the transcription factor Myc: Eμ-Myc, for the analysis of a spontaneous developing of tumors, and Myc-ER, from an in-vitro experiment. For each of them we analyze two different concentration level of Myc, low and high.

The algorithm leads us to define two clusters for each dataset and one of them is associated to regular, sharp peaks. Biological investigations on these grouped

data are performed through the comparison with datasets collected from different biological experiments like RNA-seq to analyze the regulatory contribution of the regions or DNase-seq to investigate the structure of the chromatin. Moreover some investigations on the characteristic Myc motifs are performed, both looking for the characteristic Ebox of Myc in the isolated regions and analyzing with the DREME tool the regions to identify the motifs present. We detect that also the Myc Ebox can be related to the shape of peaks.

References

[1] Bruno Amati, Morelli Marco J, Alice CL Parodi, Laura M Sangalli, Piercesare Secchi, and Simone Vantini. Functional peak shape analysis of myc chip-seq data. Work in progress.

[2] Marzia A Cremona. SIC-ChIP software. `http://cgsb.genomics.iit.it/wiki/projects/SIC-ChIP`, 2015.

[3] Marzia A Cremona, Pier Giuseppe Pelicci, Laura Riva, Laura M Sangalli, Piercesare Secchi, and Simone Vantini. Cluster analysis on shape indices for chip-seq data. In *SIS2014*, 2014.

[4] Marzia A Cremona, Laura Riva, Laura M Sangalli, Piercesare Secchi, and Simone Vantini. Clustering chip-seq data using peak shape. In *SCo2013*, 2013.

[5] Marzia A Cremona, Laura M Sangalli, Simone Vantini, Gaetano I Dellino, Pier Giuseppe Pelicci, Piercesare Secchi, and Laura Riva. Peak shape clustering reveals biological insights. Manuscript submitted for publication, 2015.

[6] Valerie Hower, Steven N Evans, and Lior Pachter. Shape-based peak identification for ChIP-seq. *BMC bioinformatics*, 12(1):15, 2011.

[7] Philip Machanick and Timothy L. Bailey. MEME-ChIP: motif analysis of large DNA datasets. *Bioinformatics*, 27(12):1696–1697, 2011.

[8] Shaun Mahony, Matthew D Edwards, Esteban O Mazzoni, Richard I Sherwood, Akshay Kakumanu, Carolyn A Morrison, Hynek Wichterle, and David K Gifford. An integrated model of multiple-condition ChIP-seq data reveals predeterminants of Cdx2 binding. In *Research in Computational Molecular Biology*, pages 175–176. Springer, 2014.

[9] Cory Y. McLean, Dave Bristor, Michael Hiller, Shoa L. Clarke, Bruce T. Schaar, Craig B. Lowe, Aaron M. Wenger, and Gill Bejerano. GREAT improves functional interpretation of cis-regulatory regions. *Nature Biotechnology*, 28(5):495–501, 2010.

[10] Marco-Antonio Mendoza-Parra, Malgorzata Nowicka, Wouter Van Gool, and Hinrich Gronemeyer. Characterising ChIP-seq binding patterns by model-based peak shape deconvolution. *BMC genomics*, 14(1):834, 2013.

[11] Peter J. Park. ChIP-Seq: advantages and challenges of a maturing technology. *Nature Reviews Genetics*, 10(10):669–680, 2009.

[12] Shirley Pepke, Barbara Wold, and Ali Mortazavi. Computation for ChIP-seq and RNA-seq studies. *Nature methods*, 6:S22–S32, 2009.

[13] Silverman B.W. Ramsay J.O. *Functional Data Analysis*. Springer in Statistics, 2005.

[14] Laura M. Sangalli, Piercesare Secchi, Simone Vantini, and Valeria Vitelli. K-mean alignment for curve clustering. *Comput. Stat. Data Anal.*, 54(5):1219–1233, May 2010.

[15] Gabriele Schweikert, Botond Cseke, Thomas Clouaire, Adrian Bird, and Guido Sanguinetti. MMDiff: quantitative testing for shape changes in ChIP-seq data sets. *BMC genomics*, 14(1):826, 2013.

[16] Elizabeth G Wilbanks and Marc T Facciotti. Evaluation of algorithm performance in ChIP-seq peak detection. *PloS one*, 5(7):e11471, 2010.

[17] Hao Wu and Hongkai Ji. PolyaPeak: detecting transcription factor binding sites from ChIP-seq using peak shape information. *PloS one*, 9(3):e89694, 2014.

Section 1.2 — Discovering new gene functionalities from random perturbations of known gene ontological annotations

Giacomo Domeniconi[2], Marco Masseroli[1], Gianluca Moro[2], Pietro Pinoli[1]

[1]*DEIB - Politecnico di Milano*
[2]*DISI - Universita' degli Studi di Bologna*

1 Introduction

A common machine learning task often performed in several application domains, including bioinformatics, is the prediction of associations between items and features characterizing them. It well supports knowledge discovery, particularly when the considered features are described by means of controlled term, especially if such terms are related within ontologies. Indeed, several terminologies and ontologies exist and are used to describe structural and functional features of biomolecular entities, mainly genes and proteins. Among them, the Gene Ontology (GO) ([15]) is the most developed and considerable one; it is widely used to annotate, i.e. characterize, genes and proteins by associating them to its terms.

The GO consists of three sub-ontologies that overall include more than 40,000 controlled terms, which characterize species-independent Biological Processes (BP), Molecular Functions (MF) and Cellular Components (CC). These terms are hierarchically related, mainly through *"is a"* or *"part of"* relationships, within a Directed Acyclic Graph (DAG) and are designed to capture orthogonal features of genes and proteins. In the GO DAG, each node represents a GO term and each directed edge from a node a to a node b represents a relationship existing from a child term a to its parent term b.

Controlled annotations are very valuable for high-throughput and computationally intensive bioinformatics analyses. Yet, some of them are less reliable, or may even be incorrect, since automatically inferred without any human curation, which is highly time consuming. Furthermore, available biomolecular annotations are incomplete, since several gene and protein features of many organisms are still to be discovered and annotated. In this scenario, computational methods able to predict new annotations and estimate incorrectness of available ones are paramount, specially the methods that provide ranked lists of inferred annotations; they can, for instance, quicken the curation process by focusing it on the prioritized novel annotations ([23]).

Here, we first apply different supervised algorithms to discover new GO term annotations of different organism genes based on available GO annotations;

then, we benchmark them with an unsupervised method previously used to this purpose. Since in this context it is not available a labeled set of instances to train a supervised algorithm, we propose to assign labels to the originally unlabeled GO annotations based on a random perturbation of the annotation matrix that switches off some known annotations. In this way, we create a novel matrix with missing annotations; thus, we can train the model to recognize from this artificial matrix the real annotations. This allows applying supervised methods to available gene annotations and predicting new gene function annotations with better performance than the previously used unsupervised methods. We introduced this general approach in ([9]); in this paper we propose its extension with the use of real values, instead of binary ones, to represent the biomolecular annotations. The proposed data representation is exactly the same as that of a classification problem represented in a Vector Space Model; thus we can apply a weighting scheme to increase the effectiveness of the predictive model. We conducted experiments with several weighting measures to analyze the behavior of the proposed method with real-valued matrices. Despite multiple heterogeneous data could be leveraged to predict gene functions through sophisticated techniques previously proposed, with the methods here presented we confirm that simpler analytical frameworks, which use faster methods based only on available annotations, are as much effective and useful.

The rest of the paper is organized as follows. Section 2 describes the annotation datasets used in our experiments. Section 3 exposes the methods used to predict new annotations. Section 4 illustrates the performed experiments and reports their results, benchmarking them with those of a previous work. Section 5 reports an overview of other works about genomic functions prediction. Finally, in Section 6 we discuss our contribution and foresee possible future developments.

2 Genomic Datasets

In order to have easy access to subsequent versions of gene annotations to be used as input to the considered algorithms or to evaluate the results that they provide, we took advantage of the Genomic and Proteomic Data Warehouse (GPDW) ([4]). In GPDW several controlled terminologies and ontologies, which describe genes and gene products related features, functionalities and phenotypes, are stored together with their numerous annotations to genes and proteins of many organisms. These data are retrieved from several well known biomolecular databases. In the context of developing and testing machine learning methods on genomic annotations, GPDW is a valuable source since it is quarterly updated and old versions are kept stored. We leveraged this feature in our method evaluation by considering differed versions of the GO annotations of the genes of three organisms. In GPDW they are available with additional information, including an *evidence code* that describes how reliable the annotation is. We leveraged it by filtering out the less reliable annotations, i.e. those with *Inferred from Electronic Annotation* (*IEA*) evidence, from the datasets used for our evaluation. Table 1 gives a quantitative description of the considered annotations.

In GPDW, as in any other biomolecular database, only the most specific controlled annotations of each gene are stored. This is because, when the con-

Table 1: Quantitative characteristics of the nine considered annotation datasets. Figures refer to the sum of direct and indirect annotations not inferred from electronic annotation, i.e. without IEA evidence code.

	Gallus gallus			*Bos taurus*			*Danio rerio*		
	CC	MF	BP	CC	MF	BP	CC	MF	BP
# genes	260	309	275	497	540	512	430	699	1,528
# terms	123	134	610	207	226	1,023	131	261	1,176
# ann. (July 2009)	3,442	1,927	8,709	7,658	3,559	18,146	4,813	4,826	38,399
# ann. (May 2013)	3,968	2,507	10,827	9,878	5,723	24,735	5,496	6,735	58,040
Δ annotations between GPDW versions									
#Δ ann.	526	580	2,118	2,220	2,164	6,589	683	1,909	19,641
%Δ ann.	15.28	30.10	24.32	29.00	60.80	36.31	14.19	39.56	51.15

trolled terms used for the annotation are organized into an ontology, as for the GO, biologists are asked to annotate each gene only to the most specific ontology terms representing each of the gene features. In this way, when a gene is annotated to a term, it is implicitly indirectly annotated also to all the more generic terms, i.e. all the ancestors of the feature terms involved in its direct annotations. This is called *annotation unfolding*.

All direct and indirect annotations of a set of genes can be represented by using binary matrices. Let $\mathcal{G}$ be the set of genes of a certain organism and $\mathcal{T}$ a set of feature terms. We define the annotation matrix $\mathbf{A} \in \{0,1\}^{|\mathcal{G}|\times|\mathcal{T}|}$ as the matrix whose columns correspond to terms and rows to genes. For each gene $g \in \mathcal{G}$ and for each term $t \in \mathcal{T}$, the value of the $\mathbf{A}(g,t)$ entry of the annotation matrix is set according to the following rule:

$$\mathbf{A}(g,t) = \begin{cases} 1, & \text{if } g \text{ is annotated either to } t \\ & \text{or to any of } t \text{ descendants} \\ 0, & \text{otherwise} \end{cases} \quad (1)$$

Examples of two versions of these matrices are shown in Figure 1a and 1b, where $\mathbf{A}_1$ is an updated version of $\mathbf{A}_0$. Each GPDW update contains some number of new discovered annotations, namely new 1 in the matrix.

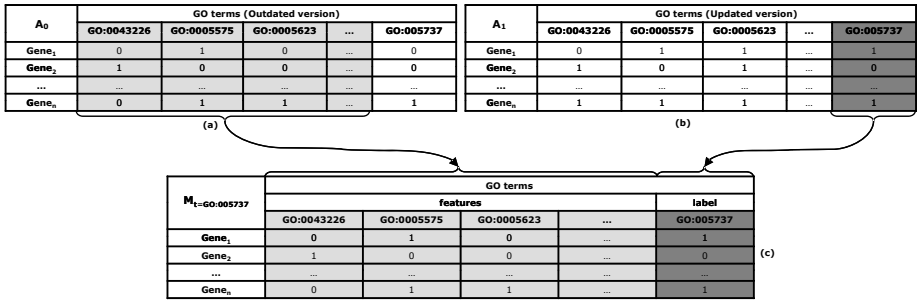
3 Annotation Discovery Methods

3.1 Data and Problem Modelling

The discovery of new genomic annotations can be modeled as a supervised problem in which, given a feature term t, you want to predict if a gene g is likely to be, or not to be, annotated to that term t, i.e. if the element $\mathbf{A}(g,t)$ of the annotation matrix is likely to be 1, or 0, basing on known annotations to other terms of the gene g, as in Figure 1c.

All the terms $t \in \mathcal{T}$ must be predicted, i.e. all the columns of the matrix, thus the problem can be modeled as a supervised multi-label classification, with the difference that we do not have a distinct set of features and labels, but we have a set of terms that are both classes and features. To address this problem, we use the most common approach in the literature, i.e. transform it into a set of binary classification problems, which can then be handled using single-class classifiers. Henceforth, for simplicity of exposition, we will refer to a single supervised task concerning the discovery of a new annotation of the gene g to the

Figure 1: Illustrative diagram of the data representation. The data set (c) is created with an older annotation version $\mathbf{A}_0$ (a) for the features and an uptdated version $\mathbf{A}_1$ (b) for the labels.

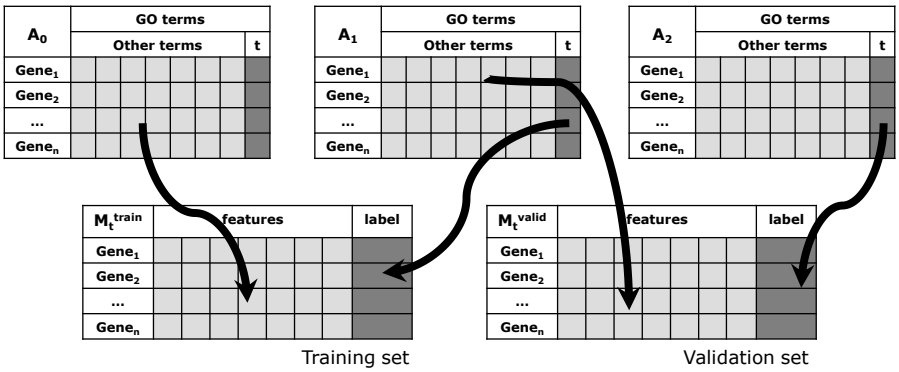


term t (for instance the term *GO:005737* in Figure 1), which is then repeated iteratively for all other genes and terms.

Let's now see how to assign a label to each instance of the data model. Given an annotation matrix, our proposal is to use as input a version of the matrix with less annotations (referred as outdated matrix, since it may resemble an outdated annotation dataset version); then, to derive from such input matrix the features of the data model, and consider as label of each record the presence or absence of an annotation in a more complete matrix (referred as updated matrix, since it may resemble a newer annotation dataset version). This representation is sketched in Figure 1. Given the feature term t considered for the prediction, called *class-term*, the representation of the data is created by taking as features, for each gene, all the annotations to all the other terms in an outdated version of the matrix $\mathbf{A}_0$, while the label is given by the value of the class-term in the updated version of the matrix $\mathbf{A}_1$. Henceforth, we refer to this representation matrix as $\mathbf{M}_\mathrm{t}$, where t is the class-term of the model.

This data representation is exactly the same as that of a supervised classification problem represented in a Vector Space Model. Thus, a classic supervised task could be envisaged by subdividing this new matrix $\mathbf{M}_\mathrm{t}$ horizontally and using a part of the genes to train the model and the remaining part to test it. In this domain, however, this approach is not applicable because it implies the availability of at least the part of the updated matrix to train the model, but new datasets are only released as a whole and not partially. Thus, the purpose is to predict which annotations are missing in the entire matrix, rather than on some part of it. The data representation matrix $\mathbf{M}_\mathrm{t}$ requires information from two different annotation dataset versions. Thus, since the aim is to make predictions over the entire dataset, to train the model we use a matrix $\mathbf{M}_\mathrm{t}^{\mathrm{train}}$ that is created by using the information from both the latest version currently available at training time, i.e. $\mathbf{A}_1$, and an older version of the matrix with missing annotations, i.e. $\mathbf{A}_0$. With this two different versions of the matrices, the training set is created by using the features derived from the outdated version $\mathbf{A}_0$ and the labels from the updated one $\mathbf{A}_1$. Then, the validation of the classification model has to be made by discovering new annotations, missing in the current state of the matrix. Therefore, the features regarding the current version $\mathbf{A}_1$ and labeled with the values of a future updated matrix $\mathbf{A}_2$ are used to create

Figure 2: Illustrative diagram of the dataset representation for the prediction model of the annotations to a term t. The training set ($\mathbf{M}_t^{\text{train}}$) is created with an older annotation version $\mathbf{A}_0$ for the features and the current annotation version $\mathbf{A}_1$ for the labels. Similarly, the validation set ($\mathbf{M}_t^{\text{valid}}$) is created using $\mathbf{A}_1$ and a future updated annotation matrix $\mathbf{A}_2$.



the validation matrix $\mathbf{M}_t^{\text{valid}}$. The training and validation data representation process is sketched in Figure 2.

3.2 Random Perturbation

The supervised problem modelling described in the previous subsection requires, at training time, two versions of the annotation matrix to create the supervised model, i.e. $\mathbf{A}_0$ and $\mathbf{A}_1$. However, biologists typically have available only the most updated version of the annotation matrix, not keeping stored the outdated versions for space reasons, given the large amount of data. Thus, with reference to Figure 1, there is available only one version of the matrix, i.e. only the current version $\mathbf{A}_1$, with which the training data representation $\mathbf{M}_t^{\text{train}}$ is created.

To overcome the problem just mentioned, we start from the observation that also the input matrix $\mathbf{A}_1$ contains missing annotations. Therefore, we could use only this matrix to obtain the representation $\mathbf{M}_t$, assuming $\mathbf{A}_0 = \mathbf{A}_1$. However, the classification model will have to discover new gene-term annotations starting from an outdated matrix; thus, it will be more effective if it is trained with a training set in which the features are taken from an outdated matrix, with a greater number of missing annotations than the matrix version from which the labels of the instances are obtained. If we consider that the annotations of genes to features are discovered by teams of biologists that work independently from each other, a reasonable hypothesis is that the new annotations discovered by the entire scientific community, on the whole, do not have any kind of bond or rule. This should be equivalent to a random process of discovery of new annotations. Such considerations led to our thesis that new gene annotations can be better discovered by artificially increasing the number of missing annotations in the input matrix $\mathbf{A}_0$. Since, as mentioned, usually only the input matrix $\mathbf{A}_1$ is available, this can be achieved by randomly deleting known annotations in the matrix $\mathbf{A}_1$ to obtain a new matrix $\mathbf{A}_0$ artificially perturbed.

Thus, to get the data to train the classification model, we propose to ran-

domly perturb the matrix $\mathbf{A}_1$ to create a new matrix $\mathbf{A}_0$, in which some annotations are eliminated with a probability p. In this way we obtain the matrix $\mathbf{A}_0 = random\_perturbation(\mathbf{A}_1, p)$. Formally, for each gene g and term t, the perturbation is done as follows:

$$\mathbf{A}_0(g,t) = \begin{cases} 0 & \text{if } \mathbf{A}_1(g,t) = 1 \wedge random \leqslant p \\ 1 & \text{if } \mathbf{A}_1(g,t) = 1 \wedge random > p \\ 0 & \text{if } \mathbf{A}_1(g,t) = 0 \end{cases} \tag{2}$$

Once the perturbed matrix $\mathbf{A}_0$ is generated, to ensure its correctness with respect to the unfolding of the annotations, the matrix $\mathbf{A}_0$ is corrected by switching to 0 also all the annotations to the same gene of all the descendants of the ontological terms with modified gene annotation; we call this process *perturbation unfolding*. It is important to note that, depending on this correction, the percentage of the actual modified annotations of the matrix $\mathbf{A}_0$ will hence be greater than the percentage derived from p. The overall data representation process is the same as that shown in Figure 2, with the difference that the matrix $\mathbf{A}_0$ is created by perturbing randomly $\mathbf{A}_1$.

Considering the annotation unfolding in the GO, in order to avoid trivial predictions (i.e. 1 if a child is 1), in the set of features of the dataset $\mathbf{M}_\mathrm{t}$ all the descendants or ancestors of the term t are not taken into consideration. Once created the training matrix $\mathbf{M}_\mathrm{t}^{\mathrm{train}}$, we can use any supervised algorithm, capable of returning a probability distribution, to train the prediction model and then validate it with $\mathbf{M}_\mathrm{t}^{\mathrm{valid}}$. The predicition model provides a probability distribution $pd(g,t)$, called *likelihood*, concerning the presence of an annotation of the gene g to the term t. To provide predictions of only new annotations, only those annotations that were missing in the outdated version of the matrix are taken into account. The supervised process described above is repeated for all the terms $t \in \mathcal{T}$, giving as final output a list of predictions of new gene annotations ordered according to their likelihood; the illustrated annotation discovery workflow is sketched in Figure 3.

3.3 Likelihood Correction

As shown above, the output of the supervised model is a list of predicted annotations, each one with a likelihood degree. According to the hierarchical structure of GO, when a gene is annotated to an ontological term, it must be also annotated to all the ancestors of that term; this constraint is also known as *True Path Rule* ([32]). The supervised classifier, however, provides a likelihood for each gene annotation regardless of the predictions of the annotation of other GO terms to the same gene. This can result in possible cases of anomalies in which a gene shall be annotated to a term, but not to one or more of its ancestor terms, thus violating the True Path Rule. To obtain a likelihood that takes into account the hierarchy of the terms, once obtained the likelihood of each gene-term association, we proceed as follows:

1. For each novel gene-term annotation, to the probability given by the model we add the average of all the probabilities of the novel annotations of the gene to all the ancestors of the term. Note that, since the classification model provides in output a probability distribution ranging between 0

and 1, the hierarchical likelihood of each gene-term annotation shall be between 0 and 2, as follows:

$$pd^H(g,t) = \frac{\sum_{t_a \in ancestors(t)} pd(g,t_a)}{|ancestors(t)|} + pd(g,t) \quad (3)$$

2. Once the likelihood is made hierarchical, the correction of the possible anomalies regarding the True Path Rule is taken into account. An iterative process is carried on from the leaf terms to the root term of the hierarchy, upgrading each likelihood with the maximum likelihood value of the descendant terms, as follows:

$$l(g,t) = \max\{pd^H(g,t), \max_{t_c \in children(t)}\{pd^H(g,t_c)\}\} \quad (4)$$

In such a way, for each ontology term, the likelihood of a gene to be annotated to that term is always greater than or equal to the likelihood of the gene to be annotated to the term descendants.

3.4 Term Weighting

To increase the associative power of the gene-term matrix, we can give a weight to each existing association between genes and terms. This approach is similar to a classic term weighting in information retrieval ([21, 10]). Some weighting schemes have already been appliated to the prediction of genomic annotations ([27]); here we tested these and other different schemes from information retrieval and data classification realms, in order to weigh each known annotation in the representation matrix $\mathbf{M}_t$.

Fixed the *class-term* t_c of the representation matrix $\mathbf{M}_t$, for each feature term $t \in \mathcal{T} : t \neq t_c$ four elements (A, B, C and D) shall be defined in order to describe the term weighting schemes: A denotes the number of genes associated both with t_c and t; B denotes the number of genes associated with t_c and not with t; C denotes the number of genes associated with t and not with t_c and D denotes the number of genes associated neither with t_c or t. The sum of all the genes is denoted with $N = A + B + C + D = |\mathcal{G}|$.

Generally, a term weighting scheme is based on three factors: i) *term frequency factor* or local weight; ii) *collection frequency factor* or global weight; iii) *normalization factor*. The *term frequency factor* measures how important a feature, namely an ontology term, is to a certain gene. For each gene g and feature term t, it can be expressed as $tf(g,t) = 1 + M$, where M is the number of descendant terms of t which are associated with the gene g, both directly or indirectly (i.e. derived from the unfolding procedure). Considering that this tf is measured in a different way than in the standard information retrieval methods, we also consider the case in which the local factor consists in a simple binary value, regarding the presence or the absence of the association between t and g.

The *collection frequency factor* may be taken from virtually all proposed weighting schemes in information retrieval or data mining. An interpretation

of the common *idf* (inverse document frequency, [31]) is the igf (inverse gene frequency, [27]); for each term t its value is:

$$igf(t) = \ln \frac{|\mathcal{G}|}{|\text{genes annotated to } t|} \equiv \log \left(\frac{N}{A+C} \right) \tag{5}$$

The combination of these two measures, *term* and *collection frequency factors*, with the *normalization factor* generates several possible weighting schemes ([12, 11]). The contribution of seven of these generated schemes to the gene annotation prediction using an unsupervised method is studied in ([27]), where a substantial improvement is shown by using some of them. Out of all the schemes analyzed in that work, we focus on the combination regarding no-transformation in the tf factor and the maximum, cosine and none normalization (i.e. the schemes named NTM, NTC and NTN). In this work, we refer to these three schemes as $tf.igf^M$, $tf.igf^C$ and $tf.igf^N$, respectively. In our experiments, we also tested the igf alone, using only a binary value (*bin*) as term frequency factor; we refer to these schemes as igf^M, igf^C and igf^N. In addition, we tested also the two term frequency measures, i.e. tf and *bin*, alone, without a collection frequency factor.

Furthermore, we use some weighting schemes derived directly from the information retrieval ([8]), such as χ^2 or ig (information gain). These two schemes are calculated as follows:

$$\chi^2 = N \cdot \frac{(A \cdot D - B \cdot C)^2}{(A+C) \cdot (B+D) \cdot (A+B) \cdot (C+D)} \tag{6}$$

$$ig = -\frac{A+B}{N} \cdot \log \frac{A+B}{N} + \frac{A}{N} \cdot \log \frac{A}{A+C} + \frac{B}{N} \cdot \log \frac{B}{B+D} \tag{7}$$

Finally, we also tested the *relevance frequency* (rf) scheme proposed by ([20]) for the text classification task. The rf of a term t is based on the idea that the higher the concentration of genes associated both with t and the *class-term*, the greater the contribution of t in the prediction model.

$$rf = \log \left(2 + \frac{A}{\max(1, C)} \right) \tag{8}$$

3.5 Evaluation

In our experiments we tested the effectiveness of supervised models in discovering new functional gene annotations from the available annotations. Since the proposed method is applicable to any supervised algorithm that returns a probability distribution, we tested different types of existing algorithms in order to measure their effectiveness, in particular: *Support Vector Machines*, *nearest neighbors*, *decision trees*, *logistic regressions* and *naive bayes*, using the implementations provided by Weka[1] in its 3.7.9 version. In the experiments we tested the Weka classifiers: *IBk* (with $k = 3$), *J48*, *Logistic*, *Naive Bayes* (*NB*), *Random Forest* (*RF*) and *SMO*. For each algorithm we used the default parameter settings provided by Weka; no tuning of parameters has been done for time reasons.

[1] http://www.cs.waikato.ac.nz/ml/weka/.

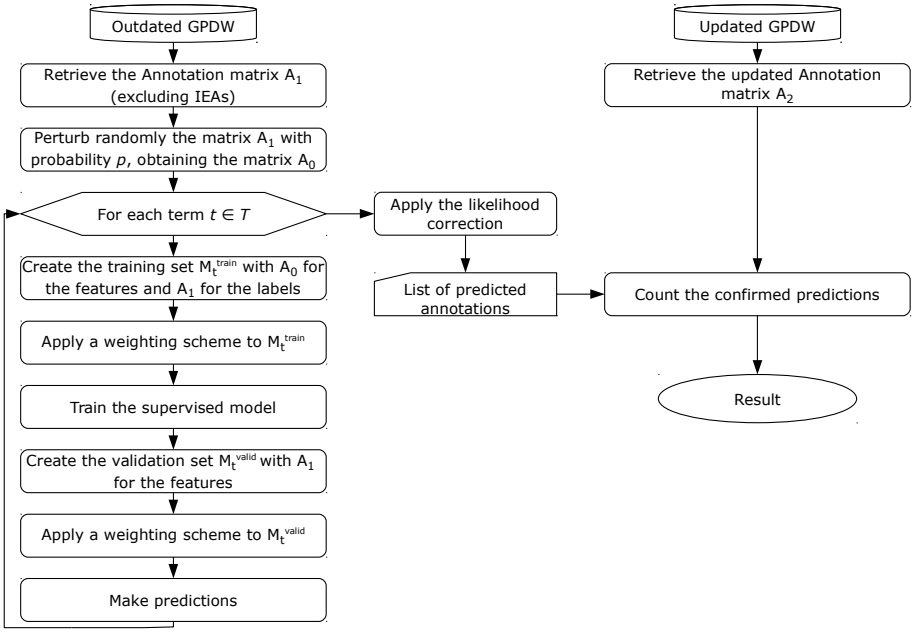


Figure 3: Workflow of the training and validation processes.

We measured the effectiveness of the predictions in the same way it was done in ([27]), in order to be able to directly compare our results with those in that work; the overall procedure was as follows.

1. We extracted the input annotations from an outdated version of the GPDW (July 2009), excluding from those annotations the ones less reliable, i.e. with IEA *evidence* code.

2. We randomly perturbed the unfolded annotation matrix to get a modified version of it, with some missing annotations.

3. We applied a weighting scheme on the representation matrix.

4. By running the prediction algorithm, we got a list of predicted annotations ordered by their confidence value (i.e. their corresponding likelihood $l(g,t)$).

5. We selected the top P predictions (we use $P = 250$) and we counted how many of these P predictions were found confirmed in the updated version of the GPDW (May 2013 version), regardless their evidence code.

6. For each experiment, steps 2, 3, 4 and 5 were repeated 10 times by varying the random seed. The effectiveness of each experiment was determined by averaging the counts obtained in all the experiment repetitions.

We depict the training and validation procedure workflows in Figure 3.

Table 2: Validation results of the predictions obtained by varying the supervised algorithm used to build the prediction model. The results show, for each of the nine considered datasets, the amount of the top 250 predicted gene annotations that have been found confirmed in the updated GPDW version. The setup of these experiments was done with random perturbation of the training matrix with probability $p = 0.05$ and the binary term weighting scheme. The first column (SIM) reports the results obtained in ([27]) with the SIM best configuration. Each result is reported as the average and corresponding standard deviation of 10 experiments repeated by changing the random perturbation seed. In bold the best result for each dataset.

Dataset	SIM	IBk	J48	Logistic	NB	RF	SMO
Gallus g. - BP	**86**	58.±20.2	47.±4.7	32.7±6.8	25.4±4.4	52.7±12.1	28.7±9.3
Gallus g. - MF	24	58.0±5.6	**79.7**±12.7	40.0±10.4	14.2±1.6	54.4±9.6	50.7±14.3
Gallus g. - CC	50	**81.5**±8.2	73.4±8.5	31.9±6.4	23.5±3.7	55.2±11.3	29.6±4.0
Bos t. - BP	55	48.9±6.8	49.7±5.1	37.0±6.5	28.4±4.2	**62.4**±7.6	31.2±4.6
Bos t. - MF	28	58.2±4.4	**58.8**±10.5	27.5±4.3	15.7±2.9	57.5±11.2	36.9±4.4
Bos t. - CC	91	**112**±9.7	94.3±9.8	38.2±5.3	8.2±2.0	93.7±10.4	48.4±6.8
Danio r. - BP	35	**70.9**±15.9	59.8±6.1	31.0±4.8	25.2±3.3	58.1±5.1	16.6±2.3
Danio r. - MF	35	77.5±10.3	75.8±7.1	54.4±11.0	41.2±2.7	**83.1**±9.6	79.7±8.7
Danio r. - CC	44	81.5±8.5	69.3±8.7	27.6±7.6	26.2±6.6	**92.3**±11.0	30.2±6.6
Total	447	**647.1**	608.8	320.3	207.9	609.4	352.0

4 Results

Table 2 shows the results obtained by varying the supervised algorithm used to train the prediction model, always using a fixed random perturbation probability $p = 0.05$ and without weighting the term associations, i.e. using the *binary* scheme. State of art methods ([27]) reach a total of 447 correct predictions; Table 2 shows that, with the proposed method, 3 out of 6 of the tested algorithms outperform them. Obtained results are excellent if we consider that they are obtained without any tuning of the algorithm parameters; therefore there is margin to improve them with an appropriate tuning. According to the results in Table 2, we can infer that using the standard parameterization provided by Weka, the algorithm that obtains the best results is IBk, with an improvement of 44.8% compared with the results of ([27]). IBk results also 6.2% better than *Random Forest* and 6.3% better than *J48*, the other two supervised algorithms that result better than the state of art.

Table 3 shows the results obtained by using the IBk algorithm with different term weighting schemes in the representation matrix. Differently from the work of Pinoli and colleagues ([27]), where weighting schemes improved their method, from our results we can infer that using weighting schemes does not lead to an improvement of the prediction effectiveness. The comparisons between the weighting schemes considered provided fluctuating results, but, in general, the best scheme appears to be the *binary* one. We can note that the schemes with the tf factor do not achieve good results, as well as, although with better performance, the information retrieval classical supervised measures, namely χ^2 and ig; the best weighting scheme results the rf, which however is, in total, slightly worst than the *binary* one.

The proposed method introduces a new parameter: the probability p of the random perturbation of the training matrix. Table 4 shows the results obtained by varying this probability p and using the best supervised algorithm

Table 3: Validation results of the predictions obtained using IBk as supervised algorithm, $p = 0.05$ as probability of perturbation and varying the term weighting scheme.

Dataset	BIN	tf	$tf.igf^N$	$tf.igf^C$	$tf.igf^M$	igf^N	igf^C	igf^M	χ^2	ig	rf
G.g.-BP	58.6	52.5	48.0	46.4	51.2	58.6	62.0	**66.6**	47.0	50.2	47.2
G.g.-MF	58	53.0	57.6	58.4	53.2	51.6	**64.4**	60.8	62.6	64.0	61.4
G.g.-CC	**81.5**	55.6	48.2	42.2	58.4	61.0	45.4	50.8	68.0	69.6	67.2
B.t.-BP	48.9	61.8	52.8	34.4	48.0	45.8	**72.6**	65.0	39.0	40.2	56.6
B.t.-MF	58.2	39.0	38.4	48.0	38.6	73.0	60.6	64.0	72.8	**75.4**	73.6
B.t.-CC	112.0	90.2	93.8	72.6	80.4	93.0	103.4	90.0	101.8	103.4	**121.4**
D.r.-BP	**70.9**	68.6	63.4	59.9	61.2	70.6	67.2	69.2	69.3	68.3	67.6
D.r.-MF	**77.5**	45.2	57.2	46.0	23.6	60.8	55.4	53.0	57.2	52.4	62.2
D.r.-CC	81.5	74.2	30.6	49.2	40.4	61.4	77.6	60.8	75.2	73.6	**82.6**
Total	**647.1**	540.1	490.0	457.1	455.0	575.8	608.6	580.2	592.9	597.1	639.8

from Table 2, namely *IBk*, and the *binary* weighting scheme. Table 4 results show that the best predictions are obtained with $p = 0.2$. Considering the *perturbation unfolding*, this p value leads to a perturbed matrix $\mathbf{A}_0$ with more than 20% of annotations less than in $\mathbf{A}_1$ (empirically they are about 30% less). Such percentage is very close to the average value of the variation of number of annotations between $\mathbf{A}_2$ and $\mathbf{A}_1$, i.e. 33.4%, notable in Table 1. Moreover, the probability p that gets the best results for each dataset seems to have a relationship with the dataset annotation variation between $\mathbf{A}_2$ and $\mathbf{A}_1$. This result leads to the conjectures that i) representing new annotations randomly leads to train a classifier able to predict the actual new annotations between two different annotation versions; ii) the more the amount of artificial missing annotations introduced in the training set is comparable to the actual missing annotations in the validation set, the more the predictions are accurate. Another result deducible from Table 4 is that using $p = 0$, namely the annotation matrix is not perturbed ($\mathbf{A}_0 = \mathbf{A}_1$), we get anyway good results, higher than those in ([27]). This is important since it allows to avoid the parameter p and the tuning of the system for any considered dataset when not top performance is required. For a graphical view, the results discussed are also shown in Figure 4, grouped by organism. Our approach outperforms the best accuracy achieved in ([27]) of 49.66%, in particular we obtain the highest improvement in large datasets, i.e. in the *Danio rerio* dataset there is an improvement of 104.56% of the correct annotations predicted.

5 Related Works

Different methods have been proposed to predict biomolecular annotations.

In ([19]), decision trees and Bayesian networks were suggested to learn patterns from available annotation profiles and predict new ones. Tao and col-

Table 4: Validation results of the predictions obtained using the IBk supervised algorithm, binary term weighting scheme and varying the probability p of random perturbation of the training matrix.

Dataset	$p = 0$	$p = 0.05$	$p = 0.10$	$p = 0.15$	$p = 0.20$	$p = 0.25$	$p = 0.30$
Gallus g. - BP	42	**58.6** ±20.2	54.8 ±16.2	51.3 ±12.5	55.9 ±10.4	50.2 ±10.2	47.4 ±9.7
Gallus g. - MF	50	58 ±5.6	61.8 ±11	59.5 ±13	58.3 ±10.2	**63.6** ±13.5	64.2 ±8.4
Gallus g. - CC	75	81.5 ±8.2	77.5 ±9.7	**82.2** ±8.1	78.1 ±7.5	73.3 ±13.2	78.8 ±12
Bos t. - BP	43	48.9 ±6.8	51.7 ±10.1	50.4 ±8.4	**53.1** ±9.6	52 ±12.5	52.2 ±15.4
Bos t. - MF	58	58.2 ±4.4	62.7 ±7.7	71.4 ±10.9	73 ±12.6	74.7 ±11.6	**77** ±13
Bos t. - CC	108	112 ±9.7	114.3 ±11	118.6 ±13	118.1 ±13	**119** ±13.1	116.7 ±22
Danio r. - BP	55	70.9 ±15.9	70.6 ±16.5	74.8 ±13.9	85.7 ±25.6	83.1 ±16.3	**90.6** ±19.4
Danio r. - MF	76	**77.5** ±10.3	72.5 ±7.1	67.7 ±10.1	62 ±7.6	58.4 ±8.7	51.4 ±15.1
Danio r. - CC	79	81.5 ±8.5	84.7 ±8.7	**90.7** ±10	85.6 ±13.5	83.3 ±14.5	75.8 ±19.9
Total	586	647.1	650.6	666.6	**669.8**	661.6	654.1

leagues ([33]) improved the results by using a k-nearest neighbour (k-NN) classifier to make a gene inherit the annotations that are common among its nearest neighbour genes in a gene network, where distance between genes is based on the semantic similarity of the GO terms used to annotate them.

Novel gene annotations can also be inferred based on multiple data sources. In ([1]), gene expression levels from microarray experiments are used to train a Support Vector Machine (SVM) classifier for each gene annotation to a GO term; consistency among predicted annotation terms is then enforced through a Bayesian network mapped onto the GO structure. Conversely, in ([30]) and ([24]), the authors took advantage of textual information by mining the literature and extracting keywords that are then mapped to GO concepts. This approach has the disadvantage to require a preparatory data integration step in order to be performed; this both adds complexity to the framework and reduces

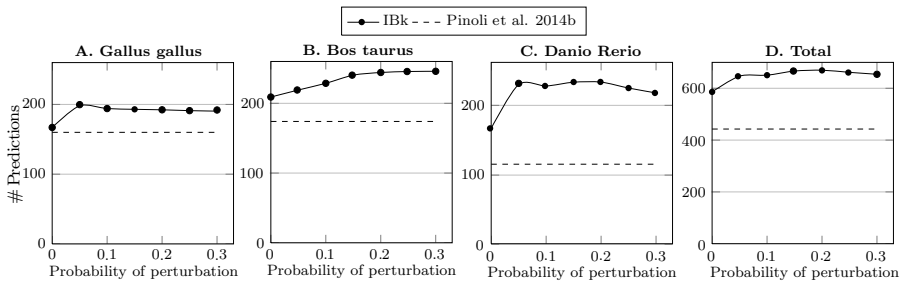


Figure 4: Validation results of the predictions obtained by varying the probability of perturbation p, compared with those obtained in ([27]). The results show, for each organism in the A, B and C charts, the sum of the predicted annotations that have been found confirmed in the updated GPDW version of the three GO ontologies. The chart D shows the total values for all the organism.

its flexibility.

In ([18]) and ([13]), Khatri and colleagues suggested a prediction algorithm based on the Singular Value Decomposition (SVD) method of the gene-to-term annotation matrix, which is implicitly derived from the count of co-occurrences between pairs of terms in the available annotation dataset. This prediction method based on basic linear algebra was then extended in ([7]), by incorporating gene clustering based on gene functional similarity computed on Gene Ontology annotations. It was further enhanced by automatically choosing its main parameters, including the SVD truncation level, based on the evaluated data ([6]). The SVD has also been used with annotation co-occurrence weights based on gene-term frequencies ([12]), ([27]). Being based on simple matrix decomposition operations, these methods are independent with regards to both the chosen organism and function term vocabulary involved in the annotation set. Anyway, obtained results highlighted their poor performance in terms of accuracy.

Other methods based on evaluation of co-occurrences exist; in particular the ones related to Latent Semantic Indexing (LSI) ([14]), which have been originally proposed in Natural Language Processing. Among them, the probabilistic Latent Semantic Analysis (pLSA) ([17]) gives a well defined distribution of sets of terms as an approximation of the co-occurrence matrix. It uses the *latent* model of a set of terms to increase robustness of annotation prediction results. In ([22]) and ([28]), pLSA proved to provide general improvements with respect to the truncated SVD method of Khatri and colleagues ([18]).

In bioinformatics, *topic modeling* has been leveraged also by using the Latent Dirichlet Allocation (LDA) algorithm ([3]). In ([2]) and ([25]), LDA was used to subdivide expression microarray data into clusters. Very recently, Pinoli et al. ([26]) took advantage of the LDA algorithm, together with the Gibbs sampling ([16]), ([5]), ([29]), to predict gene annotations to GO terms. These methods strongly overcome the ones based on linear algebra, but the complexity of the underlying model and the slowness of the training algorithms make these approaches ill-suited when the size of the dataset grows.

In summary, previously proposed methods for biomolecular annotation prediction either are general and flexible, but provide only limited accuracy mainly due to the simple model used, or improve prediction performance by either leveraging a complex integrative analytical framework, which often is difficult and time consuming to be properly set up, or adopting a more complex model, which in turn significantly slows the prediction process in particular in the usual case of many data to be evaluated.

6 Conclusions

In this paper we propose a method to discover new GO term annotations for genes of different organisms, based on available GO annotations of these genes.

Our idea is to train a model to recognize the presence of novel gene annotations using the obsolete annotation profile of the gene, labeling each term of an outdated annotation profile of a gene with a label taken from an updated version of it. This approach requires two different versions of the annotation matrix to build the training data representation. However, biologists typically have available only the most updated version of the gene annotation matrix. Given this

constrain, we have proposed a method to overcome this lack; creating a different annotation matrix, representing an older version of the input one, by perturbing the known annotation matrix in order to randomly remove some of its annotations. This allows the use of supervised algorithms even in datasets without labels and the comparison with results obtained by unsupervised methods on the same originally unlabeled datasets.

Obtained results are very encouraging, since they show a great improvement compared with unsupervised techniques. Furthermore, these results could be even better with an appropriate tuning of the parameters of the supervised algorithms used; our purpose is to thoroughly investigate this aspect in the future. The extension, using weighted real values to represent gene-term associations, did not yield better results with respect to the binary value representation. Thus, we found that the computationally simpler scheme, namely the binary scheme, achieves results generally better than other more complex schemes.

From the obtained results we can see that, by increasing the number of perturbed (removed) annotations, the results improve, reaching a peak when the number of artificially missing annotations in the training set is comparable to the number of those in the validation set, i.e. when the variety of missing annotations has been fully mapped in the training set. Furthermore, it is noteworthy also the case where we do not perturb the training matrix, avoiding the tuning of the parameter p, which gets anyway good results.

We plan to further verify the effectiveness of the proposed approach, also considering the possibility to train the prediction model using genes and annotations relating to a different organism with respect the target one.

References

[1] Zafer Barutcuoglu, Robert E Schapire, and Olga G Troyanskaya. Hierarchical multi-label prediction of gene function. *Bioinformatics*, 22(7):830–836, 2006.

[2] Manuele Bicego, Pietro Lovato, Barbara Oliboni, and Alessandro Perina. Expression microarray classification using topic models. In *Proceedings of the 2010 ACM Symposium on Applied Computing*, pages 1516–1520. ACM, 2010.

[3] David M Blei, Andrew Y Ng, and Michael I Jordan. Latent dirichlet allocation. *the Journal of machine Learning research*, 3:993–1022, 2003.

[4] Arif Canakoglu, Giorgio Ghisalberti, and Marco Masseroli. Integration of biomolecular interaction data in a genomic and proteomic data warehouse to support biomedical knowledge discovery. In *Computational Intelligence Methods for Bioinformatics and Biostatistics*, pages 112–126. Springer, 2012.

[5] George Casella and Edward I George. Explaining the gibbs sampler. *The American Statistician*, 46(3):167–174, 1992.

[6] Davide Chicco and Marco Masseroli. A discrete optimization approach for svd best truncation choice based on roc curves. In *Bioinformatics and Bio-*

engineering (BIBE), 2013 IEEE 13th International Conference on, pages 1–4. IEEE, 2013.

[7] Davide Chicco, Marco Tagliasacchi, and Marco Masseroli. Genomic annotation prediction based on integrated information. In *Computational Intelligence Methods for Bioinformatics and Biostatistics*, pages 238–252. Springer, 2012.

[8] Franca Debole and Fabrizio Sebastiani. Supervised term weighting for automated text categorization. In *In Proceedings of SAC-03, 18th ACM Symposium on Applied Computing*, pages 784–788. ACM Press, 2003.

[9] Giacomo Domeniconi, Marco Masseroli, Gianluca Moro, and Pietro Pinoli. Discovering new gene functionalities from random perturbations of known gene ontological annotations. In *International Conference on Knowledge Discovery and Information Retrieval (KDIR 2014)*, 2014.

[10] Giacomo Domeniconi, Gianluca Moro, Roberto Pasolini, and Claudio Sartori. Cross-domain text classification through iterative refining of target categories representations. In *Proceedings of the 6th International Conference on Knowledge Discovery and Information Retrieval*, 2014.

[11] Giacomo Domeniconi, Gianluca Moro, Roberto Pasolini, and Claudio Sartori. Iterative refining of category profiles for nearest centroid cross-domain text classification. In *To appear in: Knowledge Discovery, Knowlege Engineering and Knowledge Management*. Springer, 2015.

[12] Bogdan Done, Purvesh Khatri, Arina Done, and Sorin Draghici. Semantic analysis of genome annotations using weighting schemes. In *Computational Intelligence and Bioinformatics and Computational Biology, 2007. CIBCB'07. IEEE Symposium on*, pages 212–218. IET, 2007.

[13] Bogdan Done, Purvesh Khatri, Arina Done, and Sorin Draghici. Predicting novel human gene ontology annotations using semantic analysis. *IEEE/ACM Transactions on Computational Biology and Bioinformatics (TCBB)*, 7(1):91–99, 2010.

[14] Susan T Dumais, George W Furnas, Thomas K Landauer, Scott Deerwester, and Richard Harshman. Using latent semantic analysis to improve access to textual information. In *Proceedings of the SIGCHI conference on Human factors in computing systems*, pages 281–285. ACM, 1988.

[15] GO Consortium et al. Creating the gene ontology resource: design and implementation. *Genome research*, 11(8):1425–1433, 2001.

[16] Thomas Griffiths. Gibbs sampling in the generative model of latent dirichlet allocation. *Standford University*, 518(11):1–3, 2002.

[17] Thomas Hofmann. Probabilistic latent semantic indexing. In *Proceedings of the 22nd annual international ACM SIGIR conference on Research and development in information retrieval*, pages 50–57. ACM, 1999.

[18] Purvesh Khatri, Bogdan Done, Archana Rao, Arina Done, and Sorin Draghici. A semantic analysis of the annotations of the human genome. *Bioinformatics*, 21(16):3416–3421, 2005.

[19] Oliver D King, Rebecca E Foulger, Selina S Dwight, James V White, and Frederick P Roth. Predicting gene function from patterns of annotation. *Genome research*, 13(5):896–904, 2003.

[20] Man Lan, Chew Lim Tan, Jian Su, and Yue Lu. Supervised and traditional term weighting methods for automatic text categorization. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 31(4):721–735, 2009.

[21] Christopher D Manning, Prabhakar Raghavan, and Hinrich Schütze. *Introduction to information retrieval*, volume 1. Cambridge university press Cambridge, 2008.

[22] Marco Masseroli, Davide Chicco, and Pietro Pinoli. Probabilistic latent semantic analysis for prediction of gene ontology annotations. In *Neural Networks (IJCNN), The 2012 International Joint Conference on*, pages 1–8. IEEE, 2012.

[23] G. Pandey, V. Kumar, and M. Steinbach. Computational approaches for protein function prediction: A survey. Technical report, Minneapolis, MN, USA, 2006.

[24] Antonio J Pérez, Carolina Perez-Iratxeta, Peer Bork, Guillermo Thode, and Miguel A Andrade. Gene annotation from scientific literature using mappings between keyword systems. *Bioinformatics*, 20(13):2084–2091, 2004.

[25] Alessandro Perina, Pietro Lovato, Vittorio Murino, and Manuele Bicego. Biologically-aware latent dirichlet allocation (balda) for the classification of expression microarray. In *Pattern Recognition in Bioinformatics*, pages 230–241. Springer, 2010.

[26] P. Pinoli, D. Chicco, and M. Masseroli. Latent dirichlet allocation based on gibbs sampling for gene function prediction. In *Proceedings of the International Conference on Computational Intelligence in Bioinformatics and Computational Biology*, pages 1–7. IEEE Computer Society, 2014.

[27] P. Pinoli, D. Chicco, and M. Masseroli. Weighting scheme methods for enhanced genome annotation prediction. In *Computational Intelligence Methods for Bioinformatics and Biostatistics (CIBB), 2013 10th International Meeting on*, pages 76–89. LNBI, Springer International Publishing, 2014.

[28] Pietro Pinoli, Davide Chicco, and Marco Masseroli. Enhanced probabilistic latent semantic analysis with weighting schemes to predict genomic annotations. In *Bioinformatics and Bioengineering (BIBE), 2013 IEEE 13th International Conference on*, pages 1–4. IEEE, 2013.

[29] Ian Porteous, David Newman, Alexander Ihler, Arthur Asuncion, Padhraic Smyth, and Max Welling. Fast collapsed gibbs sampling for latent dirichlet allocation. In *Proceedings of the 14th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 569–577. ACM, 2008.

[30] Soumya Raychaudhuri, Jeffrey T Chang, Patrick D Sutphin, and Russ B Altman. Associating genes with gene ontology codes using a maximum entropy analysis of biomedical literature. *Genome Research*, 12(1):203–214, 2002.

[31] Karen Sparck Jones. Document retrieval systems. chapter A statistical interpretation of term specificity and its application in retrieval, pages 132–142. Taylor Graham Publishing, London, UK, UK, 1988.

[32] Junko Tanoue, Masatoshi Yoshikawa, and Shunsuke Uemura. The gen-earound go viewer. *Bioinformatics*, 18(12):1705–1706, 2002.

[33] Ying Tao, Lee Sam, Jianrong Li, Carol Friedman, and Yves A Lussier. Information theory applied to the sparse gene ontology annotation network to predict novel gene function. *Bioinformatics*, 23(13):i529–i538, 2007.

Section 2.1 — An Approach to On-Demand ETL for the GOLAM Framework

Lorenzo Baldacci, Matteo Golfarelli, Simone Graziani, and Stefano Rizzi

DISI - University of Bologna

1 Introduction

Data warehouses (DWs) have been used for almost two decades in company settings to store information useful for decision making. Most of this information is typically gathered from corporate operational databases using an ETL (Extract-Transform-Load) process that extracts relevant data, transforms them into multidimensional form, and loads them into the DW, to be later analyzed by means of reporting and OLAP tools. Traditionally, ETL is performed on a periodic basis by fast bulk-loading techniques during a time window in which the DW is in a quiescent state, i.e., is not queried by end-users. This means that, at query time, the information available has already been loaded in its entirety into the DW.

Over the last few years, the scope of the analyses carried out by decision makers has been progressively enlarging to encompass a relevant quantity of data that are not necessarily stored in corporate databases. For instance this is the case for social business intelligence, in which relevant data are fetched from the web in the form of user-generated content made available in forums, blogs, social networks, and the like; or it is the case for scientific applications where huge datasets (e.g., containing genomic data) are shared worldwide and publicly available for research purposes. In these cases, loading *all* available data into the DW at ETL time may be either inconvenient (because data are supplied from a provider for a fee) or unfeasible (because of their size); on the other hand, directly launching each analysis query on source data would not enable data reuse, thus leading to poor performance and high costs.

The alternative investigated in this report is that of incrementally fetching and storing data *on-demand*, i.e., as they are needed during the analysis process. The main application scenarios for this approach are summarized below:

- When source data are supplied for a fee by one or more data providers, on-demand ETL enables exactly extracting the data that are actually necessary and reusing those data by several users at different times, thus reducing the overall costs.
- In scientific settings, the amount of possibly useful data shared by all the specialized repositories available worldwide can be intractable [3], so

on-demand ETL can effectively cut the bootstrapping time by allowing incremental extraction and reuse of data.

- In a *situational business intelligence* scenario, the decision process is empowered with open/linked data that have a narrow focus on a specific business problem and, typically, a short lifespan for a small group of users [2]; in this context, on-demand ETL is a key for fetching, at each time, the relevant data needed for each specific analysis.

- More and more companies are using so-called *data lakes* to "park" huge volumes of data in their native format until they are needed; in this Big Data setting, the overall size of data makes a traditional batch ETL approach unfeasible.

1.1 Motivating Example

The GenData 2020 project aims at managing genomic data through an integrated data model, expressing the various features that are embedded in the produced bio-molecular data and in their correlated phenotypic data. This goal is achieved by enabling viewing, searching, querying, and analyzing over a worldwide-available collection of shared genomic data. One of the analysis services envisioned in this context is the multi-resolution analysis of the mappings between *regions* (i.e., segments of the genome) and *samples* (i.e., sets of regions and correlated metadata resulting from an experiment), achieved in the GOLAM framework [3]. As sketched in Figure 1, these mappings are computed by issuing a query in an ad-hoc language called GMQL (GenoMetric Query Language) against some repositories of genomic data such as ENCODE.[1] The query output, called *genome space*, comes in the form of a set of GTF (*Gene Transfer Format*, `http://genome.ucsc.edu`) files and related metadata, and due to its huge size is stored using the Hadoop platform.

OLAP-like queries are a valuable tool for biologists [3] because they enable multi-resolution analyses based on standard hierarchies of concepts; besides, they are preferred to traditional browser-based approaches because they enable a far more flexible and user-driven navigation of data. Unfortunately, the genome space generated by most biologically-relevant queries is too large to enable a traditional ETL process to load it into a multidimensional cube on an RDBMS for OLAP analyses. This is where on-demand ETL comes into play. When a user formulates an OLAP query q in GOLAM, the front-end translates q into MDX [1] and sends it to the on-demand ETL component for processing. If all the multidimensional data (called *facts* from now on, and including both the actual mappings and the correlated dimensional data about the involved regions and samples) necessary to answer q are already present in the mapping cube (i.e., they have been previously loaded), they are sent to the front-end and shown to the user. Otherwise, the genome space is accessed via FTP to fetch all the missing data, that are then transformed and loaded onto the mapping cube, so that q can be answered. Of course, from time to time, some facts used for

[1] ENCODE, the *Encyclopedia of DNA Elements*, is a public repository (accessible via FTP) created and maintained by the US National Human Genome Research Institute to identify and describe the regions of the 3 billions base-pair human genome that are important for different kinds of functions [5].

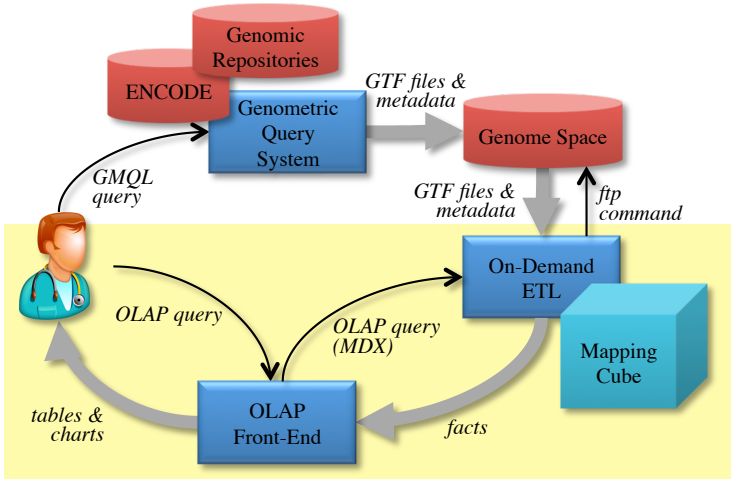


Figure 1: Analysis of genomic mappings in the GenData 2020 framework

past queries must be dropped from the cube to make room for the facts needed for new queries.

1.2 Contribution and Outline

In this report we present *QETL* (Query-Extract-Transform-Load), an approach to on-demand ETL for feeding a ROLAP cube in scenarios where batch-loading the whole cube *before* query-time is either unfeasible (e.g., for space reasons) or inconvenient (e.g., for time or cost reasons). In QETL, facts are incrementally fetched from the source data provider and loaded into the cube only when they are needed to answer some OLAP query, to be possibly later dropped when they can be considered obsolete or when some free space is needed to load other facts. We remark that, in this context, with the term fact we mean not only the core multidimensional data (i.e., the measure values for a given multidimensional coordinate), but also the correlated dimensional data (i.e., the coordinate values and the corresponding hierarchy values). This means that, with reference to a classical star schema implementation, QETL works by loading/dropping tuples of fact tables and dimension tables at the same time.

The reason for storing the loaded facts into the cube (rather than simply using them to answer the OLAP query on-the-fly) is twofold. In scenarios where several users are concurrently analyzing the same cube, this caching-like mechanism encourages data reuse and cuts the cost for re-fetching the same facts twice or more. On the other hand, even when facts are mostly accessed by a single user (as in the genomic example, because each user normally builds her own mappings using custom GMQL queries), caching the facts extracted is convenient because the queries expressed during an OLAP sessions normally tend to be contiguous in terms of the facts they require [6].

Overall, the main contributions of this report are:

1. We present a case and an architectural framework for on-demand ETL.

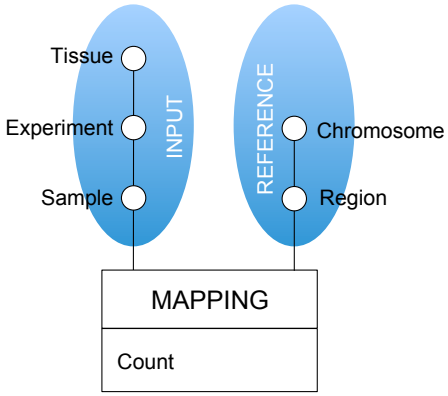


Figure 2: The MAPPING schema (the two All top levels are not shown)

2. We introduce an abstraction called *dice* for compactly representing the facts available in the cube at each time, and we show how dice can be used to efficiently determine the facts missing to answer an OLAP query.

3. We present a heuristic algorithm that, given the missing facts and considering the features of the source data provider, finds the cheapest set of extractions that the ETL can carry out to fetch the data required.

2 Formal Background

In this section we introduce a basic formal setting to manipulate multidimensional data. For simplicity here we will consider hierarchies without branches, i.e., consisting of chains of levels, and facts with a single measure.

Definition 1 (Multidimensional Schema) *An* n-dimensional schema *(or, briefly, a* schema*)* $\mathcal{M}$ *is a couple of*

- *a finite set of disjoint* hierarchies, $\{h_1, \ldots, h_n\}$, *each characterized by a set* L_i *of levels and a* roll-up *total order* $\prec_{h_i}$ *of* L_i. *We will use superscripts to denote the hierarchy each level belongs to, so* $l^i \in L_i$. *Each level* l^i *is defined over a categorical domain of* members, $Dom(l^i)$*; the domain of the top level* l^i_{all} *of each hierarchy has a single* All *member. For notational simplicity, we will order the indexes of the levels in each hierarchy according to their roll-up order:* $l^i_1 \prec_{h_i} l^i_2 \prec_{h_i} \ldots \prec_{h_i} l^i_{all}$.

- *a family of* roll-up functions *including a function* $RollUp^{l^i_k} : Dom(l^i_1) \rightarrow Dom(l^i_k)$ *for each level* l^i_k.

Example 1 *As a working example we will use a simplified form, shown in Figure 2, of the* MAPPING *schema adopted in the GOLAM framework of the GenData 2020 project for OLAP analysis of mappings. The schema includes two hierarchies, namely* INPUT *and* REFERENCE. *Within* INPUT *it is* Sample $\prec_{\mathsf{INPUT}}$ Experiment $\prec_{\mathsf{INPUT}}$ Tissue, $RollUp^{\mathsf{Experiment}}(\mathrm{S21}) = \mathrm{Exp1}$, *and* $RollUp^{\mathsf{Tissue}}(\mathrm{S21}) = \mathrm{Spleen}$ *(see Figure 3).*

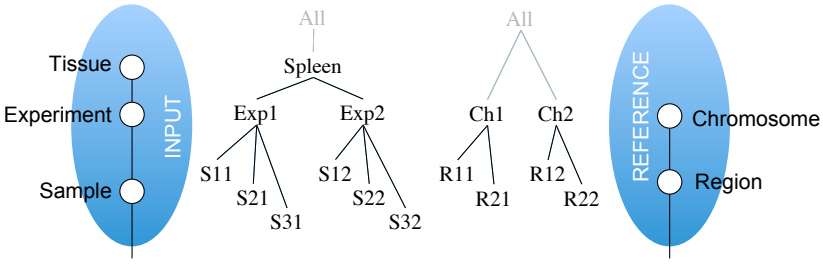


Figure 3: Fictitious roll-up functions for the INPUT and the REFERENCE hierarchies

A group-by includes one level for each hierarchy, and defines a possible way to aggregate facts.

Definition 2 (Group-by) *A* group-by *of schema $\mathcal{M}$ is an element $G \in \times_{i=1}^{n} L_i$. A* coordinate *of $G = \langle l^1, \ldots, l^n \rangle$ is an element $g \in \times_{i=1}^{n} Dom(l^i)$. The group-by including the bottom level of each hierarchy (i.e., the finest group-by of $\mathcal{M}$) will be denoted as $G_\perp = \langle l_1^1, \ldots, l_1^n \rangle$.*

Example 2 *Three examples of group-by's on the* MAPPING *schema are $G_\perp = \langle$*Sample, All*$\rangle$, $G_1 = \langle$*Sample, Chromosome*$\rangle$, and $G_2 = \langle$*Tissue, Region*$\rangle$. A coordinate of G_1 is $\langle$*S21, Ch2*$\rangle$.*

A schema is populated with facts, each recording a useful information for the decision-making process. A fact is characterized by a group-by set G that defines its aggregation level, by a coordinate of G, and by a numerical value v. The hierarchy values corresponding to each member of the fact coordinate are determined by the roll-up functions as from Definition 1.

Definition 3 (Cube) *A* cube *at group-by G is a (partial) function C that maps each coordinate $g \in G$ to a numerical value called* measure*. Each couple $\langle g, v \rangle$ such that $C(g) = v$ is called a* fact *of C.*

Example 3 *Two examples of facts of* MAPPING *are $\langle\langle$*S21, R22*$\rangle, 2\rangle$ and $\langle\langle$*S21, Ch2*$\rangle, 900\rangle$. The measure in this case counts the number of regions of each input sample that overlap with each reference region.*

3 The QETL Approach

A functional view of the QETL process is shown in Figure 4, and its components are explained below. The abstraction we use to compactly represent the facts currently stored in the cube, those required to answer an OLAP query, those missing, and those to be requested to the source data provider through the ETL is called *dice* and formally defined in Section 3.1; intuitively, a dice is a multidimensional interval of coordinates that determines a set of facts.

- The **dice management** process takes an OLAP query q (received in MDX form from the OLAP front-end) and checks, using a map of the

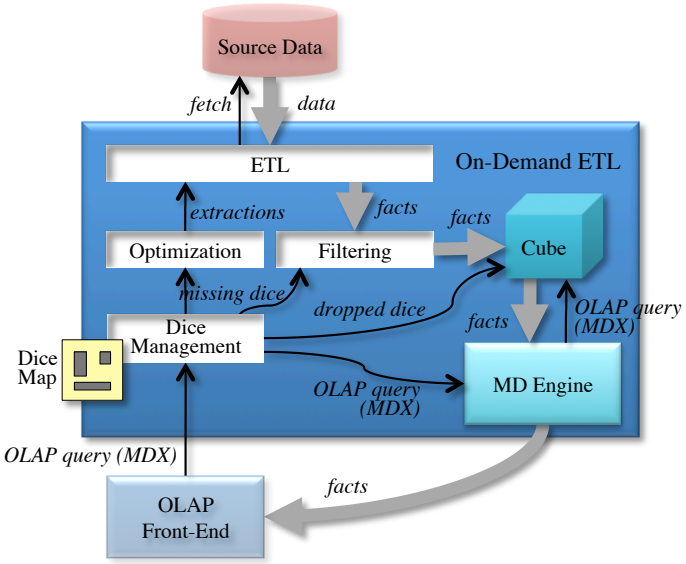


Figure 4: Functional architecture for on-demand ETL

dice currently available in the cube (*dice map*), if q can be immediately answered or some facts are missing. In the first case, q is sent to the multidimensional engine for processing. In the second case, the difference between the dice required by q and the available dice is computed in terms of a set of missing dice and handed to the optimization process. This process is also in charge of choosing the dice to be dropped from the cube when some room is needed.

- **ETL**: this is a traditional ETL process that offers an interface consisting of a set of *(extraction) services*. Considering the limitations possibly posed by the source data provider and by its query language, each service supports selection predicates on a subset of levels and is capable of returning the set of facts corresponding to a single dice. When a service is called with a specific selection predicate, the ETL turns it into a query on the source data provider, fetches the required data, and transforms them into multidimensional form. The ETL has a model for estimating the cost of each call to a service, based in general on both the cost for data fetching and those for their transformation.

- The **optimization** process knows the interface offered by the ETL and the cost for each service call as exposed by the ETL. Based on this information, it determines a set of extractions that cover all the missing dice and has total minimum cost. Each extraction entails a call to a service.

- Since the interface exposed by the source data provider does not necessarily allow full querying expressiveness, the facts fetched at each time may be a superset of those actually needed. The **filtering** process filters them before loading them into the cube; then it sends the set of loaded dice to the dice management process that updates the dice map accordingly.

The single processes mentioned above are described in more detail in the following subsections.

3.1 Query and Extraction Model

An OLAP query is normally defined by a group-by G and some selection predicates expressed on levels. To start simple, here we assume that facts are extracted and loaded into the cube only at their finest granularity, to be then aggregated by G by the multidimensional engine. For this reason, G is not relevant from the point of view of on-demand ETL, and we can simply represent a query q as a set of multidimensional intervals —those induced through the roll-up functions on the domains of the finest levels $l_1^1, \ldots l_1^n$, etc. by the selection predicates of q— that determine the coordinates of the facts to be returned to the user. The abstraction we use to this end is called *dice* and defined below.

Definition 4 (Range and Dice) *A* range r^i *of level* l_1^i *is an interval of members* (m', m'') *such that* $m', m'' \in Dom(l_1^i)$ *and* $m' \leqslant m''$. *A* dice d *is an* n*-dimensional interval of coordinates,* $d = \times_{i=1}^{n} r^i$ *where* r^i *is a range of* l_1^i *for* $i = 1, \ldots, n$.

Working with ranges requires that a total order is defined on the members of each level l_1^i. To define such order we observe that, in several OLAP front-ends, the default behavior when a user clicks on a row/column of a pivot table (corresponding to a member of a level) is to disaggregate the measure values for that row/column into its components, which in OLAP terms means slicing and drilling down [6]. For instance, starting from a report showing mappings per tissue and chromosome, clicking on member Spleen would trigger a query showing mappings for experiments Exp1 and Exp2, while clicking on Ch1 would trigger a query showing mappings for regions R11 and R21. Normally, within each group, members are alphabetically sorted. For this reason, to define ranges and dice we will adopt a *hierarchy-based lexicographic order*, i.e., one in which the members that roll-up to the same member are lexicographically ordered.

From the topological point of view, two dice d and d' are either disjoint ($d \parallel d'$), overlapping ($d \sim d'$), or one of them is included in the other ($d \subseteq d'$). Of course, the exact relationship between two dice depends on whether each range in each dice is left-open/closed and right-open/closed. If you consider the 2-dimensional example in Figure 5, with $d = (1, 3) \times (\mathrm{a}, \mathrm{i})$, $d' = (5, 8) \times (\mathrm{l}, \mathrm{p})$, and $d'' = (3, 10) \times (\mathrm{d}, \mathrm{p})$, it is always $d \parallel d'$, but the other relationships depend on the range closeness. Specifically, if d is right-closed and d'' is left-closed on the first dimension, then it is $d \sim d''$, otherwise $d \parallel d''$. Similarly it can be either $d' \sim d''$ (when d' is right-closed and d'' is right-open on the second dimension) or $d' \subseteq d''$ (in all other cases).

Definition 5 (OLAP Query) *An* OLAP query *is defined as a set* Q *of dice that represent the coordinates of the facts to be returned.*

Example 4 *A dice of our* MAPPING *schema is* $d = (\mathrm{S11}, \mathrm{S12}) \times (\mathrm{R22}, \mathrm{R22})$. *Adopting the hierarchy-based lexicographic order for the domains of both regions and chromosomes (like in Figure 3), this dice includes* 4×1 *coordinates. The query asking for the number of mappings between samples of tissue* Spleen *and regions of chromosome* Ch2 *is defined by* $Q = \{(\mathrm{S11}, \mathrm{S32}) \times (\mathrm{R12}, \mathrm{R22})\}$ *(which includes* 6×2 *coordinates).*

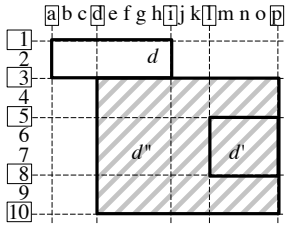


Figure 5: Topological relationships between three dice

Like for OLAP queries, our model for the extractions supported by the ETL process is based on dice. However, while an OLAP query can correspond to any set of dice, a data provider normally has some limitations about the queries it can answer to (for instance, selection may be possibile only on a subset of levels), and these limitations restrict the set of dice that the ETL can return in practice. This is captured by the definition of service and interface. An interface is the set of services supported by ETL. A service allows the specification of selection (range) predicates on the members of one or more levels of different hierarchies, and corresponds to a sequence of queries to the source data provider to fetch the necessary data, plus some transformations to put these data in multidimensional form.

Definition 6 (Interface and Service) *An* interface *is a set I of services. A* service *is defined by a group-by $S \in \times_i L_i$ that includes, for each hierarchy, the level on which it supports a selection.*

An extraction is issued by calling a service with a specific selection predicate. For simplicity we will assume that each extraction returns the (non-aggregated) facts corresponding to exactly one dice.

Definition 7 (Extraction) *An* extraction *using service $S = \langle l^1, \ldots, l^n \rangle$ is any dice $e = \times_{i=1}^{n} r^i$ such that, for each $i = 1, \ldots, n$, there exists an interval (m', m'') of $Dom(l^i)$ such that $Drill((m', m'')) \equiv r^i$, where $Drill((m', m'')) = \{m \in Dom(l_1^i) \mid RollUp^{l^i}(m) \in (m', m'')\}$.*

Intuitively, the extractions that use service S are those whose ranges can be induced through the roll-up functions by range predicates formulated on the levels of S. Note that, as a consequence of these definitions, if $l_{all}^i \in S$ for some i, then all extractions using S are characterized by range $(-\infty, +\infty)$ on h_i, which means that no selection on h_i is supported by S.

Example 5 *A possible interface for the GOLAM framework is $I = \{S_1, S_2\}$ where $S_1 = \langle \mathsf{Sample}, \mathsf{All} \rangle$ and $S_2 = \langle \mathsf{Experiment}, \mathsf{Chromosome} \rangle$. Examples of extractions using services S_1 and S_2, respectively, are $e_1 = (\mathrm{S31}, \mathrm{S32}) \times (-\infty, +\infty)$ (which can be obtained using predicate $(\mathsf{Sample} \geqslant \mathrm{S31})$) and $e_2 = (\mathrm{S11}, \mathrm{S31}) \times (\mathrm{R12}, \mathrm{R22})$ (which can be obtained using predicate $(\mathsf{Experiment} = \mathrm{Exp1}) \wedge (\mathsf{Chromosome} = \mathrm{Ch2})$).*

3.2 Dice Management

The main function of this process is that of determining the set F of missing dice to answer a given OLAP query. To this end, this process must be capable of executing dice operations; in particular, given a set Q of query dice and the set D of dice in the dice map (those currently available in the cube), it can compute their difference F using Algorithm 1. The basic idea of the dice difference operation is to split each dice in Q into fragments based on ranges "aligned" to the ranges in the dice of D, so that each resulting fragment is either included in a dice of D (in which case it needs not be loaded) or disjoint from all dice of D (in which case it is missing and must be loaded in its entirety).

Definition 8 (Range and Dice Fragmentation) *Given range $r^i = (m', m'')$ of level l_1^i and an (ordered) set of members $M^i \in Dom(l_1^i)$, let $\overline{M}^i = \{m_1, \ldots, m_p\}$ be the (ordered) subset of M^i included in r^i. The* fragmentation *of r^i according to M^i is the set of ranges $Frag_{M^i}(r^i) = \{(m', m_1), (m_1, m_2), \ldots, (m_p, m'')\}$. Given dice $d = \times_i r^i$ and an n-ple of sets of members $M = \langle M^1, \ldots, M^n \rangle$, where $M^i \in Dom(l_1^i)$ for $i = 1, \ldots, n$, the* fragmentation *of d according to M is the set of dice $Frag_M(d) = \times_i Frag_{M^i}(r^i)$. The right/left openness/closure for the ranges of the dice in $Frag_M(d)$ is chosen in such as way that the fragmentation is disjoint and complete.*

Remarkably, since $Frag_M(d)$ is based on the Cartesian product of ranges, it is the finest fragmentation that can be obtained starting from M. This ensures maximum flexibility in determining cheap extractions to obtain the missing dice during the optimization process (see Section 3.3). As a further note, the reason why we must allow for open ranges in defining dice is that, since QETL is based on incremental loading, we generally do not know the complete domains of the levels in $G_\perp$. Indeed, if all members were known from the beginning, ranges could be easily defined with closed predicates only, thus avoiding unnecessary complexity.

Example 6 *Consider again the example in Figure 5. Let $M = \langle \{2, 5, 8\}, \{\mathrm{i}, \mathrm{l}\} \rangle$; the fragmentation of dice d'' according to M includes the 9 grey dice shown by dashed lines (note that member 2 is external to d'', so it does not contribute to the fragmentation).*

Initially, Algorithm 1 considers all dice in Q to be missing (line 1). Then, for each dice q in Q it checks if there is an overlap with any dice in the dice map D (line 6). If not, q is entirely missing and stays in F. If some overlapping dice are found, the end members of their ranges are used to fragment q (line 12). Finally, we just have to delete from F the fragments of q that are already present in the dice map D (line 13). In case a dice q is completely included into a dice of D, it is simply removed from F (line 4). The details about the management of open/closed ranges are not shown in Algorithm 1 for the sake of simplicity, but they will be briefly commented in the following example.

Example 7 *Consider the 2-dimensional example in Figure 6, with $D = \{d, d'\}$, $Q = \{q\}$, $d = (1, 4) \times (\mathrm{a}, \mathrm{i})$, $d' = (9, 12) \times (\mathrm{n}, \mathrm{s})$, and $q = (3, 10) \times (\mathrm{d}, \mathrm{p})$. Since dice d and d' are in the dice map (i.e., they have already been loaded in the*

Algorithm 1 $DiceDifference(Q, D)$

Require: A set of query dice Q and a set D of available (disjoint) dice
Ensure: A set F of missing dice

```
1: F ← Q
2: for all q ∈ Q do
3:     if ∃d ∈ D | q ⊆ d then                          ▷ Dice q is already covered by D...
4:         F ← F\{q}                                    ▷ ...so it is not missing
5:     else
6:         O ← {d ∈ D | d ∼ q}                          ▷ Dice that overlap with q
7:         if (O ≠ ∅) then
8:             M ← ⟨∅, ..., ∅⟩                          ▷ Members for fragmentation
9:             for all d ∈ O, i = 1, ..., n do          ▷ For each range of each overlapping dice d...
10:                M^i ← M^i ∪ Ends^i(d)
11:                ▷ ... Ends^i(d) returns both end members of the i-th range in d
12:            F ← F\{q} ∪ Frag_M(q)                    ▷ Fragment q according to M
13: F ← F\{f ∈ F | ∃d ∈ D, f ⊆ d}                       ▷ Delete from F the dice covered by D
14: return F
```

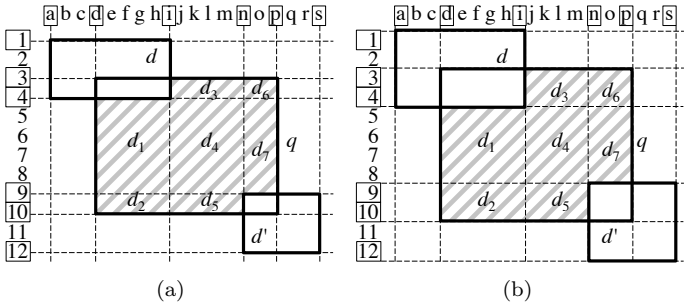


Figure 6: Dice difference on the dice map with generic ranges (a) and closed ranges (b)

cube), the missing facts that must be loaded to answer q are those in the grey-dashed area, that correspond to the following missing dice resulting from the dice difference operator (from left-top in Figure 6.a):

$$d_1 = (4, 9) \times (\mathrm{d}, \mathrm{i})$$
$$d_2 = (9, 10) \times (\mathrm{d}, \mathrm{i})$$
$$d_3 = (3, 4) \times (\mathrm{i}, \mathrm{n})$$
$$d_4 = (4, 9) \times (\mathrm{i}, \mathrm{n})$$
$$d_5 = (9, 10) \times (\mathrm{i}, \mathrm{n})$$
$$d_6 = (3, 4) \times (\mathrm{n}, \mathrm{p})$$
$$d_7 = (9, 10) \times (\mathrm{n}, \mathrm{p})$$

However, the specific queries to be issued to load the missing dice depend on whether the ranges that define d, d′, and q are actually closed or open. If we assume that all ranges in d, d′, and q are closed (e.g., $3 \leqslant l \leqslant 10$*), the actual situation is the one depicted in Figure 6.b. So it becomes clear that, for instance, the missing dice* d_1 *in this case must be left-open on* 4 *and left-closed on* d*, while* d_5 *must be right-closed on* 10 *and right-open on* n*.*

As already stated and shown in Figure 4, the set of dice available at any time is stored in a dice map (D in Algorithm 1). In practice, the dice map is

implemented by coupling a B^+-tree index (to record, for each dimension, the members currently loaded in the dimension tables) and a list of dice (to keep track of the facts currently present in the fact table). All dice in the dice map are disjoint.

The dice management process is also in charge of dropping some facts used for past queries from the cube to make room for the facts needed to answer new queries. The basic policy we adopt to this end is *least-recently used*, which discards the least-recently used dice first. Note that, since the dice size estimate may be imprecise for different reasons, the choice of the dice to be dropped is made based on the actual size of the new dice to be loaded, that is determined during the filtering process (see Section 3.4).

3.3 Optimization

When several missing dice must be loaded and different services are available, determining the cheapest set of extractions becomes an optimization issue related to both the specific services to be called and to the set of missing dice to be retrieved through a single extraction. Having a separate extraction for each missing dice could be quite expensive and time-consuming, for instance if most of the cost/time is paid to connect to the service rather than for data transfer and processing.

Different source data providers and different ETL processes can entail very different costs for extractions; the cost function to be used clearly depends on the features of the application domain and on how costs are measured (e.g., in terms of time, money, etc.). For the sake of flexibility we will not impose any specific constraints on the cost function, except that of being non-negative. For example, a simple family of cost functions that matches the application scenarios depicted in Section 1 is the one where a fixed cost for calling the service is summed to a cost proportional to the number $|e|$ of facts returned by extraction e. So in this case it is $cost(e) = \alpha_S + \beta_S|e|$, where $\alpha_S \geqslant 0$ and $\beta_S \geqslant 0$ depend on the service S used by e. When the application scenario is the pay-per-download one, α_S is the fee to be payed for each connection, while β_S is the cost per byte to be downloaded. In all of the other scenarios, α_S is the time needed to set up the connection while β_S is the time-per-byte needed to extract, transfer, and transform the data. To be independent of the sparsity of the cube and of the specific distribution of its facts, we will approximate the number of facts returned by extraction $e = \times_i r^i$ with the size of the corresponding dice, defined as $|e| = \prod_i |r^i|$.

Given the cost function, the optimization process takes in input the set F of missing dice produced by Algorithm 1 and produces in output a set E of extractions to be requested to the ETL. The specific problem to be solved to this end can be formulated as follows:

Problem 1 *Given a set F of (missing) dice and an interface I, find a set E of extractions such that (i) each extraction $e \in E$ uses a service in $S \in I$, (ii) $\bigcup_{f \in F} f \subseteq \bigcup_{e \in E} e$, and (iii) $\sum_{e \in E} cost(e)$ is minimal.*

Before we explain how Problem 1 can be solved, we need to show how a given set F of missing dice can be obtained by calling a service S of interface I.

Definition 9 (Minimal and Cheapest Extraction) *Let F be a set of dice, I be an interface, and S be a service. Then, the* minimal extraction *of F from*

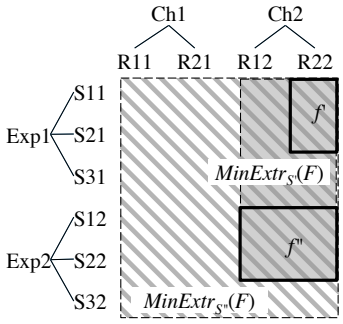


Figure 7: Minimal extractions with different services

S, denoted $MinExtr_S(F)$, is the smallest extraction e using S such that $f \subseteq e$ for each $f \in F$. The cheapest extraction *of F from I, denoted $CheapExtr_I(F)$, is the extraction $MinExtr_S(F)$ such that $S \in I$ and $cost(MinExtr_S(F))$ is minimum.*

Intuitively, $CheapExtr_I(F)$ determines the cheapest way to fetch all the facts belonging to F using I.

Example 8 *With reference to Figure 7, let $F = \{f', f''\}$ with $f' = (\mathrm{S11}, \mathrm{S21}) \times (\mathrm{R22}, \mathrm{R22})$ and $f'' = (\mathrm{S12}, \mathrm{S22}) \times (\mathrm{R12}, \mathrm{R22})$. Let the interface include two services: $S' = \langle \mathsf{Sample}, \mathsf{Chromosome} \rangle$ and $S'' = \langle \mathsf{Experiment}, \mathsf{All} \rangle$. Then it is $MinExtr_{S'}(F) = (\mathrm{S11}, \mathrm{S22}) \times (\mathrm{R12}, \mathrm{R22})$ (this extraction, in solid grey in the figure, is obtained calling S' with predicate $(\mathsf{Sample} \geqslant \mathrm{S11}) \wedge (\mathsf{Sample} \leqslant \mathrm{S22}) \wedge (\mathsf{Chromosome} = \mathrm{Ch2})$), and $MinExtr_{S''}(F) = (\mathrm{S11}, \mathrm{S32}) \times (\mathrm{R11}, \mathrm{R22})$ (this extraction, in dashed grey in the figure, is obtained calling S'' with predicate $(\mathsf{Experiment} \geqslant \mathrm{Exp1}) \wedge (\mathsf{Experiment} \leqslant \mathrm{Exp2})$).*

Solving Problem 1 by enumeration is obviously incompatible with an interactive analysis scenario like ours. To reduce the problem complexity, we approach it as a clustering problem where each cluster corresponds to an extraction, i.e., to a set of dice whose facts are fetched by a single service call; note that, by doing so, we do not consider all solutions in which a single dice in F is further fragmented to be fetched using multiple extractions (which could allow the number of unnecessarily fetched facts to be cut down). Even with this simplifying assumption, the search space is large enough to be hardly explorable in its entirety. Indeed, given a set F of dice and an interface I, the size of the search space is $\sum_{k=1}^{|F|} \left\{ {|F| \atop k} \right\} \cdot |I|^k$, where the first and the second terms represent, respectively, the Stirling number of second kind and the number of dispositions with repetitions.

The greedy solution we propose to tackle Problem 1 is outlined in Algorithm 2 and is based on hierarchical clustering [4]. Starting from a clustering —i.e., a partition— of the dice in F where each cluster is a singleton, we proceed by iteratively merging the two most promising clusters, i.e., those with minimum distance. The inter-cluster distance function we use to this end is

$$\delta(c', c'') = \frac{cost(CheapExtr_I(c' \cup c''))}{|MBD(c')| + |MBD(c'')|}$$

where c' and c'' are two sets of dice and $MBD(F)$ is the minimum bounding dice of a set of dice F, i.e., the smallest dice g such that $f \subseteq g$ for each $f \in F$. To avoid favoring the merging of small clusters, the denominator of the distance function weighs the cost for fetching all the facts in the two clusters with the total size of the minimum bounding dice for the two clusters.

The merging process is iterated for $|F| - 1$ times to build a complete dendrogram. The clustering C generated at each iteration corresponds to a set of extractions defined as

$$E = \{CheapExtr_I(c)\ ,\ c \in C\}$$

Eventually, the clustering corresponding to an extraction set with minimum cost is chosen as the solution.

Algorithm 2 *Optimize*(F, I)

Require: A set F of missing dice and an interface I
Ensure: A set E of extractions
1: $C \leftarrow \{\{f\}, f \in F\}$ ▷ Create a clustering of singletons
2: $E \leftarrow \{CheapExtr_I(c), c \in C\}$ ▷ Initialize the set of extractions
3: **while** $|C| > 1$ **do**
4: find $c', c'' \in C$ s.t. $c', c'' \in C \wedge c' \neq c'' \wedge \delta(c', c'')$ is minimum
5: ▷ Find the most promising couple of clusters
6: $C \leftarrow C \setminus \{c', c''\} \cup \{c' \cup c''\}$ ▷ Merge the two clusters
7: $E' \leftarrow \{CheapExtr_I(c), c \in C\}$ ▷ Best extraction set of the new clustering
8: **if** $\sum_{e' \in E'} cost(e') < \sum_{e \in E} cost(e)$ **then** ▷ Compare the costs of the two extraction sets
9: $E \leftarrow E'$ ▷ New extraction set
10: **return** E

The overall computational complexity of Algorithm 2 is $O(|F|^3 \cdot |I|)$, where F is the set of missing dice and I is the interface exposed by the data provider. $O(|F|^3)$ is the total number of comparisons between clusters (at each iteration $O(|F|^2)$ comparisons are done, and the total number of iterations is $|F| - 1$). For each comparison, $|I|$ services must be evaluated to determine the cheapest extraction, from which the total complexity follows.

3.4 Filtering

The extractions determined by optimization are requested to the ETL component, which executes them by querying the provider and returns a set of facts to the filtering process. As previously mentioned, the interface exposed by the source data provider may allow for selecting the data to be extracted using predicates on some hierarchy levels only, so the facts returned may be a superset of those actually needed. The aim of this process is mainly to discard all the facts not required to answer the user's current query, which requires to check, for each fact, if it is included in one of the missing dice returned by Algorithm 1; since extractions could overlap, attention should be paid to the management of duplicates.

This process also determines the exact number of facts included in each dice extracted. As mentioned in Section 3.2, the exact number of facts per dice is used during the dropping process to decide how many dice must be dropped to make room for new facts (the granularity of a single drop operation is exactly one dice, i.e., a dice is either completely dropped or not dropped at all). Counting the facts per dice at this stage is necessary because the dice size

used by optimization to estimate the cost of each extraction is imprecise due to the cube sparsity and to the partial knowledge of the level domains.

So far, we have assumed that, after each OLAP query, only the facts strictly required to answer that query are loaded into the cube (*strict loading*). However, another viable approach is that of loading in the cube *all* the facts extracted (*loose loading*) The choice of the best approach depends on the particular situation at hand. For example, if the goal is just to minimize the processing time, strict loading might be more suitable because it does not overload the cube with unnecessary facts. Conversely, if data must be purchased from the provider, loose loading might be a better option. We remark that, from an implementation point of view, the only relevant difference between strict and loose loading concern the management of overlapping extractions: indeed, in case of loose loading, a dice difference operation must be performed to avoid representing overlapping dice in the dice map.

4 Conclusions

In this report we have presented QETL, an approach to incremental on-demand ETL based on the query-extract-transform-load paradigm. Our approach is beneficial within scenarios in which traditional batch ETL is unfeasible or inconvenient for either time, space, or cost reasons. Essentially, in QETL the multidimensional cube is operated as a sort of cache to enable data reuse for single and multiple users. Experimental results show that the execution times of QETL are normally compatible with the interactivity requirement of OLAP, and that data reuse and optimization successfully contribute to the approach efficiency.

References

[1] MDX reference, 2015.

[2] Alberto Abelló, Jérôme Darmont, Lorena Etcheverry, Matteo Golfarelli, Jose-Norberto Mazón, Felix Naumann, Torben Bach Pedersen, Stefano Rizzi, Juan Trujillo, Panos Vassiliadis, and Gottfried Vossen. Fusion cubes: Towards self-service business intelligence. *IJDWM*, 9(2):66–88, 2013.

[3] Lorenzo Baldacci, Matteo Golfarelli, Simone Graziani, and Stefano Rizzi. GOLAM: A framework for analyzing genomic data. In *Proc. DOLAP*, pages 3–12, Shanghai, China, 2014.

[4] Leonard Kaufman and Peter J. Rousseeuw. *Finding Groups in Data: An Introduction to Cluster Analysis*. Wiley Series in Probability and Statistics. Wiley, 2009.

[5] Brian Raney, Melissa Cline, Kate Rosenbloom, Timothy Dreszer, Katrina Learned, Galt Barber, Laurence Meyer, Cricket Sloan, Venkat Malladi, Krishna Roskin, Bernard Suh, Angie Hinrichs, Hiram Clawson, Ann Zweig, Vanessa Kirkup, Pauline Fujita, Brooke Rhead, Kayla Smith, Andy Pohl,

Robert Kuhn, Donna Karolchik, David Haussler, and James Kent. ENCODE whole-genome data in the UCSC genome browser. *Nucleic Acids Res.*, (39):D871–D875, 2011.

[6] Stefano Rizzi and Enrico Gallinucci. CubeLoad: A parametric generator of realistic OLAP workloads. In *Proc. CAiSE*, pages 610–624, Thessaloniki, Greece, 2014.

Section 2.2 — Discovering frequent correlations from genomic metadata

Elena Baralis, Luca Cagliero, Tania Cerquitelli,
Silvia Chiusano, and Paolo Garza

DAUIN - Politecnico di Torino

1 Introduction

Gene expression is the process by which information from a gene is used in the synthesis of a functional gene product [9]. These products are often proteins, but in non-protein coding genes such as rRNA genes or tRNA genes, the product is a functional RNA. In recent years, the rapid advance of molecular biology techniques (e.g., microarray analysis) has allowed biologists to generate thousands of gene expression measurements in a short time. Gene Expression Datasets (GEDs) usually collect the expression values of thousands of genes within hundreds of samples. Samples can relate to different organisms or tissues and can be acquired in different environmental conditions.

Data mining, which focuses on studying effective and efficient algorithms to transform large amounts of data into useful knowledge [25], may provide valuable insights into GEDs. Several works have exploited clustering algorithms to identify groups of genes that are strongly correlated with each other, but uncorrelated with those of other groups [4, 6, 8]. In [3] a step further towards the generation of 3D gene clusters has been made. The authors propose ParTriCluster, an algorithm that discovers groups of genes behaving similarly across samples and time stamps. The research community also proposed effective classification techniques, i.e. supervised data analysis methods, to correlate gene expression patterns with given classification labels [15, 16, 17]. In the context of GED analysis, frequent itemset and association rule mining [1] have been exploited to (i) extract biologically relevant co-expressions among multiple genes [11]; (ii) discover correlations between environmental effects and gene expressions [19]; (iii) profile gene expressions according to a worthwhile subset of gene correlations [23]; (iv) determine biological data duplicates [12]. A parallel effort has also been devoted to developing novel itemset mining algorithms that are able to effectively cope with high-dimensional biological data (e.g. GEDs containing thousands of genes) [10, 20]. However, to perform itemset and rule mining, gene expression values are commonly discretized into a predefined number of bins. Specifically, experts are first asked to partition gene expression values into three discrete subsets (i.e., low-expressed, unexpressed, high-expressed). Then frequent itemsets, i.e. sets of co-regulated genes (items) that frequently co-occur in a GED, are extracted from discretized GEDs. The discretization step could bias the quality of the mining result because experts

have to assume a reliable data distribution. Consequently, analysts often analyze and compare the results produced by different discretization methods [5, 23].

This deliverable presents a novel and more effective approach to discovering itemsets from GEDs while avoiding the discretization step. Rather than discretizing gene expression values before executing the itemset mining process, we represent per-sample gene expression values as item weights. In other words, we consider GEDs as weighted datasets [26] for which expression values are mapped to item (gene) occurrences within each sample. Then, *weighted itemsets* are extracted from weighted data. Since item weights can be continuous, discovering weighted itemsets instead of traditional (not weighted) ones prevents experts from discretizing GEDs before analyzing them. For this reason our approach improves the effectiveness of the knowledge discovery process. To the best of our knowledge, this work is the first attempt to discover weighted itemsets from GEDs.

Several weighted itemset mining algorithms (e.g., [24, 26]) have been proposed to consider item weights during the itemset extraction process. In this study we adopted the weighted itemset mining strategy that has recently been proposed in [7]. To demonstrate the effectiveness of our approach we analyzed many real GEDs. The results achieved show the applicability of the proposed approach and significance of the patterns discovered.

This deliverable is organized as follows. Section 2 thoroughly describes the weighted itemset mining process from GEDs. Section 3 presents the performed experiments, while Section 4 draws conclusions.

2 Weighted itemset mining from Gene Expression Data

The weighted itemset mining process from Gene Expression Datasets (GEDs) entails the following steps:

(A) *Data preparation.* This step focuses on preparing GEDs to the subsequent itemset mining phase. To make data preparation as simple as possible, we applied the minimal amount of preprocessing steps. Notably, we prevent experts from discretizing gene expression values before executing the itemset mining algorithm.

(B) *Weighted itemset extraction.* The preprocessed GED data is analyzed to discover significant co-expressions among multiple genes.

(C) *Weighted itemset selection and ranking.* To allow experts to manually explore the extracted patterns, the mined itemsets are ranked and filtered according to their main quality measures.

In the following each step is thoroughly described.

2.1 Data preparation

A GED consists of a set of samples, where for each sample the expression values of a subset of genes is given. For our purposes, we model GEDs as weighted relational datasets. Let us consider a fixed subset $G=\{g_1, g_2, \ldots, g_m\}$

Table 1: Example of weighted relational dataset

sample ID	**⟨Gene, expression value⟩ pairs**
r_1	⟨g_1,0.61⟩, ⟨g_2,-0.31⟩, ⟨g_3,-0.72⟩, ⟨g_4,-0.45⟩
r_2	⟨g_1,0.52⟩, ⟨g_2,0.45⟩, ⟨g_3,0.28⟩, ⟨g_4,0.39⟩
r_3	⟨g_1,0.51⟩, ⟨g_2,0.67⟩, ⟨g_3,0.45⟩, ⟨g_4,0.38⟩

of m genes g_i. Genes will be also called *items* throughout the deliverable. A weighted relational dataset D=$\{r_1, r_2, \ldots, r_n\}$ is a set of n records r_i, one for each GED sample. A record r_i, $[1 \leq i \leq n]$, consists of a set of pairs $\{\langle g_1, ev_{1i}\rangle, \langle g_2, ev_{2i}\rangle, \ldots, \langle g_m, ev_{mi}\rangle\}$, where $g_j \in G$, $\forall\ 1 \leq j \leq m$. Gene occurrences in r_i are characterized by a weight, which indicates the gene expression value within the corresponding sample. We will denote as ev_{ji} the expression value (weight) of the j-th gene g_j in r_i throughout the deliverable.

Since the expression values of different genes in different samples are often spread across a relatively large value range, we normalized item weights using z-score normalization [25]. Normalization is commonly applied in GED analysis [5, 23]. Note that, unlike many data discretization methods, z-score normalization does not require experts to set appropriate threshold values.

Table 1 reports an example of normalized dataset with 3 samples and 4 genes. Gene occurrences within each sample are weighted by the corresponding expression value. For example, the normalized expression value of gene g_1 in sample r_1 is 0.61. Note that genes can take either negative, or null, or positive continuous normalized expression values.

2.2 Weighted itemset extraction

Frequent weighted itemsets are extracted from a weighted relational GED dataset D using a recently proposed weighted itemset mining strategy [7].

In the context of GED analysis, a k-itemset (i.e. an itemset of length k) is a set of k distinct genes in D. For example, $\{g_1,\ g_2\}$ is a 2-itemset that occurs in Table 1. Traditional (not weighted) itemset mining algorithms (e.g. Apriori [2]) commonly generate and select itemsets based on the relative frequency of occurrence (i.e. the support [1]) in the analyzed data (disregarding item weights).

To consider item weights during itemset mining, the concept of weighted support has already been introduced [26]. The key idea is to weigh itemset occurrences in each record (sample) by the weight (expression value) of the corresponding items (genes). In [7] the occurrences of an arbitrary itemset I in D are weighted by the weight of the least weighted item in I within each sample.

Definition 1 Weighted itemset support. *Let D be a weighted relational dataset, I a k-itemset, and $G(r_i)$ the subset of genes that are contained in an arbitrary record $r_i \in D$. The weighted support of I in D is defined as follows.*

$$wsup\ (I, D) \quad = \quad \frac{\sum_{r_i \in D|\ I \subseteq G(r_i)} \min_{j|\ g_j \in I}\ ev_{ji}}{|D|}$$

In the context of GED analysis, itemsets represent gene combinations, while the weighted support measure indicates their relative frequency of occurrence

in D weighted by the expression value of their least expressed gene within each sample. For example, the weighted support of $\{g_1, g_2\}$ in Table 1 is 0.22, because the least weighted gene expression values in r_1, r_2, and r_3 are -0.31, 0.45, and 0.51, respectively.

The frequent weighted itemset mining task entails extracting all the *frequent* weighted itemsets, i.e., the itemsets whose weighted support is equal to or above a given (analyst-provided) threshold wminsup.

However, the mined itemset set is often redundant, because frequent itemsets can represent partially overlapped information. Hence, the interestingness of part of the mining result can be limited. To address this issue, a relevant research effort has been devoted to discovering compact and not redundant frequent itemset subsets [18, 21, 22]. In this deliverable we target the extraction of two established itemset subsets, i.e. the maximal and closed itemsets [21, 22], because they have already been considered to be relevant itemset subsets in the context of GED analysis [10, 20].

Closed itemsets. Closed itemsets [21] are frequent itemsets for which none of their immediate supersets have their same support. Since the weighted support measure satisfies the anti-monotonicity property [7], it trivially follows that the immediate supersets of a closed itemset have support strictly less than those of the itemset itself. In the context of weighted itemset mining, I is closed if and only if (i) wsup(I,D) $\geq$ wminsup and (ii) for every $I_2 |\ I \subset I_2$ wsup(I_2,D) $\leq$ wsup(I,D).

Recalling the previous example, if we set wminsup=0.10 then $\{g_1, g_2\}$ is closed because it is frequent and none of its immediate supersets (i.e., $\{g_1, g_2, g_3\}$ and $\{g_1, g_2, g_4\}$) have its same support value (0.22).

Maximal itemsets. Maximal itemsets [22] are frequent itemsets for which all of their immediate supersets are infrequent with respect to the given support threshold. In the context of weighted itemset mining, I is maximal if and only if (i) wsup(I,D) $\geq$ wminsup and (ii) for every $I_2 |\ I \subset I_2$ wsup(I,D) $<$ wminsup. Maximal itemsets are the subset of closed itemsets characterized by maximal length.

For example, if we set wminsup=0.20 then $\{g_1, g_2\}$ is maximal because all of its immediate supersets are infrequent with respect to the support threshold.

To extract closed and maximal weighted itemsets, we adapted the FP-Growth-like [13] weighted itemset mining algorithm implementation, which was first proposed in [7], to closed and maximal itemset mining.

2.3 Weighted itemset selection and ranking

The extracted itemsets are analyzed by domain experts to discover significant co-expressions among multiple genes. Since the number of extracted closed or maximal itemsets can be relatively high, analysts can select the top-K itemsets in order of decreasing weighted support (where K is an analyst-provided parameter). Top-K itemsets are the most frequent gene correlations that occur in a gene expression dataset. According to Definition 1, itemset occurrences are weighted by the expression value of the least expressed gene. Hence, high-support itemsets represent gene combinations for which *all* the genes are highly expressed within each sample. On the other hand, low-support itemsets could represent noisy or less relevant information. Note that if experts are interested in discovering valuable correlations among multiple genes, then the analysis of

Table 2: Gene expression datasets and characteristics of the itemsets extracted

Name	Num. of samples	Num. of genes	wminsup	Num. of closed				Num. of maximal			
				Total	Len.=1	Len.=2	Len.≥3	Total	Len.=1	Len.=2	Len.≥3
BRC-ABL	15	12625	0.007	10	7	1	2	3	1	0	2
T-ALL	42	12625	10	865	13	112	740	78	2	1	75
COLON	62	2000	-0.0002	1998	1993	2	3	1991	1988	0	3
NEUROBL.	14	22283	0.007	12	10	2	0	8	6	2	0
SRBCT	88	2308	-0.17	2405	2308	81	16	2314	2245	53	16

the 1-itemsets (i.e. itemsets that consist of a single gene) is meaningless. Therefore, in such context of analysis, 1-itemsets can be discarded early to further reduce the mined set cardinality.

3 Experiments

We conducted experiments to analyze (i) the number and characteristics of the extracted itemsets and (ii) the biological significance of the mining result. We analyzed five publicly available GEDs, whose main characteristics are summarized in Columns (1)-(3) of Table 2. Each GED contains a subset of genes that appear in all samples. BRC-ABL, T-ALL[1], and NeuroBlastoma[2] had already been analyzed in previous research works concerning traditional (not weighted) itemset mining (e.g. [20, 10]), whereas COLON[3] and SRBCT[4] had already been used to assess the performance of biological data classifiers (e.g. [15]).

3.1 Characteristics of the extracted itemsets

Table 2 reports the main characteristics of the weighted closed and maximal itemsets that were mined from the analyzed datasets. For each dataset Column (4) indicates the wminsup value enforced, while Columns (5) and (9) report the number of mined weighted closed and maximal itemsets, respectively. To demonstrate that our approach also discovers high-order gene correlations we report the per-length itemset cardinality. For 4 out of 5 datasets 3-length itemsets or longer (i.e. sets of co-expressed genes composed of at least three genes) were extracted. Such patterns are often not considered by previous approaches. Nevertheless to allow experts to manually explore the mining result the number of discovered itemsets should be limited. To achieve this goal without discarding potentially interesting gene co-expressions, experts could consider only sets of co-expressed genes (i.e., 2-length itemsets or longer). This pruning step yields a significant itemset set cardinality reduction (i.e. above 50%) for 4 out of 5 GEDs.

[1] http://www.stjuderesearch.org/data/
[2] http://www.broadinstitute.org/cgi-bin/cancer/datasets.cgi
[3] http://genomics-pubs.princeton.edu/oncology/affydata/index.html
[4] http://research.nhgri.nih.gov/microarray/

3.2 Result validation

We validated the significance of the results achieved on two representative gene expression datasets, i.e., BRC-ABL and T-ALL, which had previously been analyzed in [27] to perform classification, subtype discovery, and prediction of outcome in pediatric lymphoblastic leukemia. Specifically, both datasets relate the treatment of pediatric acute lymphoblastic leukemia (ALL). The research goal is to tailor the intensity of therapies to a patient's risk of relapse. Biologists used oligonucleotide microarrays to analyze the pattern of genes expressed in leukemic blasts from 360 pediatric ALL patients. High-density oligonucleotide arrays offer the opportunity to examine patterns of gene expression on a genome scale.

Each dataset consists of a set of expression profiles which are related to a specific prognostically important leukemia subtype, i.e., T-ALL, and BRC-ABL. Let us consider the T-ALL dataset first. Setting a minimum weighted support threshold wminsup=10, the itemset {RPSA RPS23}, with weighted support equal to 11.62, is extracted and ranked first in order of decreasing weighted support. This pattern represents an established correlation between two human ribosomal protein genes [14]. Similarly, the top-ranked itemset {BioB-3, SPECC1L, MAGED2} (wsup=0.01), which was extracted from the BRC dataset, represents a co-expression between the genes BioB-3, SPECC1L and MAGED2, which are targeted by the microarray probes. Gene co-expressions may provide important insights into the biology of the considered leukemia subgroups. Moreover, within each genetic subgroup the expression profiles that are highlighted by the patterns discovered could allow biologists to early identify those patients that would eventually fail therapies.

4 Conclusions

This deliverable presents a novel approach to itemset mining from Gene Expression Datasets (GEDs). The aim of this work is to ease GED preparation, which commonly requires a not trivial and expert-driven data discretization step. Instead of discovering traditional itemsets from discretized GEDs, we propose to consider gene expression values as item weights, which indicate gene expression intensity within each sample, and apply a weighted itemset mining algorithm [7] directly to non-discretized GED. The experimental results show the applicability and usefulness of the proposed approach on real GEDs.

References

[1] R. Agrawal, T. Imielinski, and Swami. Mining association rules between sets of items in large databases. In *ACM SIGMOD 1993*, pages 207–216, 1993.

[2] Rakesh Agrawal and Ramakrishnan Srikant. Fast algorithms for mining association rules in large databases. In Jorge B. Bocca, Matthias Jarke, and Carlo Zaniolo, editors, *VLDB'94, Proceedings of 20th International Conference on Very Large Data Bases*, pages 487–499. Morgan Kaufmann, 1994.

[3] Renata Braga Araújo, Guilherme Henrique Trielli Ferreira, Gustavo Henrique Orair, Wagner Meira, Renato Antônio Celso Ferreira, Dorgival Olavo Guedes Neto, and Mohammed Javeed Zaki. The partricluster algorithm for gene expression analysis. *Int. J. Parallel Program.*, 36(2):226–249, April 2008.

[4] Wai-Ho Au, Keith C. C. Chan, Andrew K. C. Wong, and Yang Wang. Attribute clustering for grouping, selection, and classification of gene expression data. *IEEE/ACM Trans. Comput. Biol. Bioinformatics*, 2(2):83–101, April 2005.

[5] Vincenzo Belcastro, Velia Siciliano, Francesco Gregoretti, Pratibha Mithbaokar, Gopuraja Dharmalingam, Stefania Berlingieri, Francesco Iorio, Gennaro Oliva, Roman Polishchuck, Nicola Brunetti-Pierri, and Diego di Bernardo. Transcriptional gene network inference from a massive dataset elucidates transcriptome organization and gene function. *Nucleic acids research*, 39(20):8677–8688, November 2011.

[6] Amir Ben-Dor and Zohar Yakhini. Clustering gene expression patterns. In *Proceedings of the third annual international conference on Computational molecular biology*, RECOMB '99, pages 33–42, 1999.

[7] Luca Cagliero and Paolo Garza. Infrequent weighted itemset mining using frequent pattern growth. *IEEE Transactions on Knowledge and Data Engineering*, 99(PrePrints):1, 2013.

[8] Yizong Cheng and George M. Church. Biclustering of expression data. In *ISMB*, pages 93–103, 2000.

[9] D.P. Clark and N.J. Pazdernik. *Molecular Biology: Understanding the Genetic Revolution*. Elsevier Science, 2012.

[10] Gao Cong, Anthony K. H. Tung, Xin Xu, Feng Pan, and Jiong Yang. Farmer: finding interesting rule groups in microarray datasets. In *Proceedings of the 2004 ACM SIGMOD international conference on Management of data*, SIGMOD '04, pages 143–154. ACM, 2004.

[11] Chad Creighton and Samir Hanash. Mining gene expression databases for association rules. *Bioinformatics*, 19(1):79–86, 2003.

[12] Ahmed K. Elmagarmid, Panagiotis G. Ipeirotis, and Vassilios S. Verykios. Duplicate record detection: A survey. *IEEE Trans. Knowl. Data Eng.*, 19(1):1–16, 2007.

[13] Jiawei Han, Jian Pei, and Yiwen Yin. Mining frequent patterns without candidate generation. In *Proceedings of the 2000 ACM SIGMOD International Conference on Management of Data*, pages 1–12, 2000.

[14] Kyota Ishii, Takanori Washio, Tamayo Uechi, Maki Yoshihama, Naoya Kenmochi, and Masaru Tomita. Characteristics and clustering of human ribosomal protein genes. *BMC Genomics*, 7(1):1–16, 2006.

[15] Mark A. Iwen, Willis Lang, and Jignesh M. Patel. Scalable rule-based gene expression data classification. In *ICDE*, pages 1062–1071, 2008.

[16] Mehdi Khashei, Ali Zeinal Hamadani, and Mehdi Bijari. A fuzzy intelligent approach to the classification problem in gene expression data analysis. *Know.-Based Syst.*, 27:465–474, March 2012.

[17] Ying Lu and Jiawei Han. Cancer classification using gene expression data. *Inf. Syst.*, 28(4):243–268, June 2003.

[18] Michael Mampaey, Nikolaj Tatti, and Jilles Vreeken. Tell me what I need to know: Succinctly summarizing data with itemsets. In *Proceedings of the 17th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, 2011.

[19] Ricardo Martinez, Nicolas Pasquier, and Claude Pasquier. Computational intelligence methods for bioinformatics and biostatistics. chapter Mining Association Rule Bases from Integrated Genomic Data and Annotations, pages 78–90. 2009.

[20] Feng Pan, Gao Cong, Anthony K. H. Tung, Jiong Yang, and Mohammed J. Zaki. Carpenter: finding closed patterns in long biological datasets. In *Proceedings of the ninth ACM SIGKDD international conference on Knowledge discovery and data mining*, KDD '03, pages 637–642. ACM, 2003.

[21] Nicolas Pasquier, Yves Bastide, Rafik Taouil, and Lotfi Lakhal. Discovering frequent closed itemsets for association rules. In *Proceedings of the 7th International Conference on Database Theory*, ICDT '99, pages 398–416, London, UK, UK, 1999. Springer-Verlag.

[22] J. Roberto and J.r. Bayardo. Efficiently mining long patterns from databases. In Laura M. Haas and Ashutosh Tiwary, editors, *SIGMOD 1998*, pages 85–93, 1998.

[23] Pedro C. Saez, Monica Chagoyen, Andres Rodriguez, Oswaldo Trelles, Jose Carazo, and Alberto P. Montano. Integrated analysis of gene expression by association rules discovery. *BMC Bioinformatics*, 7(1):54+, 2006.

[24] Ke Sun and Fengshan Bai. Mining weighted association rules without preassigned weights. *IEEE Transactions on Knowledge and Data Engineering*, 20(4):489 –495, 2008.

[25] Pang-Ning Tan, Michael Steinbach, and Vipin Kumar. *Introduction to Data Mining*. Addison Wesley, 2005.

[26] Wei Wang, Jiong Yang, and Philip S. Yu. Efficient mining of weighted association rules (WAR). In *Proceedings of the sixth ACM SIGKDD international conference on Knowledge discovery and data mining, KDD'00*, pages 270–274, 2000.

[27] Eng J. Yeoh, Mary E. Ross, Sheila A. Shurtleff, Kent W. Williams, Divyen Patel, Rami Mahfouz, Fred G. Behm, Susana C. Raimondi, Mary V. Relling, Anami Patel, and Cheng. Classification, subtype discovery, and prediction of outcome in pediatric acute lymphoblastic leukemia by gene expression profiling. *Cancer Cell*, 1(2):133–143, 2002.

Section 2.3 — Discovering frequent correlations from medical data

Elena Baralis, Luca Cagliero, Tania Cerquitelli,
Silvia Chiusano, and Paolo Garza

DAUIN - Politecnico di Torino

1 Introduction

Healthcare systems are nowadays integrated platforms that can take advantage of advanced data management and analysis solutions. Large amount of data on medical patient history is commonly stored by healthcare organizations. Data mining techniques can be used to analyze these large data collections and to extract knowledge useful for physicians and healthcare organizations.

This study addresses the problem of analyzing patient historical data to identify valuable correlations among patient treatments and profiles. The extracted patterns allow experts to (i) identify the medical treatments commonly followed by patients with a given disease, (ii) verify the adherence of medical treatments to shared medical guidelines, (iii) improve the effectiveness of medical treatments, and (iv) plan resource allocation and reduce costs incurred by organizations.

Association rule extraction is an established data mining technique to discover interesting correlations among large datasets [14]. Since patient history data is relatively sparse, discovering association rules from these datasets is a challenging task. Discovering correlations among data items that rarely co-occur may become computationally intractable when coping with large datasets. On the other hand, focusing only on most frequent item recurrences could provide not fruitful enough information. Furthermore, since a large rule set could be mined, inferring useful and actionable knowledge from the extracted rules can be a complex task.

This deliverable presents: (i) MeTA (Medical Treatment Analysis), a new data analysis framework targeted to the discovery of underlying multiple-level correlations among patient treatments and profiles. (ii) The classification of the mined rules into classes according to the represented data features (e.g., examinations, drugs). (iii) The exploration of rules in order of descending level of abstraction of the represented information in the input taxonomy. (iv) The application of MeTA to a real-life use case, i.e., the analysis of diabetic patient data provided by the National Health Center (NHC) of an Italian province.

Patients datasets consist of log files holding information about patient treatments and census data. Each row contains a set of pairs (*feature*, *value*), where *feature* corresponds to a specific data feature (i.e., *Examination*, *Drug*, *Age*, or

Gender), while *value* is the corresponding feature value. MeTA discovers interesting and multiple-level correlations among patient data called generalized rules [11]. These rules are represented in the form $X \rightarrow Y$, where X and Y are disjoint sets of items (called itemsets). The implication means that (i) itemsets X and Y frequently co-occur in the analyzed dataset (regardless of the temporal order of occurrence of X and Y in the source data), (ii) the strength of the implication between X and Y is above a given threshold, and (iii) X and Y may also include items belonging to different abstraction levels. Item generalization is driven by a taxonomy, which consists of a set of is-a hierarchies built over the analyzed data. For example, drugs can be generalized based on the addressed pathology [4], while examinations are clustered based on the focused area (e.g., liver or cardiovascular system). Aggregating items into higher-level concepts (e.g., examinations into the corresponding category) prevents the discarding of potentially useful knowledge and thus counteracts the issue of data sparseness. In our context of analysis, we disregard the temporal order of prescriptions and we specifically focus on discovering multiple-level co-occurrences among examination/drug prescriptions. In Section 3 we will demonstrate that these patterns are worth considering for targeted analysis (e.g., resource allocation, healthcare service management). To make the mined result more manageable by domain experts for manual inspection, MeTA considers a worthwhile rule subset, i.e., the non-redundant rules [16]. Non-redundant rules are generated from closed itemsets [10], which are a compact and non-redundant subset of frequent itemsets. Furthermore, MeTA categorizes the rules into four groups according to the represented data facet (e.g., examination, drugs). Within each group rules are further classified according to their level of abstraction of the contained items in the input taxonomy.

As an example, let us consider rule {(*Examination*, *Routine*), (*Examination*, *Cardiovascular*)} $\rightarrow$ {(*Drug*, *Blood and blood forming organs Category*)}. It indicates that drugs in category "Blood and blood forming organs" have been prescribed to a relatively large number of patients to whom routine and cardiovascular examinations have been prescribed as well (disregarding the temporal order of prescriptions). This information could be deemed to be useful, for example, for shaping drug provision to medical divisions according to the most commonly performed examinations. The rule is high-level, because it contains only examination and drug categories. Conversely, rules containing *also* or *only* single examinations/drugs will be denoted as cross- or low-level rules, respectively. The cross-level rule {(*Examination*, *Routine*), (*Examination*, *Cardiovascular*)} $\rightarrow$ {(*Drug*, *Acetylsalicylic Acid*)} can be extracted as well in case drug Acetylsalicylic Acid has predominantly been prescribed among the "Blood and blood forming organs" drug category. Note that the aforesaid high- and cross-level rules are more likely to be frequent than their low-level descendant rules (e.g., {(*Examination*, *Complete blood count*), (*Examination*, *Cholesterol*)} $\rightarrow$ {(*Drug*, *Acetylsalicylic Acid*)}). High- and cross-level rules are worth considering because (i) they represent, from a high-level viewpoint, valuable information that may remain hidden in sparse datasets at lower abstraction levels and (ii) they are typically more manageable than low-level rules for manual result exploration.

We assessed the usability of MeTA on a real dataset of diabetic patients provided by the National Health Center (NHC) of an Italian province. The experiments demonstrate that, starting from a large collection of raw data on

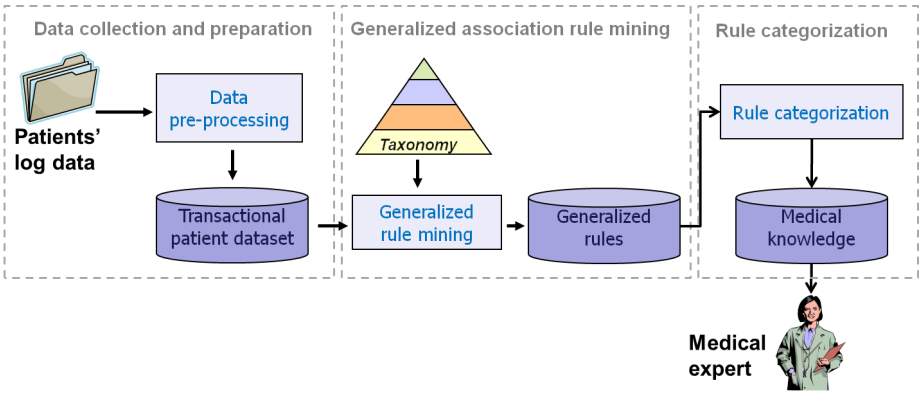


Figure 1: The Medical Treatment Analysis framework

patient history, the framework allows experts to identify several interesting high-, cross-, and low-level correlations among patient treatments and profiles. The results were validated by clinical domain experts. The extracted rules appear to be consistent with the guidelines for diabetes disease [1, 7, 8]. Furthermore, the extraction of high- and cross-level rules appears to effectively overcome limitations of traditional approaches.

This deliverable is organized as follows. Section 2 presents the architecture of the proposed framework and it describes its main blocks. Section 3 assesses the effectiveness of the system in performing knowledge discovery from a real diabetic patient dataset, while Section 4 draws conclusions of this work.

2 The Medical Data Generalized Rule Miner system

MeTA (Medical Treatment Analysis) is a novel framework for medical data analysis, which focuses on characterizing medical treatments at different granularity levels.

The main MeTA architectural blocks are depicted in Figure 1. A brief description of each block follows.

Data collection and preparation. This block aims at making information about patient characteristics, examinations, and drugs suitable for the mining process. Patient datasets are tailored to a transactional data format, where each transaction corresponds to a different patient and it consists of a set of items, which represent patient census data (e.g., age, gender), prescribed examinations (e.g., Glucose level), or prescribed drugs (e.g., Acetylsalicylic Acid). Transactional datasets are enriched with an (analyst-provided) taxonomy built over the data items.

Generalized association rule mining. This block focuses on discovering multiple-level correlations among the preprocessed data in the form of generalized association rules. The extraction process is driven by the input taxonomy to generalize data items at higher abstraction levels. To extract only the rules

that (i) occur frequently and (ii) represent positively correlated implications among pairs of item sets in the source dataset, rules are filtered according established quality measures, i.e., support and lift [13]. Furthermore, to filter out less informative rules only the subset of non-redundant rules [16] is considered for further analyses.

Rule categorization. To make the mining result manageable by experts for manual inspection, rules are categorized according to their represented information. To analyze correlations among patient data regardless of the patient profile, rule subsets that represent (i) correlations among examinations, (ii) correlations among drugs, and (iii) correlations between examinations and drugs are analyzed separately. On the other hand, to gain more insights into specific user profiles (e.g., elderly men, kids) implications between specific patient characteristics and examinations/drugs are considered. To easily explore rule categories the corresponding rules are further classified as high-level, cross-, or low-level according to the level of abstraction of the contained information in the input taxonomy.

A more thorough description of each block is reported below.

2.1 Data collection and preparation

Healthcare systems usually collect heterogeneous personal information about patients into log datasets. For example, the list of prescribed examinations and drugs is stored in separate log files to allow doctors to keep track of diagnosis and therapies and healthcare system managers to plan purchases and resource allocations. In parallel, to characterize the patient population, census data about patients, such as gender and age, are usually collected in separate datasets.

The MeTA framework collects and stores into a unique data repository these three main patient data types. More specifically, for each patient the list of (i) prescribed examinations, (ii) drugs, and (iii) the main patient characteristics are stored.

To enable the mining process, the collected patient data is tailored to a transactional data format. A transactional patient dataset is a set of transactions, where each transaction corresponds to a patient and it consists of a set of patient features, called items. Items can be related to examinations (e.g., *Glucose level*), drugs (e.g., *Acetylsalicylic Acid*), or patient census data (e.g., *Male*). In this work we focus on age and gender as peculiar patient census data. Items are represented in the form (*feature*,*value*), where *feature* is *Examination*, *Drug*, *Age*, or *Gender*, while *value* is the corresponding feature value. A more formal definition of transactional patient dataset is given below.

Definition 1 Transactional patient dataset. *Let E be the set of all possible patient examinations, M the set of all possible drugs, and C the set of census data features. Let $\Omega(c_i)$ be the domain of an arbitrary census data feature $c_i \in C$ (i.e., the set of all possible values assumed by c_i). An item is a pair (feature,v_i), where $v_i \in E$ if feature=Examination, $v_i \in M$ if feature=Drug, and $v_i \in \Omega(c_i)$ if feature=c_i. A transactional patient dataset $\mathcal{D}$ is a set of transactions, where each transaction $t_i \in \mathcal{D}$ is a set of items.*

Table 1: Example of patient transactional dataset.

Pid	Transaction
1	{(*Age*,*Elder*), (*Gender*,*Male*), (*Examination*,*HDL Cholesterol*), (*Examination*,*Glucose level*), (*Examination*,*Electrocardiogram*), (*Drug*,*Acetylsalicylic Acid*)}
2	{(*Age*,*Elder*), (*Gender*,*Female*), (*Examination*,*Glucose level*), (*Drug*,*Acetylsalicylic Acid*), (*Drug*,*Moxifloxacin*)
3	{(*Age*,*Elder*), (*Gender*,*Male*), (*Examination*,*Glucose level*), (*Examination*, *Electrocardiogram*), (*Examination*,*Blood count*), (*Drug*,*Moxifloxacin*)
4	{(*Age*,*Teenager*), (*Gender*,*Female*), (*Examination*,*Glucose level*), (*Drug*,*Acetylsalicylic Acid*), (*Drug*,*Moxifloxacin*)
5	{(*Age*,*Elder*), (*Gender*,*Male*), (*Examination*,*Electrocardiogram*), (*Examination*,*Blood count*), (*Drug*,*Acetylsalicylic Acid*)

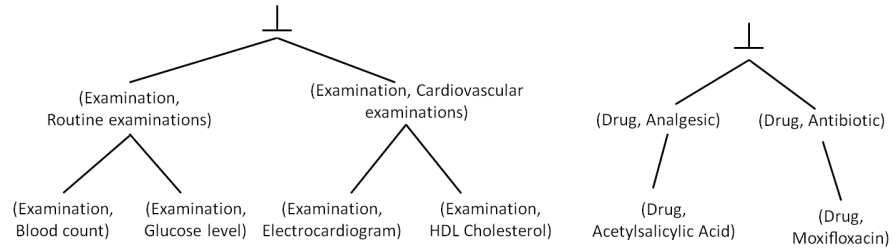


Figure 2: Example of taxonomy built over the transactional patient dataset.

Let us consider, as running example, the dataset reported in Table 1. It consists of 5 records, each one related to a different patient. For each patient the identifier (Pid), age (Age), gender (Gender), and a list of prescribed examinations and drugs is given. The dataset contains four different examinations (*HDL Cholesterol*, *Glucose level*, *Electrocardiogram*, and *Blood count*) and two different drugs (*Acetylsalicylic Acid* and *Moxifloxacin*). For example, patient with Pid 5 is an elderly man to whom examinations *Electrocardiogram* and *Blood count* have been prescribed at least once. Furthermore, he has already taken *Acetylsalicylic Acid* but not *Moxifloxacin*.

To enable the process of generalized rule mining from a transactional patient dataset $\mathcal{D}$, a taxonomy (i.e., a set of generalization hierarchies) is built over the items in $\mathcal{D}$. The taxonomy aggregates examinations and drugs into high-level concepts, i.e., examinations are generalized as examination categories while drugs as drug categories.

Definition 2 Taxonomy. *Let $\mathcal{D}$ be a transactional patient dataset and $\mathcal{I}$ the set of items in $\mathcal{D}$. A generalization hierarchy GH_{I_k} ($\mathcal{I}_k \subseteq \mathcal{I}$) built over $\mathcal{D}$ is a predefined hierarchy of aggregations defined over a subset of items in $\mathcal{I}$, where hierarchy leaves are items in $\mathcal{I}$, while non-leaf nodes in GH_{I_k} are ancestors of their corresponding children. Each hierarchy has a root node (denoted as $\perp$) which aggregates all its items. A taxonomy $\mathcal{T}$ built over $\mathcal{D}$ consists of a set of generalization hierarchies GH_{I_k} for which $\cup_{GH_{I_k} \in \mathcal{T}} \mathcal{I}_k = \mathcal{I}$.*

Although taxonomies can potentially contain many generalizations over the same item (e.g., many categories for the same examination), in this work we will consider only taxonomies containing at most one generalization per item.

An example of taxonomy built over the running example dataset is reported in Figure 2. Examinations *Blood count* and *Glucose level* are classified as *Routine examinations*, whereas examinations *Electrocardiogram* and *HDL Cholesterol* are generalized as *Cardiovascular*. Finally, drugs *Acetylsalicylic Acid* and *Moxifloxacin* are classified as *Analgesic* and *Antibiotic*, respectively.

2.2 Generalized association rule mining

This block focuses on discovering multiple-level associations, in the form of generalized association rules, from the transactional patient dataset $\mathcal{D}$ with a taxonomy $\mathcal{T}$.

Association rules represent underlying correlations among the analyzed data items [2]. More specifically, an association rule is an implication $A \Rightarrow B$, where A and B are itemsets, i.e., sets of data items. A k-itemset I a set of items of size k that occurs in $\mathcal{D}$.

For example, {(*Examination,Glucose level*),(*Examination,Electrocardiogram*)} is a 2-itemset that represents the co-occurrence of two specific examinations in medical treatments, while the association rule {(*Examination,Glucose level*)} $\rightarrow$ {(*Examination,Electrocardiogram*)} indicates that the occurrence of examination *Glucose level* "implies" those of examination *Electrocardiogram* in the analyzed data.

Generalized association rules [11] are rules that may also contain items at higher abstraction levels, i.e., the generalized items. Every item that is associated with a non-leaf node of the taxonomy $\mathcal{T}$ (see Definition 2) is considered as a generalized item. Similarly, generalized itemsets are itemsets including at least one generalized item.

Definition 3 Generalized itemset. *Let $\mathcal{D}$ be a transactional patient dataset and $\mathcal{I}$ be the set of distinct items in $\mathcal{D}$. Let $\mathcal{T}$ be a taxonomy built over $\mathcal{D}$ and $\mathcal{G}$ the set of generalized items (high-level tag aggregations) derived by all the generalization hierarchies in $\mathcal{T}$. A generalized itemset I is a subset of $\mathcal{I} \bigcup \mathcal{G}$ including at least one generalized item in $\mathcal{G}$.*

For example, according to the taxonomy in Table 2,
{(*Examination,Routine*),(*Examination,Electrocardiogram*)} is a generalized itemset.

Generalized itemsets are characterized by two quality indexes, i.e., the level and support. The level of a generalized item i_k with respect to a taxonomy indicates the degree of abstraction of the represented information.

Definition 4 Generalized itemset level. *Let $\mathcal{D}$ be a transactional patient dataset and $\mathcal{I}$ be the set of distinct items in $\mathcal{D}$. Let $\mathcal{T}$ be the taxonomy defined over $\mathcal{D}$ and i_k an arbitrary item or generalized item in $\mathcal{T}$. The level of (generalized) item i_k is defined as the height of $\mathcal{T}$'s subtree rooted in i_k. The level of a generalized itemset is defined as the maximum level among the levels of its items.*

Generalized itemsets whose items have all the same level are called *level-sharing itemsets* [6]. The level of a level-sharing itemset with respect to a taxonomy corresponds to the one of its items.

The support of a generalized itemset evaluates its observed frequency of occurrence in the analyzed data. It is defined in terms of the itemset coverage with respect to the analyzed data.

Definition 5 Generalized itemset coverage. *Let $\mathcal{D}$ be a transactional patient dataset and $\mathcal{T}$ the corresponding taxonomy. A (generalized) itemset I covers a given transaction $t_i \in \mathcal{D}$ if all its (possibly generalized) items $i_k \in I$ are*

either included in t_i or ancestors (generalizations) of items $i_k \in t_i$ with respect to $\mathcal{T}$.

The support of a generalized itemset I is given by the ratio between the number of transactions $t_i \in \mathcal{D}$ covered by I and the cardinality of $\mathcal{D}$.

A (generalized) itemset I is said to be a descendant of another generalized itemset Y if (i) I and Y have the same length (i.e., the same number of items) and (ii) for each item $y \in Y$ there exists an item $i \in I$ that is a descendant of y.

The concept of generalized association rule extends traditional association rules to the case in which they may include either generalized or not generalized itemsets. A more formal definition follows.

Definition 6 Generalized association rule. *Let A and B be two (generalized) itemsets. A generalized association rule is represented in the form $R : A \Rightarrow B$, where A and B are the body and the head of the rule respectively.*

A and B are also denoted as antecedent and consequent of the generalized rule $A \Rightarrow B$. Generalized association rule extraction is commonly driven by rule support (s) and confidence (c) quality indexes [11]. While the support index represents the observed frequency of occurrence of the rule in the source dataset, the confidence index represents the rule strength.

Definition 7 Generalized association rule support. *Let $\mathcal{D}$ be a transactional patient dataset and $\mathcal{T}$ a taxonomy. The support of a generalized rule $R : A \Rightarrow B$ is defined as the support (i.e., the observed frequency) of $A \cup B$ in $\mathcal{D}$.*

Definition 8 Generalized association rule confidence. *Let $\mathcal{D}$ be a transactional patient dataset and $\mathcal{T}$ a taxonomy. The confidence of a rule $R : A \Rightarrow B$ is the conditional probability of occurrence in $\mathcal{D}$ of the generalized itemset B given the generalized itemset A.*

For example, the generalized association rule {(*Examination,Routine*)} $\rightarrow$ {(*Examination,Electrocardiogram*)} (s=60%,c=100%) indicates that examinations belonging to category *Routine* co-occur with examination *Electrocardiogram* in $\frac{3}{5}$ of the transactions of the analyzed dataset (Pids 1, 3, and 5) and the implication holds in $\frac{3}{3}$=100% of the cases.

In some cases, measuring the strength of a rule in terms of support and confidence may be misleading [12]. When the rule consequent is characterized by relatively high support value, the corresponding rule may be characterized by a high confidence even if its actual strength is relatively low. To overcome this issue, the lift (or correlation) index [13] may be used, rather than/beyond the confidence index, to measure the (symmetric) correlation between body and head of the extracted rules.

Definition 9 Generalized association rule lift. *Let $A \rightarrow B$ be an association rule. Its lift is given by*

$$\mathrm{l(A,B)} = \frac{\mathrm{c}(A \rightarrow B)}{\mathrm{s(B)}} = \frac{\mathrm{s}(A \rightarrow B)}{\mathrm{s(A)s(B)}} \tag{1}$$

where $\mathrm{s}(A \rightarrow B)$ and $\mathrm{c}(A \rightarrow B)$ are, respectively, the rule support and confidence, and $\mathrm{s}(A)$ and $\mathrm{s}(B)$ are the supports of the rule antecedent and consequent.

If l(A,B) is equal to or close to 1, itemsets A and B are not correlated with each other. Lift values significantly below 1 show negative correlation, whereas values significantly above 1 indicate a positive correlation between itemsets A and B, i.e., the implication between A and B holds more than expected. For example, rule $\{(Examination,Routine)\} \rightarrow \{(Examination,Electrocardiogram)\}$ is positively correlated, because its lift value is equal to $\frac{5}{3}$.

Since the interest of uncorrelated or negatively correlated rules is marginal in our context of analysis, MeTA only considers frequent and positively correlated generalized association rules. Specifically, given a transactional patient dataset, a taxonomy, a minimum support threshold (*minsup*), and a minimum lift threshold (*minlift*) MeTA discovers all the generalized association rules whose:

- support value is above a given minimum support threshold *minsup*, i.e., s(R)>*minsup*, and
- lift value is above a given minimum lift threshold *minlift*, i.e., l(R)>*minlift*.

The generalized rules that satisfy all the above conditions will be denoted as *strong rules* throughout the deliverable.

Since the set of strong rules may still contain less informative rules, a further pruning step is applied prior to performing further analyses. Specifically, MeTA discovers non-redundant generalized rules [16], which are a worthwhile subset of strong generalized rules. Extensions of a strong generalized rule are classified as redundant if they have the same support and confidence of their specialized version. A more formal definition is given below. It extends the concept of non-redundant rule, first proposed in [16], to the case in which rules may also contain generalized items.

Definition 10 Non-redundant generalized association rule. *Let $R : A \Rightarrow B$ be a strong generalized rule. R is non-redundant if there exists no strong rule $R^* : C \Rightarrow D$, $C \subseteq A \wedge D \subseteq B$ such that the support and confidence of R and R^* are equal.*

To generate non-redundant generalized rules we used the publicly available implementation of the algorithm proposed in [16] on an extended dataset version, in which transactions contain both items and their corresponding generalizations according to the input taxonomy. This generalized rule mining strategy is similar to the one previously adopted in [11] in the context of market basket analysis.

2.3 Rule categorization

The generalized rules extracted during the last MeTA step are explored by domain experts to discover valuable information. Unfortunately, when coping with relatively large or complex transactional patient datasets the number of mined rules could be so large that a manual inspection becomes unfeasible. To overcome this issue, this block focuses on categorizing the extracted rules into homogeneous groups, according to their represented information.

MeTA partitions rules into worthwhile subsets that characterize the underlying data from different viewpoints, because they contain different combinations

Table 2: Rule categories. $*^1$ represents an examination or an examination class, $*^2$ represents either a drug or a drug class $*^3$ represents either an age or an age group, while $*^4$ is a gender value (male or female).

Class ID	Name	Template
E-Rules	*Correlations between examinations*	$\{(Examination,*^1)\} \rightarrow \{(Examination,*^1)\}$
D-Rules	*Correlations between drugs*	$\{(Drug,*^2)\} \rightarrow \{(Drug,*^2)\}$
ED-Rules	*Correlations between examinations and drugs*	$\{(Drug,*^2)\} \rightarrow \{(Examination,*^1)\}$ $\{(Examination,*^1)\} \rightarrow \{(Drug,*^2)\}$ $\{(Examination,*^1),(Drug,*^2)\} \rightarrow \{(Examination,*^1)\}$ $\{(Examination,*^1),(Drug,*^2)\} \rightarrow \{(Drug,*^2)\}$
P-Rules	*Profile-based correlations*	**Age Profiles** $\{(Age,*^3)\} \rightarrow \{(Examination,*^1)\}$ $\{(Age,*^3)\} \rightarrow \{(Drug,*^2)\}$ $\{(Age,*^3),(Examination,*^1)\} \rightarrow \{(Drug,*^2)\}$ $\{(Age,*^3),(Drug,*^2)\} \rightarrow \{(Examination,*^1)\}$ **Gender Profiles** $\{(Gender,*^4)\} \rightarrow \{(Examination,*^1)\}$ $\{(Gender,*^4)\} \rightarrow \{(Drug,*^2)\}$ $\{(Gender,*^4),(Examination,*^1)\} \rightarrow \{(Drug,*^2)\}$ $\{(Gender,*^4),(Drug,*^2)\} \rightarrow \{(Examination,*^1)\}$ **Age-Gender Profiles** $\{(Age,*^3),(Gender,*^4)\} \rightarrow \{(Examination,*^1)\}$ $\{(Age,*^3),(Gender,*^4)\} \rightarrow \{(Drug,*^2)\}$ $\{(Age,*^3),(Gender,*^4)\} \rightarrow \{(Examination,*^1)\}$ $\{(Age,*^3),(Gender,*^4)\} \rightarrow \{(Drug,*^2)\}$

of patient features and/or medical treatments. We highlighted four representative rule classes, which are thoroughly described below. Table 2 reports the rule template for each class.

Class E-Rules: Correlations between examinations. Rules in this group represent correlations among examinations regardless of the characteristics of the analyzed patients and prescribed drugs. For example, $\{(Examination,Routine)\} \rightarrow \{(Examination,Electrocardiogram)\}$ belongs to Class E-Rules. This class may potentially include more complex rules, such as $\{(Examination,Routine), (Examination,Blood\ count)\} \rightarrow \{(Examination,Electrocardiogram)\}$. In other words, rule antecedent can be, in general, itemsets of arbitrary size.

Class D-Rules: Correlations between drugs. Rules in this group focus the experts' attention on correlations among the prescribed drugs, disregarding examinations and patient characteristics. For example, $\{(Drug,Acetylsalicylic\ Acid)\} \rightarrow \{(Drug,Moxifloxacin)\}$ belongs to Class D-Rules. Even in this class rules can represent implications where the antecedent is an itemset of arbitrary size.

Class ED-Rules: Correlations between examinations and drugs. This group of rules represents co-occurrences between drugs and examinations into the patient dataset, regardless of patient characteristics. More specifically, all the rules that contain both examinations/examination category and drugs/drug categories into their antecedent/consequent are assigned to class ED-Rules. For example,
$\{(Examination,Routine), (Drug,Aspirin)\} \rightarrow \{(Examination,Electrocardiogram)\}$
is assigned to this class. It indicates the association between the co-occurrence of an examination category and a drug and a specific examination.

Class P-Rules: Profile-based correlations. The former rule classifications disregard patient characteristics. Nevertheless, experts can deem such information to be useful for characterizing specific user profiles (e.g., elderly men, kids). This class consists of all the rules that contain any item related to a census feature in their rule antecedent. This rule subset can be further categorized according to the considered census features, because each combination of patient census features may represent a distinct and potentially meaningful user profile. Since in our work we target our analysis on age and gender census features, rules belonging to Class P-Rules can be partitioned into the three subgroups reported in Table 2.

For example, {(*Age,Elder*), (*Gender,Male*)} → {(*Examination,HDL Cholesterol*)} indicates that the *HDL Cholesterol* examination has frequently been prescribed to elderly men. Similarly, rule {(*Age,Elder*), (*Drug,Acetylsalicylic Acid*)} → {(*Examination,HDL Cholesterol*)} indicates that the *HDL Cholesterol* examination has frequently been prescribed to elderly people (males or females) who have taken drug Acetylsalicylic Acid (regardless of the temporal order of drug/examination prescriptions).

2.3.1 Level-wise exploration of rule categories

Given a worthy set of rule categories, experts are asked to go into detail about the contained rules. However, since generalized rules potentially represent information at different levels of granularity, rule class exploration could be challenging unless considering taxonomy abstraction levels as reference information.

To easily explore rule categories, the corresponding rules are further classified as high-, cross-, or low-level according to the level of abstraction of the contained information in the input taxonomy.

High-level rules are generalized rules $A \rightarrow B$, where A and B are level-sharing itemsets with the same level $l > 1$. They typically represent general knowledge and thus they should be considered first during manual result exploration.

For example, {(*Examination,Routine*)} → {(*Examination,Cardiovascular examination*)} is a high-level rule, because both rule antecedent and consequent are level-2 itemsets.

Cross-level rules are generalized rules $A \rightarrow B$, where A and B are either not level-sharing itemsets or level-sharing itemsets with different level. They combine detailed and general information by climbing up and down the taxonomy for different data features. Given a subset of high-level rules, cross-level rules can be considered as an intermediate step to perform drill-down (i.e., moving from general to detailed information).

For example, {(*Examination,Blood count*)} → {(*Examination,Cardiovascular examination*)} is a cross-level rule, because the rule antecedent is a level-1 itemset, whereas the rule consequent is a level-2 itemset. If the high-level rule {(*Examination,Routine*)} → {(*Examination,Cardiovascular examination*)} is deemed to be useful for advanced analysis, then considering the former cross-level rule can be relevant to analyze the underlying correlations between a specific routine examination and the Cardiovascular examination category.

Low-level rules are not generalized rules $A \rightarrow B$, i.e., both A and B are not generalized (level-1) itemsets. They typically represent very detailed knowledge. When coping with relatively sparse datasets, many of these rules could be discarded during the mining process by enforcing the minimum support thresh-

old. However, their peculiar information is likely to be covered, to a certain extent, by cross- and high-level rules. For example, {(*Examination,Urine test*)} $\rightarrow$ {(*Examination,Electrocardiogram*)} is an example of low-level rules. These rules can be analyzed to gain more insights on a specific subset of cross-level or high-level rules.

3 Experimental results

We performed various experiments on a real-life dataset collected by an Italian Health Center to demonstrate effectiveness and efficiency of the MeTA framework.

All the experiments were performed on a quad-core 3.30 GHz Intel Xeon workstation with 16 GB of RAM, running Ubuntu Linux 12.04 LTS. The software used to perform rule extraction and post-processing is available online at [9].

3.1 Diabetic patient dataset and taxonomy

The dataset considered in this study was collected by an Italian Local Health Center (LHC). Specifically, in 2007 they collected into a unique LHC dataset all the accesses to the medical center year-round. Then, from the LHC dataset the examination log data of all the patients with overt diabetes were extracted. Raw data consist of 95,788 records and they include examinations and drugs prescribed to 3,565 patients. The dataset contains information about male and female patients in a wide age range (i.e., between 4 and 95 years). To analyze diabetes complications at various degrees of severity both routine and more specific examinations were recorded jointly with prescribed drugs. The diagnostic and therapeutic procedures were defined using the ICD 9-CM (International Classification of Diseases, 9th revision, Clinical Modification) [7]. Drugs were identified by the pharmaceutical coding system adopted by the Anatomical Therapeutic Chemical (ATC) Classification System [4].

The generalization hierarchy over examinations is shown in Table 3. It contains 26 examinations clustered into 7 examination categories. The selected examination categories are based on the expert-driven classification reported in [3].

The drug generalization hierarchy contains as leaves the drugs encoded by using the fifth level of the ATC classification system defined in [4]. Drugs are aggregated into the corresponding drug category, according to the first level of the standard ATC classification system. For instance, drug acetylsalicylic acid (i.e., code: B01AC06) is a leaf node of the drug generalization hierarchy and Category B (i.e., Category Blood and blood forming organs) is its generalization. Our dataset contains 200 distinct drugs and 14 distinct categories. Table 4 reports the hierarchy defined over drugs.

Human life is often divided into various age ranges (e.g., infancy, middle-adulthood, old age). Age feature values have been discretized into the following 8 age groups, which represent established ranges of the human lifespan [15]: [0-6], [7-12], [13-22], [23-39], [40-59],[60-75], [76-90], and [91-101].

Table 3: Generalization hierarchy over examinations.

Examination category	**Examination**
Routine examinations	*Checkup visit* *Glucose level* *Urine test* *Venous blood* *Complete blood count* *Hemoglobin*
Cardiovascular examinations	*Electrocardiogram* *Cholesterol* *HDL Cholesterol* *Triglycerides*
Eye examinations	*Fundus oculi* *Angioscopy* *Complete eye examination* *Retinal photocoagulation*
Liver examinations	*AST* *ALT* *Bilirubin* *Gamma GT*
Kidney examinations	*Urin acid* *Microscopic urine analysis* *Culture urine* *Creatinine clearance* *Creatinine* *Microalbuminuria*
Carotid examinations	*ECO Doppler carotid*
Limb examinations	*ECO Doppler limb*

3.2 Analysis of the mined rules

We performed several generalized rule extractions from the patient dataset by enforcing different minimum support (*minsup*) and lift (*minlift*) thresholds. To perform knowledge discovery from the mined rules, we selected a representative configuration setting, i.e., we set *minsup* to 1% and *minlift* to 1.1. The reasons behind the choice of the support threshold are twofold. Firstly, too low/high support threshold values yield very detailed/general rule sets and thus they may not produce manageable yet interesting knowledge. Secondly, it is well-known that averagely low-support rules commonly represent potentially interesting knowledge if they represent positive correlations among items (i.e., their lift is above 1) [5]. Nevertheless, by generalizing items at higher abstraction levels some of the low-support correlations among data are still represented by higher-level rules. Hence, we selected *minsup*=1% as a good trade-off between rule set specialization and generality. We also enforced a minimum lift threshold equal to 1.1 to prune both negatively correlated and uncorrelated item combinations. On the one hand, the interest of negatively correlated rules (i.e., rules with lift below 1) is marginal in our context of analysis. On the other hand, rules with lift close to 1 are misleading because their occurrences are not actually correlated with each other. Hence, among the positively correlated rules we further pruned approximately 10% of them whose lift value is between 1 and 1.1. Finally, we focused on the rules with length below or equal to 3, i.e., the rules consisting of pairs or triples of (generalized) items, because they represent the most actionable correlations among drugs/examinations. However, to specialize the rules that provide peculiar information, experts could performed further extractions and explore longer rules complying with the given category.

We first analyzed the rules than hold for all patients, i.e., we considered rules belonging to Classes E-Rules, D-Rules, and ED-Rules (Section 2.3). Then, we

Table 4: Generalization hierarchy over drugs.

Drug category	**Drug**
Category A: Alimentary tract and metabolism	*A01AA01: Sodium fluoride* *A05AX01: Piprozolin* *...*
Category B: Blood and blood forming organs	*B01AC06: Acetylsalicylic acid* *B03AA03: Ferrous gluconate* *...*
Category C: Cardiovascular system	*C09AA05: Ramipril* *C10AA07: Rosuvastatin* *...*
Category D: Dermatologicals	*D01AA02: Natamycin* *D01AA03: Hachimycin* *...*
Category G: Genito-urinary system and sex hormones	*G04CB01: Finasteride* *G04CX03: Mepartricin* *...*
Category H: Systemic hormonal preparations, excluding sex hormones and insulins	*H02AA02: Fludrocortisone* *H02AB07: Prednisone* *...*
Category J: Antiinfectives for systemic use	*J01MA12: Levofloxacin* *J02AC04: Posaconazole* *...*
Category L: Antineoplastic and immunomodulating agents	*L01AA07: Trofosfamide* *L01AB01: Busulfan* *...*
Category M: Musculo-skeletal system	*M03AC10: Mivacurium chloride* *M03BA05: Febarbamate* *...*
Category N: Nervous system	*N04AA02: Biperiden* *N04AB01 Etanautine* *...*
Category P: Antiparasitic products, insecticides and repellents	*P01AA04: Chlorquinaldol* *P01AC01: Diloxanide* *...*
Category R: Respiratory system	*R03AC02: Salbutamol* *R03BA02: Budesonide* *...*
Category S: Sensory organs	*S02AA10: Acetic acid* *S02BA03: Prednisolone* *...*
Category V: Various	*V10XX01: Sodium phosphate* *V10XA01: Sodium iodide* *...*

Table 5: Examples of correlations between examinations (Class E-Rules).

ID	Rule	Sup%	Conf%	Lift	Type
1	$\{(Examination, Liver)\} \rightarrow \{(Examination, Kidney)\}$	30.8%	94.3%	2.52	High-leve
2	$\{(Examination, Kidney)\} \rightarrow \{(Examination, Liver)\}$	30.8%	82.2%	2.52	High-leve
3	$\{(Examination, Liver)\} \rightarrow \{(Examination, Cardiovascular)\}$	30.8%	94.4%	1.97	High-leve
4	$\{(Examination, Cardiovascular)\} \rightarrow \{(Examination, Liver)\}$	30.8%	64.4%	1.97	High-leve
5	$\{(Examination, Kidney)\} \rightarrow \{(Examination, Cardiovascular)\}$	33.7%	90.1%	1.88	High-leve
6	$\{(Examination, Cardiovascular)\} \rightarrow \{(Examination, Kidney)\}$	33.7%	70.5%	1.88	High-leve
7	$\{(Examination, Liver), (Examination, Cardiovascular)\} \rightarrow \{(Examination, Kidney)\}$	29.3%	95.1%	2.54	High-leve
8	$\{(Examination, Liver)\} \rightarrow \{(Examination, Uric\ acid)\}$	24.4%	74.8%	2.85	Cross-leve
9	$\{(Examination, Liver)\} \rightarrow \{(Examination, Microscopic\ urine\ analysis)\}$	21.7%	66.3%	2.62	Cross-leve
10	$\{(Examination, Liver)\} \rightarrow \{(Examination, Culture\ urine)\}$	21.6%	66.2%	2.72	Cross-leve
11	$\{(Examination, Liver)\} \rightarrow \{(Examination, Creatinine\ clearance)\}$	16.5%	50.7%	2.79	Cross-leve
12	$\{(Examination, Liver)\} \rightarrow \{(Examination, Microalbuminuria)\}$	13.1%	40.1%	2.79	Cross-leve
13	$\{(Examination, Liver)\} \rightarrow \{(Examination, Creatinine)\}$	12.7%	38.7%	2.69	Cross-leve
14	$\{(Examination, Bilirubin)\} \rightarrow \{(Examination, Uric\ acid)\}$	1.4%	82.0%	3.12	Low-level
15	$\{(Examination, AST)\} \rightarrow \{(Examination, Uric\ acid)\}$	24.0%	75.3%	2.87	Low-level
16	$\{(Examination, ALT)\} \rightarrow \{(Examination, Uric\ acid)\}$	24.3%	75.2%	2.86	Low-level
17	$\{(Examination, Gamma\ GT)\} \rightarrow \{(Examination, Uric\ acid)\}$	5.3%	69.7%	2.66	Low-level

focused on rules that concern patient profiles (i.e., Class P-Rules).

3.2.1 Correlations between examinations and drugs

Tables 5 and 6 report worthwhile subsets of correlations between sets of examinations and drugs, respectively. The former rules represent potentially interesting correlations among examinations, whereas the latter correlations among drugs. A worthy subset of correlations between examinations and drugs (Class ED-Rules) is summarized in Table 7. For each rule, we reported support (percentage), confidence (percentage), lift, and the corresponding type, according to the level-dependent classification reported in Section 2.3.1 (low-level, cross-level, high-level).

Analysis of correlations between examinations (Class E-Rules). This section addresses the analysis of a subset of interesting correlations between examinations. First, we are particularly interested in analyzing the co-occurrences among examination categories, while disregarding the temporal order of prescriptions. High-level rules, such as rules (1)-(7) in Table 5, represent positive correlations between examination categories. They can be used to target the analysis towards specific issues. For example, rules (1) and (2) highlight a pairwise association between liver and kidney examinations, which hold 2.52 times more than expected according to the corresponding lift value[1]. In other words, the expected frequency of co-occurrence of the two examination category (assuming the independence between the occurrences of the single examination categories) is significantly lower than the observed one. The high-level rules (1) and (2) can be used to efficiently schedule medical examination timetables according to their corresponding prescriptions. For example, since liver and kidney examinations are frequently prescribed to the same patient, scheduling both examinations at the same day could reduce patient recovery time. A deeper insight into liver and kidney examinations may be focused on (i) assessing the adherence of medical treatments to the medical guidelines suggested by the Italian Ministry of Health about liver and kidney diseases in diabetic patients or (ii) proposing new guidelines according to the observed correlations between spe-

[1]The lift value of the two rules is the same because of the symmetry of the lift measure [13].

cific liver and kidney examinations. Similar analyses can be performed starting from the pairwise correlations between the examination categories represented by rules (3)-(6). Since association rules can also represent higher-order associations among data, we should not restrict our analyses to pairwise associations among items. For example, rule (7) shows a positive correlation between liver, cardiovascular, and kidney examination categories. Longer rules can be used either to specialize known lower-order associations or to figure out new and more complex medical treatments.

To deepen into the analysis of the most specific correlations between examinations, high-level rules are often not enough. In fact, they provide a high-level view of the underlying correlations among data, which could be insufficient to perform targeted analysis. On the other hand, as discussed in the following, high-level rules are very important because they also represent those patterns that have not been separately extracted at lower abstraction levels because they are infrequent according to the support threshold.

A step forward is to consider also cross-level rules, which contain both low- and high-level information, i.e., examinations and examination categories, at the same time. To take advantage of the preliminary analysis of high-level rules, only the subset of cross-level rules that are related to some interesting high-level rule are considered. For example, based on rule (1), we can deepen our analysis into the search of underlying correlations between specific examinations and examination categories. For instance, given the subset of patients to whom liver examinations are frequently prescribed, what specific kidney examination is most likely to be frequently prescribed as well? From the comparison between the confidences of rules (8)-(13), uric acid appears to be the most likely kidney examination, because to 74.8% of the patients associated with a liver examination the uric acid examination has been prescribed as well. This information is worthy because it gives more insights into a subset of medical treatments. Similarly, other combinations of examinations and examination categories (which have been omitted for the sake of brevity) have been mined.

The last step is the analysis of low-level rules, which represent significant correlations among single examinations (disregarding the examination categories). The exploration of low-level rules is often a challenging task, because their cardinality is commonly so large that their manual inspection becomes practically unfeasible. To overcome this issue, we early pruned redundant rules (see Section 6) and we exploited the knowledge extracted from higher-level patterns (i.e., high- and cross-level rules) to prevent experts from exploring the whole rule set. For example, given rules (1) and (8), experts may wonder what is the probability of prescribing the uric acid examination to patients who have also received a prescription for a specific liver examination. To answer this question, we can consider low-level rules (14)-(17). Specifically, their confidence values indicate the conditional probability of prescription of the uric acid examination given the occurrence of specific liver examinations in the patient dataset.

Since patient data typically contain not only examination prescriptions but also drug prescriptions, it could be also interesting to analyze the correlations between drugs (i.e., Class D-rules) and the correlations between examinations and drugs (i.e., Class ED-Rules) at different abstraction levels.

Analysis of correlations between drugs (Class D-Rules). Table 6 reports a selection of correlations between drugs. They concern the pairwise correlation between the drugs belonging to the respiratory system category (category

Table 6: Examples of correlations between drugs (Class D-Rules).

ID	Rule	Sup%	Conf%	Lift	Type
1	$\{(Drug, Category\ R)\} \rightarrow \{(Drug, Category\ J)\}$ R = Respiratory system J = Anti-infectives for systemic use	12.5%	77.3%	1.46	High-level
2	$\{(Drug, Category\ J)\} \rightarrow \{(Drug, Category\ R)\}$	12.5%	23.7%	1.46	High-level
3	$\{(Drug, Category\ R)\} \rightarrow \{(Drug, Levofloxacin)\}$	3.5%	21.5%	1.94	Cross-level

Table 7: Examples of correlations between drugs and examinations (Class ED-Rules).

ID	Rule	Sup%	Conf%	Lift	Type
1	$\{(Examination, Carotid)\} \rightarrow \{(Drug, Category\ B)\}$ B = Blood and blood forming organs	3%	68%	1.55	High-level
2	$\{(Examination, Carotid)\} \rightarrow \{(Drug, Acetylsalicylic\ acid)\}$	2%	61%	1.94	Cross-level
3	$\{(Examination, HDL\ Cholesterol)\} \rightarrow \{(Drug, Rosuvastatin)\}$	3.2%	9.4%	1.26	Cross-level

R) and those belonging to the anti-infectives for system use category (category J). The contemporary use of drugs belonging to the above categories could prompt a detailed analysis of the corresponding guidelines [1]. Specifically, rule (3) highlights the association between the drugs belonging to the respiratory system category and drug Levofloxacin, which is commonly prescribed for infections of the respiratory system.

Analysis of correlations between examinations and drugs (Class ED-Rules). Guidelines commonly indicate established associations between examinations and drugs [1]. Their adherence could be verified against correlations between examinations and drugs mined from the real log patient data. Representative rules of this type are reported in Table 7. For example, the high-level rule (1) in Table 7 indicates a positive correlation between the examinations of the carotid and category B drugs (Blood and blood forming organs). This rule confirms the common knowledge that vascular diseases, such as problems to the carotid, are usually taken under control with drugs related to blood diseases. More specifically, the cross-level rule (2) indicates that carotid examinations are frequently associated with the category B drug with code B01AC06, which corresponds to the active principle Acetylsalicylic Acid. Acetylsalicylic Acid [4] is widely used to treat blood and vascular diseases in general (including carotid issues). Hence, the drug use appears to be coherent with guidelines. Finally, the low-level rule (3) in Table 7 shows another interesting correlation between examinations and drugs. Unlike the former ones, it associates a specific examination (the HDL Cholesterol cardiovascular examination) with a specific drug (active principle: rosuvastatin, code: C10AA07). Rosuvastatin is indicated for cardiovascular diseases and, in particular, it is used to treat patients affected by primary hypercholesterolemia [8].

3.2.2 Profile-based correlations (Class P-Rules)

In this section we analyze the rules representing correlations between user profiles (i.e., demographic features) and treatments. These rules represent recurrences among treatments that hold for specific patient segments identified by census features (i.e., age, gender). To facilitate the analysis of different data facets, profile-based rules are further specialized into three subcategories: Age profile, Gender profile, and Age-Gender profile-based rules. A worthwhile sub-

Table 8: Examples of correlations between user profiles, drugs, and examinations (Class P-Rules).

ID	Rule	Sup%	Conf%	Lift	Type
1	$\{(Age,[40\text{-}59])\} \rightarrow \{(Examination,Cardiovascular)\}$	14.8%	70.1%	1.47	High-level
2	$\{(Age,[40\text{-}59])\} \rightarrow \{(Drug,Rosuvastatin)\}$	2.3%	11.0%	1.48	Cross-level
3	$\{(Age,[40\text{-}59])\} \rightarrow \{(Drug,Ramipril)\}$	2.4%	11.6%	1.22	Cross-level
4	$\{(Age,[40\text{-}59]), (Examination,Cardiovascular)\} \rightarrow \{(Drug,Rosuvastatin)\}$	1.82%	12.3%	1.65	Cross-level
5	$\{(Age,[40\text{-}59]), (Examination,HDL\ Cholesterol)\} \rightarrow \{(Drug,Rosuvastatin)\}$	1.68%	13.5%	1.82	Cross-level
6	$\{(Gender,Male)\} \rightarrow \{(Drug,Finasteride)\}$	1.0%	1.9%	1.96	Cross-level

set of representative rules is reported in Table 8. Considering these rules allow us to characterize patients with different profiles (i.e., age and/or gender) based on their prescribed examinations and drugs.

Rules (1)-(5) in Table 8 have been classified as "Age profiles" rules (see Section 2.3), because patients are clustered into segments according to their age. For example, rule (1) indicates that diabetic patients in the age range [40-59] (i.e., middle-aged patients) are used to undergo cardiovascular examinations. The implication holds for most of the patients belonging to the segment (rule confidence 70.1%). Guidelines confirm that middle-aged diabetic patients are expected to undergo examinations in order to prevent cardiovascular diseases [1]. Furthermore, rules (2) and (3) in Table 8 indicate a positive correlation between middle-aged patients and drugs Rosuvastatin and Ramipril, respectively. Both drugs are likely to be prescribed to patients with cardiovascular diseases in conjunction with specific examinations. Drug Rosuvastatin is mainly used to treat patients with primary hypercholesterolemia, whereas Ramipril (code: C09AA05) is commonly prescribed to reduce blood pressure [4]. The confidence values of rules (2) and (3) indicate that approximately 11% of middle-aged patients actually take the specific drugs. Based on the achieved results, drug provision across medical centers and pharmacies could be shaped according to the patient age distribution. For example, medical centers that mainly treat middle-aged or elderly patients would purchase large amounts of these drugs. It is worth noticing that, to perform such analyses, discarding low-confidence rules would be harmful because they still provide information valuable for medical resource management. If we focus on middle-aged diabetic patients (age group [40-59]) to whom the HDL cholesterol examination has been prescribed at least once (see Rule (3)), then the percentage of patients who have also taken drug Rosuvastatin significantly increases with respect to all middle-aged patients (rule confidence 13.5% against 11.0%) and even the rule correlation increases (rule lift 1.82 against 1.48). Rule (6) represents a correlation between male patients and drug Finasteride. The rule appears to be reliable, because drug Finasteride is used for treatment and control of benign prostatic hyperplasia, which commonly arises in male.

4 Conclusions

This deliverable presents a novel approach to analyzing multiple-level correlations among medical datasets equipped with taxonomies. Since patient log dataset are often relatively sparse, discovering valuable correlations among multiple patient data features could be a challenging task. To overcome this issue, we propose to discover, categorize, and analyze non-redundant generalized as-

sociation rules, which represent worthy multiple-level associations among data items.

The experiments, performed on a real diabetic patient dataset, highlight correlations among treatments and patient profiles which are consistent with the guidelines for diabetes disease [1, 7]. Furthermore, the extracted high-level rules represent fruitful information commonly discarded by traditional rule mining approaches.

References

[1] ADA. American Diabetes Association Standards of Medical Care in Diabetes 2013. *Diabetes Care*, 36(Supplement 1):S11–S66, January 2013.

[2] R. Agrawal, T. Imieliński, and A. Swami. Mining association rules between sets of items in large databases. In *ACM SIGMOD Record*, volume 22, pages 207–216. ACM, New York, 1993.

[3] Dario Antonelli, Elena Baralis, Giulia Bruno, Tania Cerquitelli, Silvia Chiusano, and Naeem Mahoto. Analysis of diabetic patients through their examination history. *Expert Systems with Applications*, 40(11):4672 – 4678, 2013.

[4] ATC. Norwegian-Institute-of-Public-Health: ATC/DDD Index 2013. Available: http://www.whocc.no/atc_ddd_index/ . Last access on November 2013, 2013.

[5] Sergey Brin, Rajeev Motwani, and Craig Silverstein. Beyond market baskets: Generalizing association rules to correlations. In *Proceedings of the 1997 ACM SIGMOD International Conference on Management of Data*, SIGMOD '97, pages 265–276, New York, NY, USA, 1997. ACM.

[6] J. Han and Y. Fu. Mining multiple-level association rules in large databases. *IEEE Transactions on Knowledge and Data Engineering*, 11(5):798–805, 2002.

[7] I. ICD-9-CM. International Classification of Diseases, 9th revision, Clinical Modification. Available: http://icd9cm.chrisendres.com. Last access on March 2011, 2011.

[8] IDF. International Diabetes Federation. Available: http://www.idf.org/. Last access on November 2013, 2013.

[9] MeTA. http://dbdmg.polito.it/wordpress/wp-content/uploads/2014/05/META.zip Last access: May 2014, 2014.

[10] N. Pasquier, Y. Bastide, R. Taouil, and L. Lakhal. Discovering frequent closed itemsets for association rules. *Database Theory*, pages 398–416, 1999.

[11] R. Srikant and R. Agrawal. Mining generalized association rules. In *International Conference on Very Large Data Bases*, pages 407–419. Morgan Kaufmann, San Fransisco, 1995.

[12] P.-N. Tan and V. Kumar. Interestingness measures for association patterns: A perspective. *KDD 2000 Workshop on Postprocessing in Machine Learning and Data Mining*, 2000.

[13] P.N. Tan, V. Kumar, and J. Srivastava. Selecting the right interestingness measure for association patterns. In *Proceedings of the eighth ACM SIGKDD international conference on Knowledge discovery and data mining*, page 41. ACM, New York, 2002.

[14] P.N. Tan, M. Steinbach, V. Kumar, et al. *Introduction to data mining*. Pearson Addison Wesley, Boston, 2006.

[15] P.S. Timiras. *Physiological Basis of Aging and Geriatrics*. Taylor & Francis, 2013.

[16] MohammedJ. Zaki. Mining non-redundant association rules. *Data Mining and Knowledge Discovery*, 9(3):223–248, 2004.

Section 2.4 — Towards a statistical framework for attribute comparison in very large relational databases

Cesare Alippi, Elisa Quintarelli, Manuel Roveri, and Letizia Tanca

DEIB – Politecnico di Milano

1 Introduction

The technological evolution and the multiplication of information sources has brought about an ever-increasing need of techniques for the analysis of large-scale databases. The recent research, generally collected under the umbrella of "Big Data Analytics", attempts to solve this many-sided problem. We briefly describe a general methodology for the statistical analysis of large-scale databases with the aim to extract relevant, often implicit or unexpected, information about the distribution of the attribute values in two (large) tuple sets resulting from different queries on a large database. This analysis has the main aim of helping users to gain knowledge about the datasets they are exploring (see [2]). While a relatively large literature addressing a similar problem exists under the name of *subgroup discovery* (e.g., [6], [3], [1], [4] just to name a few), our framework presents the following distinctive features: 1) it manages both categorical and numerical attributes; 2) it represents subgroups as SQL queries; 3) the classification of attributes into unusualness or interest comprises statistical hypothesis tests and the Hellinger distance; 4) the search of relevant attributes relies on the joint use of sampling and incremental mechanisms for statistical hypothesis tests. We emphasize that the proposed framework, which is here presented for subgroup discovery, could be also considered for supervised descriptive rule discovery.

2 The general methodology

Assume we are given a relational database schema $\mathcal{R} = \{R_1, \ldots, R_k\}$, and an instance I of $\mathcal{R}$; we denote with $Att(R_i)$ the set of attributes of each relation $R_i \in \mathcal{R}$ and call *tupleset* the result $Q(I)$ of any conjunctive query Q applied to I. $Att(Q(I))$ are the attributes of $Q(I)$. We consider a couple of *independent queries* over I, i.e., Q_1 and Q_2, and the corresponding tuplesets $Q_1(I)$ and $Q_2(I)$. The analysis of *dependent* queries (i.e., queries that can be obtained one from the other by applying selections, projections or join operations) can be easily brought back to the case of independent ones. We call p_1 and p_2 the cardinalities of $Att(Q_1(I))$ and $Att(Q_2(I))$, respectively, and N_1 and N_2

the cardinalities of $Q_1(I)$ and $Q_1(I)$. We also assume that $Q_1(I)$ and $Q_2(I)$ share one or more attributes, i.e., $S = Att(Q_1(I)) \cap Att(Q_2(I)) \neq \varnothing$. The motivating idea of this work is the possibility to identify whether S contains any attribute A whose data "behave" differently, from a statistical viewpoint, in the two tuplesets. Following the *large-scale* assumption on I, we encompass the situation where N_1 and N_2 are so large that analyzing the whole $Q_1(I)$ and $Q_2(I)$ on-the-fly might be infeasible. The proposed statistical framework relies on the following four steps:

Extraction The *extraction* step aims at extracting a subset of tuples from $Q_1(I)$ and $Q_2(I)$. Let $\mathcal{E}$ be the extraction mechanisms (e.g., sequential extraction, random extraction or hybrid approaches); we denote with $q_i = \mathcal{E}(Q_i(I))$, $i \in \{1, 2\}$, a set of tuples extracted from $Q_i(I)$ through $\mathcal{E}$, where the key idea is that the cardinality n_i of q_i is much smaller than N_i.

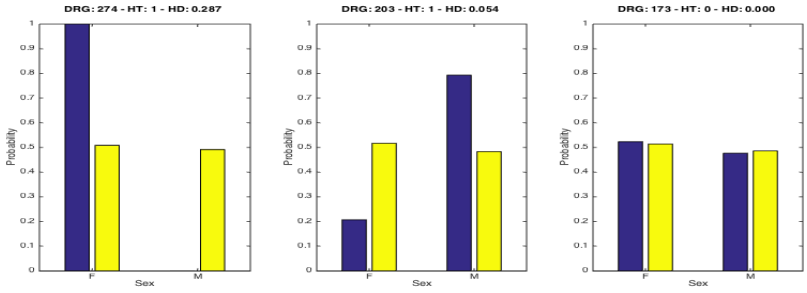
Comparison Let X_1 and X_2 be the projections of q_1 and q_2 over a specific attribute $A \in S$. Data in X_1 and X_2 can be either numerical or categorical. The comparison step aims at assessing the discrepancy between X_1 and X_2 through theoretically-grounded statistical Hypothesis Tests (HTs), i.e., $\mathcal{H}_{X_1,X_2}$. Our statistical framework encompasses HTs able to inspect variations both in numerical and categorical data (i.e., *two-sided t-test*, *two-sided Wilcoxon rank sum test*, *two-sample Kolmogorov-Smirnov test* and *two-sample Chi-square test*).

Incremental procedure The core mechanism of the incremental procedure consists in repeatedly running the extraction and comparison steps M times: at the j-th iteration, a new couple of subsets q_1 and q_2 is extracted, X_1 and X_2 are computed and $\mathcal{H}_{X_1,X_2}$ is evaluated. If the test rejects the null hypothesis, we stop since we have enough statistical confidence that there is a difference in the data distributions of attribute A in $Q_1(I)$ and $Q_2(I)$. Otherwise, the procedure proceeds to the next iteration. The procedure terminates at the M-th iteration. We consider the Bonferroni correction to keep the Type-I error of this ensemble of M HTs under control. When $M = 1$ the framework behaves as per the traditional single HT.

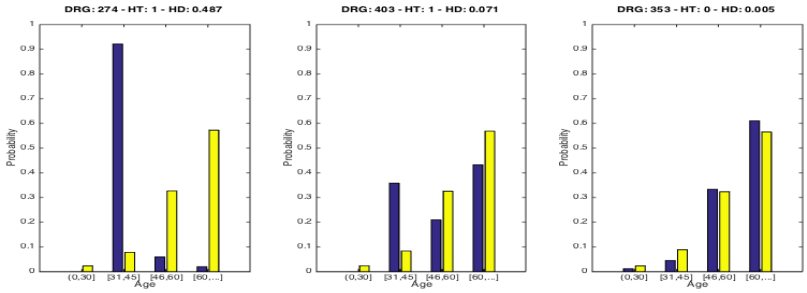
Query ranking The above procedure can be iterated over a set of possible query couples and the *Hellinger distance* between the empirical distributions is computed (measuring the difference between the distributions) either on the sampled or the whole dataset. Afterwards, a ranking of the query couples according to their Hellinger distance allows to highlight those exhibiting the largest differences. Other dissimilarity measures could have been considered as well (e.g., see [5]).

3 Experimental Section

To evaluate the effectiveness of the proposed approach we consider a real-world, rather large dataset (135 attributes and more than 13500 tuples) with hospital information including the patient profiles, the diagnoses, the patients' wards and values of their blood tests. In this preliminary analysis we have focused on a subset of the attributes, i.e., SEX (2 values), quantized AGE (4 values) and diagnosis code DRG (150 values). No sampling mechanism has been applied and $M = 1$. The two-sample Chi-square test has been used as HT and the results of all possible queries of the forms *SELECT SEX FROM db WHERE DRG = drg_code* and *SELECT AGE FROM db WHERE DRG = drg_code* have



(a) Exploration of patient sex w.r.t. disease index DRG (DRG legend: 274-breast neoplasm; 203-pancreas neoplasm; 173-digestive system neoplasm))



(b) Exploration of patient age w.r.t. disease index DRG (DRG legend: 274-breast neoplasm; 403-leukemia; 353-pelvic disease))

been ranked according to the Hellinger Distance. Examples of these rankings are in the Figure above, which shows the difference in the empirical distribution between the tuples extracted with a specific disease index DRG (blue bars) and its complement (yellow bars) for the six queries of the following Table:

QUERY	HT	HD
SELECT SEX FROM db WHERE DRG = '274' (breast neoplasm)	1	0.287
SELECT SEX FROM db WHERE DRG = '203' (pancreas neoplasm)	1	0.054
SELECT SEX FROM db WHERE DRG = '173' (digestive system neoplasm)	0	0.000
SELECT AGE FROM db WHERE DRG = '274' (breast neoplasm)	1	0.487
SELECT AGE FROM db WHERE DRG = '403' (leukemia)	1	0.071
SELECT AGE FROM db WHERE DRG = '353' (pelvic disease)	0	0.005

where HT represents the output of the hypothesis test (0: No statistical difference; 1: Statistical difference); and HD is the Hellinger Distance.

As expected, the breast neoplasm almost completely affects women, while pancreas neoplasm mostly affects men. Interestingly, as regards the digestive system neoplasm, there is no statistical difference between women and men. Furthermore, experimental results show that breast neoplasm behaves statistically different than other neoplasms in terms of age of the patient (i.e., mostly affecting people with age between 30 and 45). On the contrary, there is no statistical difference among the patient's age between the pelvic disease and the other neoplasms.

References

[1] Martin Atzmueller and Frank Puppe. Sd-map–a fast algorithm for exhaustive subgroup discovery. In *Knowledge Discovery in Databases: PKDD 2006*, pages 6–17. Springer, 2006.

[2] Nicoletta Di Blas, Mirjana Mazuran, Paolo Paolini, Elisa Quintarelli, and Letizia Tanca. Exploratory computing: a challenge for visual interaction. In *Int. Work. Conf. on Advanced Visual Interfaces, AVI' 14, Como, Italy, May 27-29, 2014*, pages 361–362, 2014.

[3] Franciso Herrera, Cristóbal José Carmona, Pedro González, and María José del Jesus. An overview on subgroup discovery: foundations and applications. *Know. and Inf. Systems*, 29(3):495–525, 2011.

[4] Branko Kavšek and Nada Lavrač. Apriori-sd: Adapting association rule learning to subgroup discovery. *Applied Artificial Intelligence*, 20(7):543–583, 2006.

[5] Donato Malerba, Floriana Esposito, Vincenzo Gioviale, and Valentina Tamma. Comparing dissimilarity measures for symbolic data analysis. *Proceedings of Exchange of Technology and Know-how and New Techniques and Technologies for Statistics*, 1:473–481, 2001.

[6] Maitreya Natu and Girish Keshav Palshikar. Interesting subset discovery and its application on service processes. In *Data Mining Workshops (ICDMW), 2010 IEEE Int. Conf. on Data Mining*, pages 1061–1068, 2010.