Politecnico di Milano

Dipartimento di Elettronica, Informazione e Bioingegneria
Doctoral Programme in Computer Science and Engineering
2011 - 26th Cycle

Computational Prediction of Gene Functions through Machine Learning methods and Multiple Validation Procedures

DOCTORAL DISSERTATION OF

Davide Chicco

davide.chicco@mail.polimi.it

Supervisor: Marco Masseroli

Tutor: Barbara Pernici
Chair of the doctoral programme: Carlo Fiorini

16^{th} January, 2014

"If you torture data enough, they will eventually confess."

Xiao-Li Meng

Acknowledgements

I would like to thank my supervisors, Marco Masseroli and Stefano Ceri, and my tutor, Barbara Pernici, for the help and the support provided me during these three years.
In addition, I thank the following colleagues and friends that helped me in understanding some thorny parts of the present project:
- Pietro Pinoli for the query design in the database comparison validation chapter;
- Arif Canakoglu for the GPDW data warehouse chapter;
- Eleonora Ciceri, Marco Tagliasacchi and Davide Martinenghi for the Kendall and Spearman correlation list chapter;
- Roberto Sarati for the initial version of the software;
- Fernando Palluzzi for the biological validation and interpretation chapter;
- the thesis reviewers, Elena Marchiori (Radboud Universiteit Nijmegen) and Jean-Philippe Vert (Mines ParisTech), for their valuable corrections and suggestions.

A special thank to my Master of Science thesis advisor, Francesco Masulli (University of Genoa), that suggested me to apply for this doctoral program at Politecnico di Milano.
Many thanks, finally, to the free communities of StackOverflow.com, CrossValidated, MetaOptimize.com/qa: they are the lifelines of every computer scientist.

Contents

1 Abstract

Genes are the most important and essential molecular units of a living organism, and the knowledge of their functions is a crucial key in the understanding of physiological and pathological biological processes, and in the development of new drugs and therapies. This association between a gene and its function has been named as *biomolecular annotation*. Unfortunately, the discovery of new annotations is often time-consuming and expensive, because so are biological *in vitro* experiments carried out by physicians and biologists.
Rapid advances in high-throughput technology have been making many new gene functions available online in public databases and data banks. Despite their undeniable importance, these data sources cannot be considered neither complete nor totally accurate, because annotations are not always revised before their publication, and sometimes include erroneous information, beside being incomplete by definition. In this scenario, computational methods that are able to quicken the curation process of such data are very important.
This has motivated the development of computational algorithms and softwares that utilize the available genomics information for gene function prediction, able to provide prioritized lists of biomolecular annotations to the biologists, in order to orientate their future research and experiments.

With this thesis, we first face the problem of predicting novel gene functions (or biomolecular annotations) through different computational machine learning methods, in which we take advantage of the properties of co-occurrences of annotations to suggest new likely gene functions. We propose some computational methods, implemented in an integrated framework, able to produce prioritized lists of predicted annotations, sorted on the basis of their likelihood. Particularly, we enhance an annotation prediction method already available in the literature, and then developed two variants of it, based on gene clustering and term-term similarity weight.

In addition, we also deal with the issue of the validation of the predicted annotations. Scientists keep adding new data and information to the annotation data banks as long as they discover new gene functions, and sometimes these data are erroneous or inaccurate. In addition, new discoveries are made every day, and the available information cannot be considered definitive. For these reasons, such databases are always incomplete. This leads to a significant problem of *validation*, because we do not have a true *gold standard* to refer. So, we designed and developed different validation procedures able to check the quality of our predictions. We introduce a validation phase consisting of a Receiver Operating Characteristic (ROC) analysis, a search for the predicted annotations into a new updated database version, and possibly an analysis of the available knowledge in the literature and through some available web tools.

To better understand the variation of the output predicted lists of annotations, we design and develop new measures, based on the Spearman coefficient and the Kendall distance. Such measures are able to state the correlation level between two lists by analyzing the difference between positions of the same element in two lists, and by evaluating the number of element couples having contrary order in the two lists. These measures demonstrated to be able to show important

patterns otherwise difficult to notice.
Finally, we provide a visualization and statistical tool able to state the novelty of the predicted gene annotations, denoted as *novelty indicator*. For each gene, this tool is able to depict the tree graph of the predicted ontological annotation terms, producing images easily understandable also by non-experts, and also a statistical value that states the level of *novelty* of the prediction.

Our tests and experiments confirmed the efficiency and the effectiveness of our algorithms, by retrieving manifold predicted annotations as confirmed in the updated database or in the literature. The similarity measures resulted very useful to understand the similarity of our predicted lists, making us able to see specific similarity patterns while key parameters vary.
The *novelty indicator*, possibly, resulted very useful in producing tree graphs able to make our lists of predicted biomolecular annotations clearly usable by biologists and scientists.
We believe that the tools presented within this thesis may be very useful to the bioinformatics and scientific community to address future research experiments about gene functions.

Sommario

I geni sono le unità essenziali e più importanti d'un organismo vivente, e la conoscenza delle loro funzioni è un punto centrale nella comprensione dei processi biologici fisiologici e patologici, nonché nello sviluppo di nuovi farmaci e terapie. Nella comunità biologica e scientifica, le assocazioni di questo tipo tra un gene ed una funzione sono state recentemente denominate *annotazioni biomolecolari*. Purtroppo, la scoperta di nuove annotazioni è spesso un'operazione lunga e costosa, perché lunghi e costosi sono gli esperimenti biologici *in vitro* eseguiti da medici e biologi.
I rapidi avanzamenti nelle tecnologie d'analisi dei dati del genoma hanno reso molte funzioni geniche disponibili online in pubbliche basi di dati e banche dati. Nonostante la loro innegabile importanza, queste sorgenti di dati non possono essere considerate né complete né del tutto accurate, perché le annotazioni non sono sempre riviste prima della pubblicazione, ed a volte includono informazioni errate, oltre ad essere incomplete per definizione. In questo scenario, metodi computazioni in grado d'accelerare il processo di mantenimento di queste banche dati sono davvero importanti.

Questo ha motivato lo sviluppo di algoritmi computazionali e software che utilizzano le informazioni genomiche disponibili per fare predizioni di funzioni geniche, in grado di fornire liste ordinate per priorità d'annotazioni biomolecolari ai biologi, per orientare la loro ricerca ed i loro futuri esperimenti.
Con questa tesi, prima affrontiamo il problema della predizione di nuove funzioni geniche (o *annotazioni biomolecolari*) attraverso diversi metodi d'apprendimento automatico (*machine learning*), nei quali sfruttiamo le proprietà statistiche delle co-occorrenze tra annotazioni per suggerire l'esistenza di nuove probabili funzioni geniche.
Proponiamo alcuni metodi computazionali, implementati in un software integrato, in grado di generare liste d'annotazioni predette, ordinate sulla base delle loro probabilità. In particolare, proponiamo un miglioramento d'un metodo di predizione d'annotazioni già presente in letteratura, e ne sviluppiamo poi due varianti, basate sul raggruppamento (*clustering*) di geni e sull'inserimemnto di pesi di similarità tra termini.

Inoltre, affrontiamo la questione cruciale della validazione delle annotazioni predette. Dato che gli scienziati, mentre scoprono nuove annotazioni, continuano ad aggiungere nuovi dati ed informazioni (a volte errati o non accurati) alle banche dati tutti i giorni, queste basi di dati risultano incomplete per definizione. Questo porta ad un significativo problema di validazione, perché chi predice computazionalmente nuove annotazioni non ha un vero *gold standard* a cui riferirci, con cui confrontare i propri risultati.
Perció abbiamo progettato e sviluppato diverse procedure di validazione in grado di controllare la qualità delle nostre predizioni: nella tesi introduciamo una fase di validazione che consiste nell'analisi delle curve *Receiver Operating Characteristic (ROC)*, nella ricerca delle annotazioni predette un una versione aggiornata della base di dati, ed infine nell'analisi della conoscenza disponibile in letteratura e tramite alcuni strumenti web.

Per comprendere meglio la variazione delle liste d'annotazioni predette gen-

erate in output, abbiamo progettato e sviluppato due nuove misure, basate sul coefficiente di Spearman e sulla distanza di Kendall. Questi coefficienti sono in grado d'esprimere il livello di similarità tra due liste analizzando il numero di coppie d'elementi che hanno diverso ordine nelle due liste. Le due misure introdotte hanno dimostrato d'essere in grado d'evidenziare interessanti andamenti e correlazioni altrimenti difficili da osservare.

Infine, forniamo un strumento statistico e di visualizzazione in grado d'esprimere il livello di *novità* delle predizioni d'annotazioni fatte per i geni, chiamato *novelty indicator*. Per ogni gene, questo strumento è in grado di disegnare un grafo ad albero dei termini predetti, producendo immagini facilmente comprensibili anche dai non-esperti, insieme ad un valore statistico che esprime il livello di novità della predizione.

I nostri test ed esperimenti hanno confermato l'efficienza e l'efficacia dei nostri algoritmi, ritrovando molte annotazioni predette come confermate nel database aggiornato o in letteratura. Le misure di correlazione sono risultate molto utili nel capire il livello di similarità tra le nostre liste di predizione, permettendoci d'osservare specifici andamenti di similairtà quando alcuni parametri-chiave variano.
L'indicatore di novità, infine, è risultato molto utile nel generare grafi ad albero ed immagini in grado di rendere le nostre liste d'annotazioni biomolecolari usabili facilmente da biologi e scienziati

Crediamo che gli strumenti presetati all'interno di questa tesi possano essere molto utili per la comunità bioinformatica e scientifica, per meglio indirizzare futuri esperimenti sulle funzioni geniche, e permettere così nuove scoperte o aprire nuove strade nella comprensione della biologia.

2 Introduction

The amount of biomedical and biomolecular information and data has grown in recent years, presenting new challenges and goals in the bioinformatics scientific community. Such large amounts of information makes computerized integration and analysis of these data unavoidable [1] [2], in particular controlled annotations of biomolecular entities (mainly genes and their protein products) are of great value to support scientists and researchers.
These data can effectively support the biomedical interpretation of biomolecular test results and the extraction of knowledge; which can then be used to formulate and validate hypotheses, and possibly discover new biomedical knowledge. Several computational tools, such as those for enrichment analysis [3] [4] [5] [6] [7], are available to analyze such annotations.
The information is often present in multiple resources, in different locations and formats, and within different technologies and electronic infrastructures; so, the integration of these data, in order to infer new knowledge, can be very difficult [20] [21].

To better understand the functions of genes and proteins, the biology community developed the novel concept of *annotation* that is an association of a gene (or gene product) with useful information. For example, the statement "the human gene ITGA6 is involved in the biological process named *anatomical structure formation*" represents a typical biomolecular annotation, made by the coupling <ITGA6, *anatomical structure formation*>.
This information is expressed through controlled vocabularies, sometimes structured as ontologies, where every controlled term of the vocabulary is associated with a unique alphanumeric code. Such a code associated with a gene or protein ID constitutes an annotation.
The annotation curator's task is paramount for the correct use of the annotations. Curators discover and/or validate new annotations; and publish them in online web databanks, thereby making them available to scientists and researchers.

However, available annotation databanks have many flaws. Since scientists discover new annotations every day, bio-databases are incomplete by definition. Furthermore, for recently studied genomes, they are mostly computationally derived. So only a few of them represent highly reliable human-curated information.
For example, since many curators manually add new annotations to the databanks after a literature review process, it is possible that they may accidentally lose some scientific papers that refer to a certain gene and/or a function. The consequence is having a lacking dataset of available annotations for that gene in the database.

To support and quicken the time-consuming curation process, prioritized lists of computationally predicted annotations are extremely useful. In the last decades, many computer scientists proposed algorithms and implemented softwares to face the task of prediction of biomolecular annotations from an informatics perspective. Our work described in this thesis befalls in this category.

Starting from a preliminary work developed in our research group [8], we propose some machine learning methods able to tackle the prediction of biomolecular annotations (Section 5).
One is the Singular Value Decomposition (SVD) [9], previously used by Draghici et al. [54], for which we propose some automated procedures for the choice of the best key parameters. Another is the Semantically Improved SVD (that we named *Semantically IMproved*, SIM1), where we take advantage of a gene clustering technique proposed by Drineas et al. in [10]. The third and last one is an evolution of the previous algorithms (that we named SIM2), enhanced with a term similarity computation, such as Resnik [14] [15], Lin [16], Jiang [17] and Relevance [18].

These methods are able to produce as output lists of ordered biomolecular annotations, that are *suggested* to the biologists as likely associations between genes and features. For example, in one of our tests we knew that the human gene MUC5AC is involved in the cellular component *fibril* (GO:0043205), among other seven, and our methods predicted that the gene is involved also in the cellular component *microfibril* (GO:0001517). This way, our methods *suggested* to the biologists' community the existence of the annotation <MUC5AC, *microfibril*>, not yet discovered.

Since we have to manage large amounts of data with efficient methods that have to use limited time, we also focus on the software architectures more suitable to respond our requisites. The software we extended was designed by a former member of our research group and described in [8], and developed with two interdependent components, a Java unit and a C++ unit, to take advantage of the different technological features provided by these programming languages (Section 6).

Another important task in the bio-annotation prediction is validation. Since we do not have a true *gold standard* to which we can compare our results, we developed different validation techniques, described in Section 7. We tested the efficiency and the efficacy of our methods in many ways. Among them, we counted how many predicted annotations can be found on an updated version of the database we use for the analysis. Results demonstrate the effectiveness of our methods, showing how the enhancements we introduced lead to better and more numerous predictions.

Together with the computational learning approaches, we also designed and implemented some methods to allow the automated choice of some key paramters, such as the SVD truncation level. We present the analysis of the tests made to verify the efficiency of these methods in Section 8.

To better understand the variation of the predicted annotation lists when varying the input key parameters, we introduce novel list similarity rates based on the origiinal Spearman's correlation coefficient [103] and on the Kendall distance [104]. These new rates show important correlations between the lists of predicted annotations (Section 10).

Finally, since the ultimate goal of bioinformatics is inferring biological knowl-

edge, we designed and implemented a software feature able to produce, for each gene, the Directed Acyclic Graph (DAG) of the biological feature terms predicted by our software. We also introduce a new *novelty* indicator to measure the gene annotation prediction we consider the most interesting and significant, from a biological point of view (Section 11).

Document structure The thesis is structured as follows. In this Introduction Section, we inserted a description of the controlled vocabularies and ontologies (Section 2.1) currently used in bioinformatics. We then describe the concept of *biomolecular annotation* and the problems of the annotation databanks (Section 2.2), followed by a depiction of the methods and algorithms used for the prediction of biomolecular annotations (Section 2.4), together with a literature review of the field (Section 2.5).
After the statements about the main motivations and goals of the thesis (Section 3), we describe the datasets used for our tests in Section 4.
In Section 5, we introduce the main component of the present thesis: the machine learning methods used for the prediction of biomolecular annotations. In Section 6 we discuss our choices about the software architecture, the programming languages, the libraries and the packages used.
In Section 7, we present another key part of the thesis. There we describe the multiple validation procedures we designed and implemented to verify the correctness of our predictions.
We describe the tests and the analysis made to test the algorithm key parameters in Section 8.
We then present the main validated results of our prediction methods, in Section 9, for different datasets in different modality.
In Section 10, we introduce our new variant of the Spearman's correlation coefficient and the Kendall tau distance, and the main results of the application of these measures to our annotation lists.
As mentioned before, we discuss also the biological importance and significance of our prediction by showing the graphs of the terms predicted for genes, together with a *novelty indicator* visualization and statistical tool, in Section 11. Within this section, we also provide a final list of predicted annotations, for which we give a biological interpretation, too.
Finally, in the Section 12 we state the general conclusions about the work, and sketch some possible future development direction. The reference Bibliography (Section 13) and the lists of tables, figures, acronyms, algorithms, gene symbols (Section 14) closes the thesis.

2.1 Controlled vocabularies and ontologies

Since the biological and bio-molecular knowledge increased a lot in the last fifty years, soon the scientific community started to develop new tools and instruments able to manage it.
A *controlled vocabulary* is a collection of words, phrases or names, which are used to tag units of information (document or work) so that they may be more easily retrieved by a search. They provide an efficient approach to organize the knowledge in an ordered way, where every term can be easily accessed by a computer or an electronic system. A controlled vocabulary is created according to

the rules of the curators and the owners, and (differently from the natural language) it does not contain homographs, synonyms, neither polysemies. These elements would enlarge the ambiguity of the controlled vocabulary.
Every element of the controlled vocabulary is usually identified with a unique specific ID code, and represents a particular concept or feature of the vocabulary domain.
For example, the Unified Medical Language System (UMLS) [22] is a compendium of many controlled vocabularies in the biomedical sciences, and can be considered also a comprehensive thesaurus of biomedical concepts. It consists of many knowledge bases and of many software tools to analyze them.

In genomics, the controlled vocabularies can be classified in two categories:
- *terminologies*, or *flat controlled vocabularies*, where the contained terms and concepts are not related through relationships;
- *ontologies*, or *structured controlled vocabularies*, where the contained terms are hierarchically organized. Often the hierarchy is a structured tree, where the most generic element is the root, and the most specific elements are the leaves, and every relationship is a precise semantic meaning.

Important flat controlled vocabularies are Online Mendelian Inheritance in Man (OMIM) [28] for genetic diseases, and Reactome [27] for metabolic pathway descriptions.
Since the semantic relationships between *-omics* terms has become very important in the last decades, the number of ontologies and elements of the ontologies has rapidly increased.

Meaningful structured controlled vocabularies - ontologies are, for example, Ontology for Biomedical Investigations (OBI) [29] about biological experiments and investigations details; Chemical Entities of Biological Interes (ChEBI) [30] made of molecular entities focused on chemical compounds; Systems Biology Ontology (SBO) [31] about terms commonly used in Systems Biology, and in particular in computational modeling; Medical Subject Headings (MeSH) [32] containing subject headings and descriptor terms of medical journals and books.

The most important and widespread biological ontology in bioinformatics is the Gene Ontology (GO) [23]. In the current thesis, we use the Gene Ontology terms and annotations as datasets to test our methods and to infer new knowledge about the analyzed genomes.

2.1.1 The Gene Ontology

The Gene Ontology is a bioinformatics initiative run by the Gene Ontology Consortium, a set of biological database curators, scientific communities and organizations whose aim is to record the features of genes and gene products among all the species [23]. The GO controlled vocabularies cover three areas, that are also three (almost) separate ontologies:

- *Cellular Component* (CC): the parts of a cell or its extracellular environment;
- *Molecular Function* (MF): the elemental activities of a gene product at

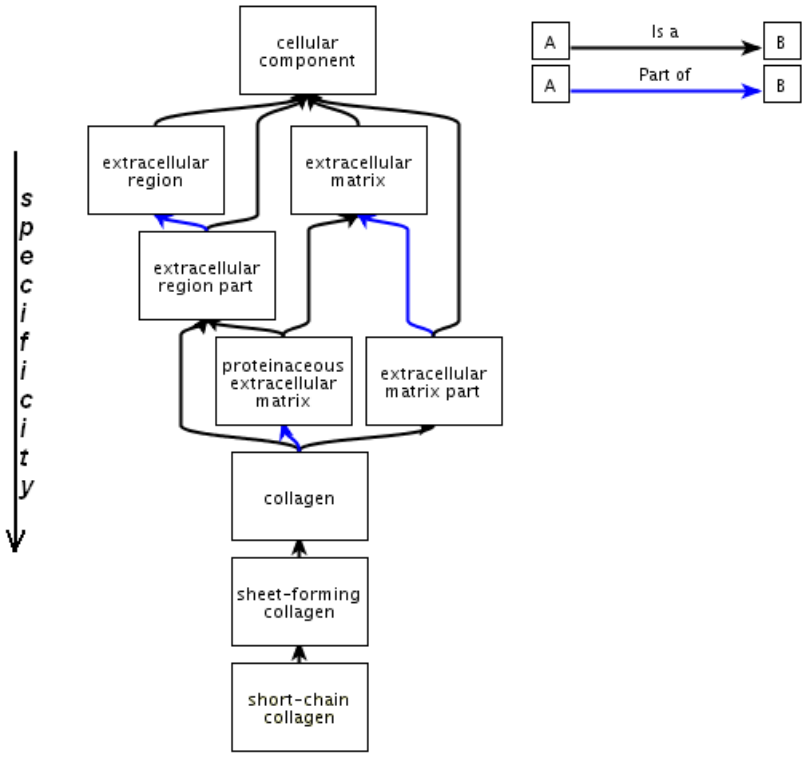

Figure 1: Part of the DAG of the GO ontology Cellular Component. The term Cellular Component is the root of the tree, and all the elements are connected throught the relationship "is_a" (black) or "part_of" (blue). The term "short-chain collagen" is the leaf, that is the most specific term in this tree portion. Image created with QuickGO web-tool [33].

the molecular level, such as binding or catalysis;

- *Biological Process* (BP): operations or sets of molecular events with a defined beginning and end, pertinent to the functioning of integrated living units: cells, tissues, organs, and organisms.

Every ontology is structured as a Directed Acyclic Graph (DAG), where every node is a term, and every edge is a relation of partition ("part_of") or association of inheritance ("is_a"). Every ontology/tree has a root element, that is also the name of the ontology (CC, MF, BP).

In September 2012, it contained around 36,500 terms used to describe gene (and gene product) features [26]. At that time, 23,907 terms were present for BP, 9,459 terms for MF, and 3,050 terms for CC.
Beyond biological and biomolecular terms, GO provides also a large dataset of annotations, that are associations between genes (or gene products) and biological feature terms. In September 2012, the Gene Ontology contained around 350,000 annotations [26].

2.2 Biomolecular annotations

In molecular biology, the annotation is a key concept. An annotation is an association (i.e. a couple) of a gene (or gene product) to a feature, that describes it. For example, the association of the gene VMA9 and the biological feature ID that represents the concept "is involved in the transmembrane transport" is a typical biomolecular annotation (see Fig. 2).

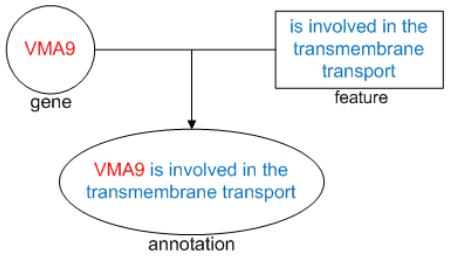

Figure 2: Depiction of the annotation associating the gene VMA9 to the biological feature "being involved in the transmembrane transport".

Annotations are very useful for the scientific community, because they allow biologists and physicians to annotate their discoveries about genes (and gene products) and their information features into large databanks, available online. An annotation recorded is the result of a discovery made through a validated biological experiment or a statistical test. Once scientists hypothesize a new annotation, curators validate it through biological experiments or through a literature research based on the published articles about that gene or gene product and that feature. When the new candidate annotation satisfies these tests, it is added to the set of the available confirmed annotations.
If a gene or gene product is annotated to a term, it is annotated also to the term ancestors present in the DAG structure.

Beyond the biomolecular entity fields (ID, name, origin data bank), the term fields (ID, name, origin data bank), the referenced publication, and the date of the annotation insertion, other useful fields are available for the Gene Ontology annotations [24]
An important field is "qualifier", a flag that indicates the explanation of the annotation. It can be one of:

- Not: indicates explicitly that no annotation between that biomolecular entity and that term exists. We consider these elements as *non-annotations*, because they state that curators discover that a certain gene or protein is never associated to a certain term.
- Colocalizes_with: used only in the Cellular Component annotations, indicates that the term may be associated to the gene product only temporarily during the life cell cycle.
- Contributes_to: used only for the Molecular Function annotations, indicates that gene product that is part of a complex and the annotated term describes the function of the complex.

- (Null): all the other cases.

Another important field is "evidence", that is an item of information on how that annotation was discovered and recorded in the database by the curators. The evidence codes can be:

- *experimental* evidence codes: best quality annotations, that come from the presence of scientific publications that support the description of the biomolecular entity (Inferred from Experiment (EXP), Inferred from Direct Assay (IDA), Inferred from Physical Interaction (IPI), Inferred from Mutant Phenotype (IMP), Inferred from Genetic Interaction (IGI), Inferred from Expression Pattern (IEP));

- *computational analysis* evidence codes: these annotations are based on an *in silico* analysis of the gene sequence and/or other data as described in the cited reference (Inferred from Sequence or structural Similarity (ISS), Inferred from Sequence Orthology (ISO), Inferred from Sequence (ISA), Inferred from Sequence Model (ISM), Inferred from Genomic Context (IGC), Inferred from Biological aspect of Ancestor (IBA), Inferred from Biological aspect of Descendant (IBD), Inferred from Key Residues (IKR), Inferred from Rapid Divergence (IRD), Inferred from Reviewed Computational Analysis (RCA));

- *curatorial statement* evidence codes: annotations made on the basis of statements made by the authors in the reference cited (Traceable Author Statement (TAS), Non-traceable Author Statement (NAS));

- *author statement* evidence codes: annotations inserted in the database on the basis of the curators' opinion or judgement, that do no fit into one of the other evidence classifications (Inferred by Curator (IC), No biological Data available (ND)). This last code ND indicates the situation when the curator did not find any information regarding the biomolecular annotation.

- *automatically-assigned* evidence codes: annotations that were not assigned by curators, but were computed from electronic automated methods or softwares (Inferred from Electronic Annotation (IEA)).

All the previously introduced annotation evidence codes are validated by human curators, except the last (IEA). An IEA annotation is added by scientists of the Gene Ontology Consortium that have retrieved it through an automatized electronic method, such as a BLAST search, or a Swiss-Prot keyword mapping, or others.
While the other evidence codes have little quality difference, these IEA annotations must be considered the less trustworthy, together with the ND annotations.

2.3 The problem

Despite their biological significance, there are many relevant problems that concern available genes (and gene products) annotations [34].
First, they are incomplete by definition: only a subset of biomolecular entities of the sequenced organism is known, and among those entities, only a small

part has been annotated by scientists and researchers so far. As long as scientists experiment and discover new findings, new annotations are inserted in the available databases, and some of the previously put annotations are corrected or enriched with more information.
Moreover, annotations are present in many different data banks and data warehouses available all over the world. Some data banks may provide different and sometimes conflicting information about particular annotations.
Annotation profiles of individual genes (or gene products) may be lacking, since biological knowledge about the functions associated with a certain gene (or gene product) might be yet to be uncovered.
Finally, another issue is that biological *in vitro* experiments are very expensive and time-consuming. For example, if a biologist has a metabolic gene which has a high homology to a known gene and he/she is only trying to verify that biological role, he/she is going to spend around 1,000 € for all the *in vitro* experiment phase (researchers' salaries excluded), and the all experiment will last around 3 weeks [35].

2.4 Prediction of biomolecular annotations

In this scenario, the development of computational tools able to analyze annotation data in order to appraise the relevance of inferred annotations, or generate a ranked list of new predicted annotations (e.g. to quicken the curation process) are an excellent contribution to the field.
Since many bioinformatics analyses currently performed on genomics and proteomics data rely on the available annotations of genes and gene products, the improvement of such annotations both in quantity, coverage and quality is paramount to obtain better results in these analyses.

The currently available annotation prediction (often denoted also as *prediction of gene functions*) approaches may be categorized in two classes:

- *experiment-based approaches*;
- *knowledge-based approaches.*

The first class consists of methods that, on the basis of specific experiments, suggest new features of the analyzed biomolecular entity. In the [79] paper, Laskowski et al. take advantage of the 3D structure of the protein to predict new functions of it. The paper authors create some protein 3D structure templates related to specific functions, and then compare the 3D structure of a new protein to these templates. On the basis of the similarity level, they decide if the new protein has that function or not.
Another *experiment-based* prediction is shown in the [80] paper, by Jones et al. The authors propose a new method based on the the similarity between sequence of DNA or protein, computed with BLAST (Basic Local Alignment Search Tool [81]).

On the contrary, the *knowledge-based* approaches apply computational methods to information and data already available in data banks, in order to predict new knowledge related to them.

These methods can be classified into two other sub-classes:

- *literature-based approaches*;
- *pattern-based approaches.*

The first sub-class include methods that exploit the large amount of scientific papers available in biology nowadays, to find relationship between documents related to different biomolecular entities. These approaches can use techniques of Indexing, Text Mining or Natural Language Processing (NLP).

Indexing methods (like [82] and [83]) aim at extracting keywords that rely to a particular biomolecular entity from the abstracts of scientific papers, and then at mapping these keywords to terms of controlled vocabularies.
In the [84] text mining paper, Chiang et al. developed a tool that automatically extracts and analyzes critical information from literature. In the [85] paper, Raychaudhuri et al. propose a method based on the statistical analysis of the occurrences of some keywords in the studied papers.

The last category, *pattern-based approaches*, are those that try to take advantage on data and data patterns already presents in corpus and databanks. The output produced by these methods is a list of annotations, absent from the analyzed database, sorted on the basis of their likelihood of existence.
The computational methods described and used in the present thesis (Section 5) befall in these sub-class, while a literature review is provided in the next Subsubsection 2.5.
Other methods of this category measure the similarity level between biomolecular entities on the basis of the similarity between the terms annotated to these entities, as made in a web application project [117] developed by our group. If two biomolecular annotations share many common annotations, we can consider them similar. The more annotations they share, the more their similarity level increases.

2.5 Related works

In the last decades, many computer scientists have used and developed different algorithms and software tools for the prediction of biomolecular annotations [36]. Due to the specific characteristics of the bioinformatics domain, a lot of researchers have been facing this issue with algorithms of the machine learning (ML) field.
King et al. [37] proposed the use of decision trees and Bayesian networks for predicting annotations by learning patterns from available annotation profiles. A Bayesian framework is used by Jiang and colleagues [39] that combines relational, hierarchical and structural information with improvement in data usage efficiency over similar methods. Also Troyanskaya et al. [40] developed a general framework that uses formal Bayesian reasoning to integrate heterogeneous types of high-throughput biological data, and applied it to Saccharomyces cerevisiae datasets.

More recently, Tao et al. [38] propounded to use a k-nearest neighbour (k-NN) classifier [41], whereby a gene inherits the annotations that are common among its nearest neighbour genes, determined according to the functional distance between genes, based on the semantic similarity of GO terms used to

annotate them.
Mi Huaiyu and colleagues [42] used another widespread machine learning algorithm Hidden Markov Model (HMM [43]) to build a tool named PANTHER able to model the evolution of gene functions. Also Deng et al. [44] took advantage of HMMs, because they are able to manage better the noisy sequential data such as gene expressions, they state.

Missing annotations can also be inferred by taking advantage of multiple data sources. In [46] expression levels obtained in microarray experiments are used to train a Support Vector Machine (SVM) classifier [45] for each annotation term, and consistency among predicted annotation terms is enforced by means of a Bayesian network mapped onto the GO structure.
SVMs are quite common in the area of prediction of gene functions. Minneci et al. [47] used them for the implementation of FFPred 2.0, an integrated feature-based function prediction server that includes support vector regression models for the prediction Gene Ontology (GO) terms. Also Mitsakakis and colleagues [48] applied the Support Vector Machines algorithm, a sigmoid fitting function and a stratified cross-validation approach to analyze a large microarray experiment dataset from Drosophila melanogaster in order to predict possible functions for previously un-annotated genes.
A kernel-based learning method has also used by Xin Li et al. in [50], where they propose a labeled graph kernel able to predict functions of individual genes based on the function distributions on their associated gene interaction networks.

Another important approach for predicting gene functions is biological network analysis. Considering the gene domain as a large network (where genes are nodes and a link between two genes is present if they share a common function) is the perspective of network analysis in genomics. The mainly diffused approach in this field is the *guilt by association* (GBA) [49], based on the idea that genes that share similar microRNA expression profiles can be clustered in similar functional classes [51].
This approach, anyway, shows some significant flaws. As explained by Pavlidis et al. [52], since a gene with many functions is more likely to have any function, "good" predictions tend to be generic predictions.
Warde-Farley and colleagues [53] developed a gene function prediction server, named GeneMANIA, where gene query datasets are represented as networks. Each gene network is assigned a weight primarily based on how well connected genes in the query list are to each other compared with their connectivity to non-query genes.

Although all very useful and powerful, all these methods shown a common flaw. They are not able to predict any annotations external to the current available analyzed corpus and knowledge. That is, they are not able to make any extrapolations.
As clearly explained in [54], for example, if we want to retrieve all the possible annotations between the gene SLC13A2 and GO Biological Process term *organic anion transport* (GO:0015711), and no gene is annotated to this term, the previously mentioned methods will report no association between them. This term is not part of the analyzed dataset existing elements, and so will not be considered by the methods.

To overcome this flaw, Khatri et al. [54] propounded a method able predict the possible existence of annotations absent from the analyzed datasets, by making some extrapolations.

By using basic linear algebra tools, Khatri et al. [54] proposed a prediction algorithm based on the Singular Value Decomposition (SVD) of the gene-to-term annotation matrix, which is implicitly based on the count of co-occurrences between pairs of terms in the available annotation dataset. Aftwerwards, they also used SVD enhanced with weighting schemes based on the gene and term frequency [55] [56].
Tagliasacchi et al. improved the SVD with an anomaly correction method based on a Bayesian network [57]

We departed from the SVD-based method described in [54] to develop novel methods that include automatic procedures for the choice of SVD key parameters, such as truncation level and likelihood threshold. We then developed an enhanced SVD variant based on gene clustering (SIM1), and another improved method based on gene clustering and on term-term similarity methods (SIM2).

Singular Value Decomposition (SVD) The first method we propose is a Single Vale Decomposition, already used in the prediction of gene functions in the past by Khatri et al. [54]. We provide an enhanced version of SVD, where the program automatically selects some key parameters, such as truncation level and likelihood threshold. We deeply describe SVD in Section 5.2.

Singular Value Decomposion is one of the most famous and widely used machine learning algorithms. Since its first appearance in 1873 and its final common definition [9], SVD has been shown to be a very powerful applied mathematics techniques. In the last decadeds, it has been applied to many fields, such as signal processing [58], natural language processing [59], bioinformatics [54] [55] [56] [60], recommender systems [61], robotics [62] and many more.

In these applications, the input matrix A is decomposed through SVD into the U, Σ, and V^Tmatrices. Then, these matrices are studied and processed to get semantic or statistical information about the input matrix A. In many cases, author choose to use the Truncated SVD algorithm: instead of the classical matrices, they select a truncation level $k \in \mathbb{N}$ and use the truncated matrices (U_k, Σ_k, V_k^T) to reconstruct the input matrix A.
Since the resultant reconstruction matrix is slightly different from the input one, it can be used for comparison and for finding correlations between input elements and output elements.
But how to best choose the SVD truncation level k? This question remains an open challenge, because this choice can strongly influence the resulting matrix $\widetilde{A}$, and greatly affects the quality of the obtained solution.
In the past, other scientists have dealt with this issue in various ways. In [63] paper, the authors used the approximation set coding (ASC) technique to compute the capacity of a hypothetical channel for a noisy dataset. By using the matrix A as an input noisy dataset, they took advantage of the resulting max-

imum appromiation set coding model that has the highest capacity to choose the best SVD truncation level.
Earlier, Vogel proposed another method in which he chose the best SVD truncation level as the one that leads to the best approximate solution of Fredhom first kind integral equation [64].

In the electronical engineering field, Safa Isam and colleagues used the Truncated SVD to overcome the ill conditioning of a Spectrally Efficient Frequency Division Multiplexing (SEFDM) system [65]. They chose the best SVD truncation level as the one that minimizes the competitive Bit Error Rate (BER) in system performances.
More recently, in the numerical analysis field, we recall this scientific paper [66] in which authors decided the best SVD truncation level through vector extrapolation methods, and the famous article of Per Hansen where the author chooses the optimal truncation index in order to satisfy the picard condition [67].
Lastly, in the bioinformatics field, the authors in bioinformatics [54] [55] [56] [60] used a heuristically fixed value for the SVD truncation level.
Although these methods are all very good and clever, none of them provides a general solution to the SVD best truncation choice problem that is suitable for all the domains. In this paper, we present a new algorithm based on a discrete optimization method of the Receiver Operating Characteristic (ROC) Areas Under the Curves (AUCs).

Continuous clustering problem We designed and developed a improved version of the SVD-based algorithm prediction, based on gene clustering. The gene clustering issue can be attribute to the *continuous clustering problem* (CCP) family.
Following the approach of Drineas and colleagues in [10], we first cluster genes and the weight the prediction on the basis of the gene cluster we evaluate. We explain the method in Section 5.3.
The main difference between *continuous clustering* (CCP) and the more common *discrete clustering* (DCP) is that in the former case an element can belong to many clusters (with different membership degrees), while in the latter case each element can belong only to zero or one cluster.
Drineas et al. [10] proved how a continuous clustering problem can be attribute to a discrete clustering problem, and show that the optimal value of DCP is an upper bound for the optimal value of the CCP.

In the past, other researchers have dealt with the discrete clustering problem. In [73], the authors presented a polynomial time approximation scheme for the discrete clustering, with an exponential dependence on k (where k is the number of the top right singular vectors of SVD) and the error parameter. Drineas and colleagues [10] also note that a closely related problem to DCP is the k-median problem, where again we seek to partition the given points into k clusters. For the k-median problem, Charikar et al. [74] gave a constant factor approximation algorithm based on a rounding procedure for a natural linear programming relaxation.

Term-term similarity We then designed and implemented another method that extends the previous one, where the clustering of the genes is influenced by the similarity between terms, SIM2 (described in Section 5.4).

Currently, different semantic similarity measures and techniques are available in the bioinformatics literature.
We can divide the methods available for the computation of similarity between terms into two main categories: *edge-counting approaches*, and *information-theoretic approaches*.

The *edge-counting approaches* are completely based on the structure of the analyzed ontology. To compute the similarity between two elements (nodes) of the same ontology (tree), these methods count the edges of the paths between them.
These measures do not change if the size of the ontology changes; however, they change if the ontology structure change, and in this case their values are not valid anymore.
The measure proposed by Wu and colleagues in [11] is part of these approaches. Authors propound a coefficient based on the average value between the edge distance of the *last common ancestor* (LCA) of the two compared terms. This measure has the flaw of supposing that all the nodes in the tree have the same distance, but anyway it is quite diffused due to its facility.
A similar measure is proposed by Rada and colleagues [13], that takes into account only the "is_a" relationships of the tree.
Lin and colleagues invented another measure in [12], where they introduce some concepts of "commonality", "description" and "similarity", based on statistics made on the edge counts between nodes. This measure seems quite efficient but also complex to be applied to Gene Ontology trees.

Differently from these methods, the *information-theoretic approaches* are based both on the structure of the ontology and on the characteristics of the analyzed dataset. For these reason, we choose these methods to compute the similarity in our term matrix into the SIM2 method.

To compute the term similarity between genes, we implemented many coefficients, mainly based on the maximum number of terms annotated to the *last common ancestor* (LCA) of the two analyzed terms. We implemented the method proposed by and named as Resnik [14] [15], Lin [16], Jiang [17], and *relevance* proposed Schlicker and colleagues [18].

Each of these measures show advantages and flaws. We mainly used Resnik coefficient, since Diaz-Diaz et al. [75] stated that it is more consistent than the others.

List correlation measures As mentioned, we also designed and implemented two annotation list correlation measures, described in Section 10.
Currently, there are many measures able to compute the similarity between lists made of any kind of element. Fagin and colleagues in [77] clearly review many of them and state that the most useful and consistent measures are Spearman's

rank correlation coefficient [103] and Kendall tau distance [104].
Another available interesting measure is the Goodman and Kruskal's γ. [78]. This metric measures the similarity level of the data orderings when ranked by each of the quantities. It takes the number of pairs of cases ranked in the same order on both variables (number of concordant pairs, N_c), and the number of pairs of cases ranked differently on the variables (number of discordant pairs, N_d), and the computes the statistic this way:
$G = (N_c - N_d)/(N_c + N_d)$
This measure may be useful but its meaning is quite similar to the Kendall distance, so we preferred this last coefficient that may better suit the case of multiple absences in the lists, as we need.
We departed from a recent work by Ciceri et al. [108] to develop new weighted and extended correlation coefficients, adapted to our annotation lists.

Novelty indicators To detect the "novelty" of a prediction, we implemented a statistical measure able to compare the gene term DAG tree before the prediction, and the term DAG tree of the same gene after the prediction.
Several measures are reported by Schlicker and colleagues in [105].
The authors started from two similar measures by Lord et al. [106] and Speer et al. [107] to develop a new measure, named $\text{GOscore}_{\text{BM}}$, that resulted more efficient than the previous.

An interesting visualization tool able to express the novelty inside a DAG tree is QuickGO [33]. Easy to use within its interface, it allows the user to depict a DAG tree of the ancestors of the term ID inserted, just through a simple web browser. An example image is the tree depicted in Fig. 1.
Anyway, its not possible to differentiate "old" existing terms from the "new" predicted terms with QuickGO. That is why we developed a new tool, DagViewer, described in Section 11.1.

3 Motivations and goals

As stated in the Introduction, biomolecular annotations are key elements in the modern biology, but their repositories still have many flaws.
Beyond data bank inaccuracy and incompleteness, the insertion of biologically validated new annotations is a very slow process.
The main goal of this thesis is to provide an effective software, based on efficient machine learning methods, to predict new reliable gene function associations, that can address the biological validation procedure and improve the quality of the available annotation databases.

The validation of the predicted annotations, however, leads to another issue: the lack of a real *gold standard* to compare with our predictions. We dealt with this problem by designing, implementing and describing multiple validation procedures, that can make our new predicted annotations more trustworthy.

We also wanted to understand the correlation between the predicted gene-function lists, when varying the key parameters of the computational learning methods. To do this, we designed and introduced new list correlation measures, whereby we were able to measure the similarity levels between different lists.

Lastly, in order to better understand the biological significance and novelty of our predictions, we introduced a *novelty indicator* measure, able to state which are the most relevant predictions.

Here we summarize the main goals of the present thesis:

- Improvement of already available computational learning methods for annotation prediction. We started from the available machine learning algorithms for the gene-function prediction, and focused on how to improve them. So, departing from the SVD implementation by Khatri and colleagues in [54], our first goal is to define possible improvements (such as automated procedures to select some algorithm key parameters), and later design and implement them.

- Design of new prediction methods. Another goal is to propose new methods suitable for the prediction of gene function association. We search for variants of the currently available methods, with strong mathematical foundations, that could be implemented and used without excessively increase the computational time and costs.

- Design of validation procedures. Since the annotation data banks are always incomplete, and so we do not have a complete *gold standard* to refer, we have to pay much attention to the validation aspects. We need to exploit all the possible ways to make our prediction the most possible trustworthy. Starting from the available ROC curve analysis validation, and we add new validation phases able to improve the quality of our predicted annotations.

- Design of comparison measures of annotation lists. Once we compute the predicted annotation lists, it is important to understand how different they

are, and how they vary, when the method key parameters vary. We try to answer the question: given two lists, how much *similar* are they? So first have to define the concept of *similarity* in our context, and we then design some new coefficients able to measure the similarity and correlation levels between lists.

- Visualization and novelty of the biological results. Beyond the novelty of our software within the computer science framework, another purpose of this thesis is to assess of how much novel, significant and innovative are the annotation predictions that we compute, in the biology field. We try to answer the question: how much *novel* is the prediction made on a certain gene term tree? Again, we first have to define the concept of *novelty* in our concept, and then design and implement a tool for understanding which annotation predictions are more novel than others. This tool can be useful also for the non-experts in biology.

- Implementation of all the previously defined components into an integrated software. The software has to be robust, agile and efficient, manage large amounts of data, with optimized time and resource costs. It has to allow easily possible future extensions and functionalities. In addition, it has to provide user-friendly interfaces able to allow also non-expert users to benefit from it, and produce results that can be interpreted by biologists and physicians.

We summarize the major results of this thesis in Section 12, among the conclusions.
From this dissertation, it emerges that important associations between genes and biological terms can be inferred through computational methods, and comes to envision a further and broader diffusion of computational learning algorithms like these in the bioinformatics research community.

4 Considered data

To run our tests and test our prediction method, we use datasets downloaded from Genomic and Proteomic Data Warehouse (GPDW) [86] [87] developed by our research team at Politecnico di Milano.

4.1 Genomic and Proteomic Data Warehouse (GPDW)

Given the large amount of biomolecular information and data available nowadays in different web sources, information integration has become a very important task for the bioinformatics community.
In order to integrate heterogeneous bioinformatics data available in a variety of formats from many different sources, our team at Politecnico di Milano decided to follow the data warehousing approach.
This is because, generally, accessing to data warehouse systems is quicker, and more efficient and effective than accessing the on demand systems. When used for singular and specific queries, on demand approaches could be preferable. In our case, we decided to divide our system into two parts: an integration part, where all the data sources are put together; and an analysis part, where all the data are used to infer new knowledge.

Our objective was to generate, maintain updated and extend a multi-species Genomic and Proteomic Data Warehouse (GPDW) that provided transparent provenance tracking, quality checking of all integrated data, comprehensive annotation based analysis support, identification of unexpected information patterns and which fostered biomedical discoveries. This data warehouse constitutes the back-end of a system that we will exploit through suitable web services, and will allow discovery of new knowledge by analyzing our large amount of information and data available.

Data integration The integration of data from different sources in our data warehouse is managed in three phases: *importing*, *aggregation*, and *integration*. In the former (*importing*), data from the different sources are imported into one repository. In the second (*aggregation*), they are gathered and normalized into a single representation in the instance-aggregation tier of our global data model. In the third (*integration*), data are organized into informative clusters in the concept-integration tier of the global model.

During the initial importing and aggregation phases, tables of the features described by the imported data are created and populated. Then, similarity and historical ID (*IDentification*) data are created by translating the IDs provided by the data sources to our internal OIDs (*Own IDentification*).
In doing so, relationship data are coupled with their related feature entries. According to the imported data sources and their mutual synchronization, relationship data may refer to feature entries or features that have not been imported into the data warehouse. In this case, missing integrated feature entries are synthesized and labeled as such (i.e., inferred through synthesis from relationship data). However, if a missing entry has an obsolete ID and through unfolded translated historical data it is possible to get a more current ID for it, the relationship is first moved to the latest ID and is marked as *inferred*

through historical data. In this way, all the relationships expressed by the imported relationship data are preserved and allows their subsequent use to infer new biomedical knowledge discoveries (e.g., by transitive closure inference, also involving the synthesized entries).
The final integration phase uses similarity analysis to test whether single feature instances from different sources represent the same feature concepts. In this case, they are associated with a new single concept OID. To keep track of the inference method used to derive an entry, an *inferrence* field is used in all tables that have been integrated. Furthermore, a summary quality rate for each concept instance is computed based on the source–specific instances contributing to the concept instance and its *inferred* attribute value.

At the end of the integration process, the defined indexes, unique, primary and foreign key integrity constraints of all the integrated tables are enforced in order to detect and resolve possible data inconsistencies and duplications, thus improving the speed of access to the integrated data, as well as their general quality.

Quantitative characteristics GPDW currently contains more than 1.84 billion data tuples, which amount to a total of about 383 GB of disk space (included index space). It integrates data of very numerous biomolecular entities and their interactions and annotations with many biomedicalmolecular features imported from several distributed databases, including Entrez Gene, UniProt, IntAct, MINT, Expasy Enzyme, BioCyc, KEGG, Reactome, GO, GOA and OMIM.
At the time of writing, it contains data for about 9,537,645 genes of 9,631 different organisms, 38,960,202 proteins of 338,004 species and a total of 71,737,812 gene annotations and 118,165,516 protein annotations regarding 12 biomedical controlled terminologies. The latter ones included, among others, 35,252 Gene Ontology terms and their 64,185,070 annotations, 28,889 biochemical pathways and 171,372 pathway annotations, and 10,212 human genetic disorders and their 27,705 gene annotations, together with 38,901 phenotypes (signs and symptoms). These last, which we extract from OMIM clinical synopsis semistructured descriptions, at the time of writing to our knowledge are not included in any other integrative database publicly available. Furthermore, the GPDW also integrates 542,873 very valuable interaction data between several different biomolecular entities, including 539,718 protein-protein interactions.

The GPDW constitutes the backend of a Genomic and Proteomic Knowledge Base (GPKB) publicly available at `http://www.bioinformatics.dei.polimi.it/GPKB/` An easy-to-use Web interface enables the scientific community to access and comprehensively query all the data integrated in the GPDW and to take full advantage of them. The system can support also complex multi-topic queries, that cannot be performed in other available systems.

Database table structure For our tests involving biomolecular annotations between genes and GO terms, we mainly used four tables in the GPDW:

- The gene table, *gene*
- The feature term table, *biological_function_feature*
- The table containing the ancestor terms of the biological function terms, *biological_function_feature_unfolded*
- The table of the annotations between genes and biological function feature, *gene2biological_function_feature*

The annotation table *gene2biological_function_feature* content is made mainly by gene functions coming from Entrez Gene [86].
We depicted the logic scheme of these tables in Fig. 3.

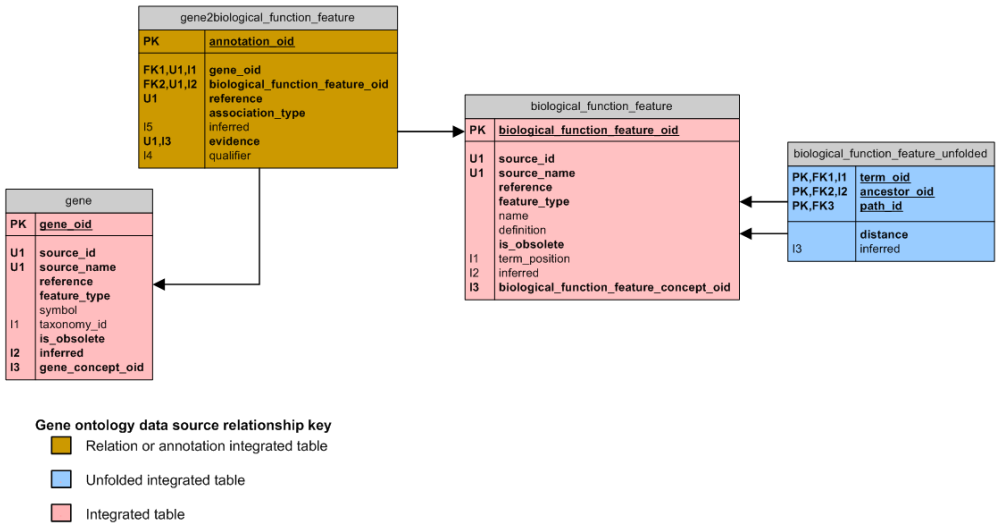

Figure 3: Logic scheme of the database GPDW tables used by our software.

As reported clearly in [86], in the (imported and integrated) tables, the *feature_type* field is set manually according to our defined feature ontology.
Some fields with imported textual data that have a limited range of values and are used as labels for data in other table fields are not stored as provided by the data source file, but are encoded to numeric values in order to improve performance of accessing data in the data warehouse.

These fields are:

- the *source_name* field (among others), whose codified values are specified in the *source_metadata* metadata table
- the *feature_type* fields, whose codified values are specified in the feature metadata table

- the *inferred* fields; they can assume the following values, whose coding is specified in the *inferred_flags* flag table:
- in the *gene*, and *biological_function_feature* tables it can assume the value: SYNTHESIZED
- in the *biological_function_feature_unfolded* table it can assume the values: SYMMETRY, TRANSITIVE_CLOSURE or UNFOLDED. The SYMMETRY and TRANSITIVE_CLOSURE values are copied from the *data_source_relationship* table, whereas the UNFOLDED value is set for all the entries
- in the *gene2biological_function_feature* table it can assume the values: TRANSITIVE_CLOSURE or HISTORY_REPLACED
- the *evidence* and *qualifier* fields placed on the *gene2biological_function_feature* table have codified values that are specified in the evidence_flags and qualifier_flags flag tables

The *is_obsolete* field placed on the *biological_function_feature* or is a boolean field, and can take these values: TRUE or FALSE. It is set as FALSE on default, and it is set as TRUE only for the IDs that belong to the Discontinued IDs of the history tables.
The *gene2biological_function_feature* table contains the *evidence* attribute in the table primary key because different evidences can be associated with a gene-go annotation.
The *name* and *definition* fields in the *biological_function_feature* table, as well as the *taxonomy_id* and *symbol* fields in the *gene* table and the *term_position* in the *biological_function_feature* table, are not mandatory because some synthesized entries without these attribute values could exist.
The *gene_concept_oid*, *biological_function_feature_concept_oid*, fields are set as mandatory, but their values is calculated and set after the population of all instances of their tables.

4.2 Datasets

For our tests, we used some datasets availabe on the GPDW. We focused on the Gene Ontology (GO) [23] annotations of *Bos taurus* (cattle), *Gallus gallus* (red junglefowl), *Danio rerio* (zebrafish), and *Homo sapiens* genes. We chose these organisms because their gene annotations to the three GO ontologies (Biological Process, Molecular Function, Cellular Component) include a representative sample of annotation numbers, involved genes and terms.

Table 1 provides a quantitative description of the annotations considered, in the July 2009 database version that we use often for our analysis, and in the March 2013 database version we use mainly for seeking confirmation of our predictions.

Table 1: Quantitative characteristics of the considered annotation datasets in the analyzed July 2009 database version and in the updated March 2013 database version from GPDW [86]. Numbers do not include IEA and ND annotations, obsolete terms, obsolete genes. #gs is the number of genes; #fs is the number of biological function features; #as is the number of annotations; Δ is the difference of annotation amounts between the two database versions, and Δ% is the percentage difference.

		July 2009			March 2013			#as comparison	
Organism	GO	#gs	#fs	#as	#gs	#fs	#as	Δ	Δ%
Homo sapiens	CC	7,868	684	14,341	12,033	1,021	31,135	16,794	117.10
	MF	8,590	2,057	15,467	10,460	2,703	25,396	9,929	64.19
	BP	7,902	3,528	21,048	11,681	7,295	64,212	43,164	205.07
Bos taurus	CC	497	234	921	1,689	543	3,437	2,516	273.18
	MF	543	422	934	1,449	984	2,830	1,896	203.00
	BP	512	930	1,557	1,698	2,986	7,075	5,518	354.40
Danio rerio	CC	430	137	607	5,395	388	6,395	5,788	953.54
	MF	700	309	934	4,718	777	5,880	4,946	529.55
	BP	1,528	960	4,106	6,491	2,541	14,217	10,111	246.25
Gallus gallus	CC	260	148	478	777	346	1,571	1,093	228.66
	MF	309	225	509	698	551	1,416	907	178.19
	BP	275	527	738	903	1,738	3,932	3,194	432.79

By observing Table 1, we can notice that the larger dataset is the Homo sapiens one. For the three sub-ontologies together, Homo sapiens genes have 50,856 annotations in the July 2009 database version, and 120,743 annotations in the updated March 2013 database version. Obviously, this is due because of the main effort given by physicians and biologists to study human gene functions.

5 Learning methods

In this section, we describe the work flow of the algorithm we developed for the gene function prediction.
First, we describe the data pre-processing operations we apply to the considered datasets (Subsection 5.1). We then depict the machine learning methods we designed and implemented: the classical SVD in Subsection 5.2; a variant of SVD improved with gene clustering, SIM1 in Subsection 5.3: and a variant of SIM1 enhanced with similarity between terms, SIM2 in Subsection 5.4.
Finally, we describe the post-processing phase we apply to the output prediction datasets (Subsection 5.5), and the comparison we make between the input matrix and the reconstructed output matrix (Subsection 5.5).

5.1 Data pre-processing

The first phase of the prediction flow is, obviously, the data reading. Given an organism taxonomy ID and a set of ontologies, our program collects all the tuples contained in the GPDW database table gene2biological_function_feature. We do not collect all the tuples:
- since we consider annotations associated with terms having evidence code IEA or ND (Subsection 2.2) not enough trustworthy, we exclude them;
- we exclude the annotations involving obsolete genes;
- we exclude the annotations involving obsolete biological function features.

In the following SQL statement we report the an initial query example for the annotations of Homo sapiens (ID: 9606) genes and features of cellular component, molecular function and biological process.

```
SELECT DISTINCT (g2bff.biological_function_feature_oid,
  g2bff.gene_oid), bff.feature_name, g2bff.qualifier
FROM gene2biological_function_feature AS g2bff
INNER JOIN biological_function_feature AS bff
 ON g2bff.biological_function_feature_oid =
  bff.biological_function_feature_oid
INNER JOIN metadata.feature
 AS mf ON bff.feature_type = mf.feature_id
INNER JOIN gene AS g ON g2bff.gene_oid = g.gene_oid
LEFT JOIN flag.evidence_flags
 AS ef ON g2bff.evidence = ef.id
LEFT JOIN flag.qualifier_flags AS qf
 ON g2bff.qualifier = qf.id
LEFT JOIN flag.inferred_flags AS fif
 ON fif.id = g2bff.inferred
WHERE g.taxonomy_id IN ( 9606 )
AND mf.feature_name IN ( 'cellular_component',
  'molecular_function', 'biological_process' )
AND ef.name NOT IN ('IEA','ND')
AND (bff.is_obsolete NOT IN ('t') OR bff.is_obsolete IS NULL)
AND (g.is_obsolete NOT IN ('t') OR g.is_obsolete IS NULL)
```

Where *gene* is the gene table, *biological_function_feature* is the table of the GO terms, and *gene2biological_function_feature* contains the annotations between genes and terms.
Note that in the previous SQL query the last three lines are related to the elements to exclude from the computation: all the annotations having $evidence = IEA$ or $evidence = ND$, and all the annotations made by obsolete genes or obsolete biological function feature.

Annotation unfolding and pruning Curators of ontological annotations always use the most specific ontology terms to describe a given feature of an annotated gene or gene product. Such annotations, named direct annotations, are the only ones available in biomolecular databases.
Yet, according to the ontology structure, when a gene is annotated to an ontological term, it is implicitly annotated also to all the ancestor terms of the directly annotated term. Such annotations to the ancestor terms, which describe more generic features, are named indirect annotations (as explained in Subsection 2.1.1).

In order to retrieve all direct and indirect annotations, when we elaborate ontological annotations, we unfold them according to the ontology structure, and retrieve the related indirect annotations. We refer to this process as *annotation unfolding* [88].

For example, if we find the annotation <RAB26, cell part> made of the GO term *cell part* (GO:0044464) associated with the gene RAB26, we find out that this term is a descendant of *cell* (GO:005623) and obviously *cellular component* (GO:0005575). We then add to our datasets the following two annotations: <RAB26, cell> and <RAB26, cellular component>, if they are not already present in our dataset.

We show an illustration of this example in Fig. 4

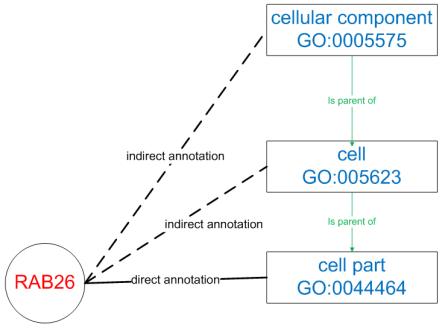

Figure 4: Depiction of the unfolding phase of the annotation <RAB26, cell part>. The program finds the indirect annotations <RAB26, cell> and <RAB26, cellular component> and adds them to the analyzed dataset.

The program executes this unfolding phase by executing a query on the database *biological_function_feature_unfolded* table where it retrieves all the ancestor of the terms found with the previously query in Section 5.1.

5.2 Singular Value Decomposition (SVD)

The first method we design and implement for our prediction tool is Singular Value Decomposition (SVD), already used in the past for this scope by Khatri et al. [54]. The enhanced methods SIM1 (Section 5.3) and SIM2 (Section 5.3) are alternatives to this algorithm.

Let $A_d \in \{0,1\}^{m \times n}$ define the matrix representing all direct annotations of a specific Gene Ontology (GO) [23] for a given organism. The m rows of A_d correspond to genes, while the n columns correspond to GO features. The entries of A_d assume values from the binary alphabet $\{0,1\}$ according to the following rule:

$$A_d(i,j) = \begin{cases} 1, & \text{If gene } i \text{ is annotated to feature } j. \\ 0, & \text{Otherwise} \end{cases} \tag{1}$$

Annotation curators are asked to always use the most specific GO feature for a given functional category. Thus, when a gene is annotated to a feature, it is implicitly assumed to be indirectly annotated also to the more generic features for that category, i.e. all the feauture ancestors in the GO Directed Acyclic Graph (DAG). As such, let A denote a modified gene-to-feature matrix, where the assignment of its entries is given by:

$$A(i,j) = \begin{cases} 1, & \text{If gene } i \text{ is annotated to feature } j \\ & \text{or to any descendant of } j. \\ 0, & \text{Otherwise.} \end{cases} \tag{2}$$

The i_{th} row of the A matrix (a_i^T) contains all the direct and indirect annotations of gene i. Conversely, the j_{th} column encodes the list of genes that have been annotated (directly or indirectly) to feature j. This process is sometimes defined as *annotation unfolding* [88] (described in Section 5.1).
At this point, the annotation prediction can be performed by computing the SVD of the matrix A, which is given by:

$$A = U \, \Sigma \, V^T \tag{3}$$

where U is a $m \times m$ unitary matrix (i.e. $U^T \, U = I$), Σ is a non-negative diagonal matrix of size $m \times n$, and V^T is a $n \times n$ unitary matrix (i.e. $V^T \, V = I$). Conventionally, the entries along the diagonal of Σ (namely singular values) are sorted in non-increasing order. The number $r \leq p$ of non-zero singular values is equal to the rank of the matrix A, where $p = \min(m,n)$

Truncated SVD If we performed the classical SVD product as described in (3), we would get the same A input matrix, and we would not be able to compute a prediction.
By considering only the most significant rows and columns of U and V, the Truncated SVD method is able to produce a good reduced rank approximation matrix of it ($\widetilde{A}$), that contains meaningful information about semantic relationships between the annotations [54]. Therefore, we then use this output matrix for a comparison to the input matrix.

For any positive integer $k < r$, it is possible to create a matrix $\widetilde{A}$, with:

$$\widetilde{A} = U_k \, \Sigma_k \, V_k^T \tag{4}$$

where U_k (V_k^T) is a $m \times k$ ($n \times k$) matrix achieved by retaining the first k columns of U (V) and Σ is a $k \times k$ diagonal matrix with the k largest singular values along the diagonal. The matrix $\widetilde{A}$ is the optimal rank-k approximation of A, i.e. the one that minimizes the norm (either the spectral norm or the Frobenius norm) $\|A - \widetilde{A}\|$ subject to the rank constraint.
In [54], the authors argued that the study of the matrix $\widetilde{A}$ shows the semantic relationships of the gene-function associations. A large value of $\widetilde{a}_{ij}$ suggests that gene i should be annotated to term j, whereas a value close to zero suggests the opposite. As a matter of fact, the SVD of the matrix A is equivalent to the method of Latent Semantic Indexing (LSI) [72], an information retrieval algorithm where the input is a matrix that contains the occurrences of words in indexed documents.
In order to better comprehend why $\widetilde{A}$ can be used to predict gene-to-term annotations, we highlight that an alternative expression of (4) can be acquired by basic linear algebra manipulations:

$$\widetilde{A} = A \, V_k \, V_k^T \tag{5}$$

Additionally, the SVD of the matrix A is related to the eigen-decomposition of the symmetric matrices $T = A^T A$ and $G = AA^T$. The columns of V_k (U_k) are a set of k eigenvectors corresponding to the k largest eigenvalues of the matrix T (G). The matrix T has a simple interpretation in our context. In fact,

$$T(j_1, j_2) = \sum_{i=1}^{m} A_{(i,j_1)} \cdot A_{(i,j_2)} \tag{6}$$

i.e. $T(j_1, j_2)$ is the number of times that terms j_1 and j_2 are used to annotate the same gene in the existing annotation profile. Consequently, $T(j_1, j_2)$ indicates the (unnormalized) correlation between term pairs and it can be interpreted as a similarity score of the terms j_1 and j_2, the computation of which is exclusively based on the use of these terms in available annotations. The eigenvectors of T (i.e. the columns of V_k) can be considered as a reduced set of eigen-terms. Intuitively, if two terms co-occur frequently, they are likely to be mapped to the same eigen-term. Based on (5), the i_{th} row of $\widetilde{A}$ can be written as

$$\widetilde{a}_i^T = a_i^T \; V_k \; V_k^T \tag{7}$$

Thus, the original annotation profile is first transformed in the eigen-term domain, while retaining only the first k eigen-terms by the multiplication with V_k, and then mapped back to the original domain by means of V_k^T. This corresponds to projecting the original vector a_i^T onto the k-dimensional subspace spanned by the columns of V_k.

The decomposition of the matrices and the difference between SVD and Truncated SVD are represented in Fig. 5.
The output matrix $\widetilde{A}$ reconstruction can be strongly influenced by the choice

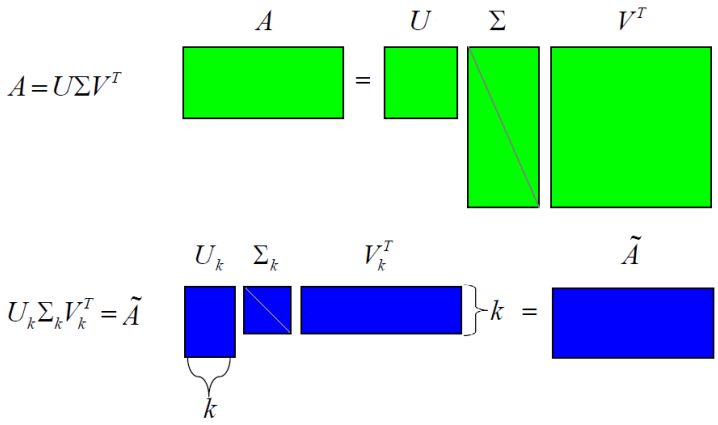

Figure 5: An illustration of the Singular Value Decomposition (upper green image) and the Truncated SVD reconstruction (lower blue image) of the A matrix. In the classical SVD decompostion, $A \in \{0,1\}^{m \times n}$, $U \in \mathbb{R}^{m \times m}$, $\Sigma \in \mathbb{R}^{m \times n}$ $V^T \in \mathbb{R}^{n \times n}$. In the Truncated decomposition, where $k \in \mathbb{N}$ is the truncation level, $U_k \in \mathbb{R}^{m \times k}$, $\Sigma_k \in \mathbb{R}^{k \times k}$, $V_k^T \in \mathbb{R}^{k \times n}$, and the output matrix $\widetilde{A} \in \mathbb{R}^{m \times n}$

of the truncation level k. We will discuss our algorithm for selecting the best truncation parameter in Section 5.6.

5.3 Semantically improved SVD with gene clustering (SIM1)

The first method we projected and propose is an semantically improved variant of the SVD, based on a gene clustering technique introduced by Drineas et al. in [10]. We denote this method as *SIM1*.

The SVD method implicitly adopts a global term-to-term correlation matrix ($T = A^T A$), estimated from the whole set of available annotations. In contrast, we propose an adaptive approach, called the SIM method, which clusters genes (the rows of matrix A) based on their original annotation profile and values a set of distinct correlation matrices, T_c, one for each cluster.
For each matrix, T_c, a predicted annotation profile for gene i is computed. The selected predicted annotation profile for gene i is the one that minimizes variation, measured by the Euclidean norm, with respect to the original annotation profile of the gene.

We choose a number C of clusters, and completely discard the columns of matrix U where $j = C+1, \ldots, n$. The best choice of this parameter is a key point of the algorithm: we describe it in Section 5.6).
Each column, u_c, of SVD matrix U represents a cluster, and the value $U(i, c)$ indicates the membership of gene i to the c^{th} cluster [10]. We use this membership degree to cluster the genes (the rows of matrix A). As such, each gene might belong to more than one cluster with different degrees of membership. We notice that the columns of U are a set of eigenvectors for the matrix $G = AA^T$, i.e. the similarity between gene pairs is measured by the inner product of their annotation profiles.

For every c cluster $c = 1...C$, we compute T_c.
To calculate T_c, first we generate a modified gene-to-term matrix $A_c = W_c A$ (where $W_c \in \mathbb{R}^{m \times m}$ is a diagonal matrix with the entries of u_c along the main diagonal), in which the i^{th} row of A is weighted by the membership score of the corresponding gene to the c^{th} cluster.
Then, we compute $T_c = A_c^T A_c$

At this point, we apply the Truncated SVD to T_c, tied to the current cluster number c and to the truncation level k:

$$SVD(T_{c,k}) \;=\; U_{c,k} \;\; \Sigma_{c,k} \;\; (V_{c,k})^T$$

Notice that the V_k columns are the k bigger eigenvectors of T_c

$$T_{c,k} \;=\; T_c \;\; V_{c,k} \;\; (V_{c,k})^T$$

Once we made the SVD decomposition for each T_c, we compute all the annotation profiles-rows.
For each row i, for each cluster number c:

$$\vec{a}_{i,c,k} \;= a_i \;\; V_{i,c,k} \;\; (V_{i,c,k})^T$$

We choose the final annotation profile-row by minimizing the variation of that profile from the original input one:
For each row i, for each cluster number c:

$$c_i^{final} \;=\; arg\ min\ |\vec{a}_{i,c,k} - a_i|$$

The vector $\vec{a}_{i,c,k}$ represents the annotation profile corresponding to the i^{th} row of the output matrix, with SVD truncation level k, for the cluster number c.
Finally, we set the definitive profile-row:

For each row i:
$a_{i,k} \;=\; a_{i,\, c_i^{final},\, k}$

We describe an overview of the SIM1 algorithm in Fig. 6.
The general idea of the just described SIM1 algorithm is that, if similar genes are grouped in clusters, we can assign common weights to all the genes grouped into a single cluster. These weights (expressed through the T_c matrix) then influence the value of each entry of the analyzed matrix before the decomposition is performed, and subsequently influence the prediction, too.
For example, a gene related to very few terms (i.e. few 1 entries surrounded by all 0's in some matrix zone) in the classical SVD might just be discarded. Instead, in SIM1 it may be grouped into a similar gene cluster that gives it a high weight, and so it may be present also in the output matrix. In SIM1, annotations (i.e. matrix entries) are not only important on the basis of their positions and their neighbors in the input matrix, but also because of the cluster their genes are included. We assume that this modification of the input matrix based on the gene clustering may lead to more consistent predictions.
We express this statement through the use of a new T matrix. In the classical

Given the annotation input matrix A of clusters:

1. Run the algorithm for the best truncation choice, the algorithm for the best τ threshold choice, and the algorithm for the best cluster number choice (if these key parameters are not available yet)
2. Consider only the columns U_j of matrix U, where $j \in \{1, ..., C\}$ and $A = U\Sigma V^T$
3. (SIM1) Compute the matrix $G = AA^T$
   (SIM2) Compute the term-term similarity matrix, $S \in \mathbb{R}^{n \times n}$
   (SIM2) Compute the matrix $G = ASA^T$
4. For each cluster c
   (a) Use the membership degree $U(i, c)$ of gene i to cluster c, to group the genes: $u_c = c^{th}$ largest eigenvector of G
   (b) Generate $A_c = W_c A$, where $W_c = diag(u_c)$ and $W_c \in \mathbb{R}^{m \times m}$
   (c) Compute the correlation matrix, $T_c = A_c^T A_c$
   (d) Compute the SVD of the correlation matrix: $SVD(T_{c,k}) = U_{c,k}\ \Sigma_{c,k}\ (V_{c,k})^T$
5. For each annotation profile-row i, and for each cluster c:
   Compute the annotation profile-row for every possible cluster number: $\vec{a}_{i,c,k}$
6. For each cluster c, and for each annotation profile-row i:
   Choose the best cluster number: c_i^{final}
7. Collect the final annotation profiles-rows $\vec{a}_i$ to create the output matrix $\widetilde{A}$
8. Compare A elements with $\widetilde{A}$ elements

Figure 6: Overview of the SIM1 and SIM2 algorithms.

Truncated SVD is simply $T = A^T A$, while in SIM1 we use the T_c matrix, based on the clustering operation just described.

5.4 Semantically improved SVD with gene clustering and term-term similarity weights (SIM2)

The gene clustering introduced in the SIM1 method consider the similarity matrix $G = A^T A$. This form suffers from the disadvantage of making all the terms weight the same.
For this reason, we include a similarity measure into the matrix G, with a variant of the SIM1 method that we denoted SIM2.

In SIM2, the G matrix is made by $G = A^T S A$, where S is a term-term similarity matrix built with one of the main term semantic similarity methods.

Starting from a pair of ontology terms, j_1 and j_2, the term functional similarity $S(j_1, j_2)$ can be calculated using different methods, such as Resnik [14], Lin [16], Jang [17], and *relevance* [18].

As already said, we implemented many of the term semantic similarity measures currently available in the scientific literature. Since the considered terms for our annotation datasets are part of the Gene Ontology [23], that is a structured controlled vocabulary, we can take advantage of the ontology structure. All the methods we choose, part of the *information- theoretic approaches*, are based on the notion of *last common ancestor* (LCA) between the two analyzed terms t_1 and t_2: the LCA is the closest ancestor node that t_1 and t_2 share.

We denote with $P(t)$ the probability of the term t among all the terms of the vocabulary:

$$P(t) = \frac{\#\, terms\, annotated\, to\, t}{total\, number\, of\, terms\, in\, the\, vocabulary} \tag{8}$$

If the term t is annotated to all the terms in the vocabulary, $P(t) = 1$, for example if t is the tree root. If t is annotated to no terms in the vocabulary, $P(t) = 0$

We introduce the concept of *information content* of the *corpus* as:

$$IC_{corpus}(t) = -\, log_{10}\, P(t) \tag{9}$$

The subscript "corpus" indicates that the IC value is tied to the corpus-vocabulary used.
If t is annotated to all the terms in the vocabulary, then $P(t) = 1$, and then $IC_{corpus}(t) = 0$.
If t is a child node of t_p, then $IC_{corpus}(t) \leq IC_{corpus}(t_p)$

The first measure we introduce is the **Resnik** [14], where the value corresponds to the maximum information content between the LCAs of the analyzed terms:

$$ResnikSimilarity(t_1, t_2) = \max_{t \in LCA(t_1, t_2)} IC_{corpus}(t) \tag{10}$$

This value can range in the interval $[0, \infty)$. This measure shows a low similarity between terms near to the root node, and it is considered by many scientists as the best term semantic similarity measure available [75] [76].

Another interesting metric we analyzed and implemented is **Lin** [16]:

$$LinSimilarity(t_1, t_2) = \frac{2 \cdot \max_{t \in LCA(t_1, t_2)} IC_{corpus}(t)}{IC_{corpus}(t_1) + IC_{corpus}(t_2)} \tag{11}$$

This measure show the advantage of providing values in the interval $[0, 1]$, but does not provide the advantage of low similarity values between terms near to the root node (differently from Resnik measure). It can also be seen as a normalized version of the Resnik measure (equation 10).
Lin's values also increase in relation to the degree of similarity shown by two

terms, and decreases with their difference.

The semantic similarity measure introduced by **Jiang** [17] consider both the information content and the ontology structure:

$$JiangSimilarity(t_1, t_2) =$$

$$= \frac{1}{1 + IC_{corpus}(t_1) + IC_{corpus}(t_2) - 2 \cdot \max_{t \in LCA(t_1,t_2)} IC_{corpus}(t)} \tag{12}$$

The values of this measure range in the $(0, 1]$ interval, and suffers the problem of the high similarity score between terms near the root node.

The last measure we studied and implemented is **relevance**, proposed by Schlicker et al. [18]:

$$RelevanceSimilarity(t_1, t_2) =$$

$$= \max_{t \in LCA(t_1,t_2)} \frac{2 \cdot IC_{corpus}(t)}{IC_{corpus}(t_1) + IC_{corpus}(t_2)} \cdot (1 - P(t)) \tag{13}$$

The method weights the Lin similarity measure with a weight $(1 - P(t))$, that is the likelihood of the LCA term. This way, if t_1 and t_2 results near to the root node, the weight decreases their similarity score.

We describe an overview of the SIM2 algorithm in Fig. 6.

5.5 Data post-processing

Once the software reconstructs the output matrix, the predicted annotations can show an inconsistency problem: in the output matrix, it is possible that a gene is annotated to a certain term, but not to its ancestor. So, we designed and implemented a post-processing solution phase, denoted *anomaly correction*. Finally, at the end of the prediction process, we divide the annotations in four categories, corresponding to indicators similar to TP, TN, FP, FN (paragraph *input and output matrix comparison*).

Anomaly correction For the previously described learning methods (in Subsections 5.2, 5.3, 5.4), annotation profiles of some single genes could be inconsistent with the hierarchical structure of the ontology, that is a gene results annotated to a particular term, but not annotated to an ancestor term, contrasting the *true path rule*.
This Gene Ontology rule states that if a gene is annotated to a term, then all the ancestor terms, reachable from it through ontology hierarchical structure, must be present in the annotations profile of the gene (as clearly described by Tanoue et al. in [25]).

Many methods can be used for the anomaly correction, depending on hierarchical structure, from a simple correction from leaves to root, or viceversa, to a use of more complex structures as Bayesian networks [57].

In our software, we implemented the first solution, by updating terms depending on their parents-terms or children-terms.

The method we used (denoted as *FromLeaves*) consists in updating parent-terms with the greater values between its children-terms. An iterative process is carried on from the leaf nodes to the root node, in order to correct possible values, that could create anomalies, for some threshold values. Such method correspond first to compute a threshold, and afterwards to correct all the present anomalies, by recomposing all the paths that satisfy the *true path rule*.
We show an illustration of the *FromLeaves* approach in Fig. 7 and in Fig. 8.

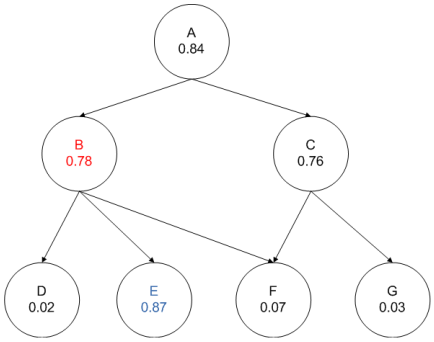

Figure 7: Output annotation profiles before the anomaly correction

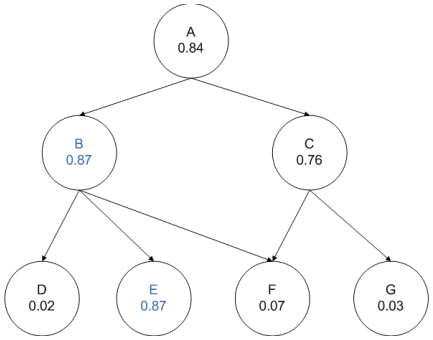

Figure 8: Output annotation profiles after the anomaly correction

Ten-fold cross validation For the production of the ROC curves, to avoid the model overfitting, and remove all the possible adjacency relationship between matrix elements that can influence the prediction, we implement a *ten-fold cross validation* phase. Cross-validation, sometimes called *rotation estimation*, is a model technique to generalize a statistical analysis from an independent data set [89].

The technique works as follows:

1. We divide the input matrix rows into ten random sub-datasets. Each time nine sub-datasets are considered the *training set*, while one sub-dataset is considered the *test set*
2. We apply the predictive method (SVD, SIM1 or SIM2) to the training set, that is made of the nine sub-datasets.
3. When managing the tenth dataset, the *test set*:
   (a) We apply the predictive method to all the test set elements having value 0 in the input matrix.
   (b) We separately predict the values of the elements having 1 in the input matrix, as follows. For every element $A(I, J) = 1$ of the test set, corresponding to the annotation between the I^{th} gene and the J^{th} feature:

i. We delete the annotation present at the position (I, J), setting to zero the element: $A(I, J) \leftarrow 0$. We also set to zero all the elements (I, j) with $j \in [1, m]$ that correspond to annotations between the I gene, and the j columns that are kin (ancestors or descendants) of the analyzed term J

ii. We re-set to 1 all the elements (I, j) related to the terms j ancestors of terms part present annotations (having value 1) in the row I after the cancellation made at the previous point.

iii. If, at this point, the row I contains at least a present annotation (that is a 1 valued element), we apply the predictive method on the row I.

iv. We apply the *anomaly correction* (described in Section 5.5) for the row I

v. We take the predicted value for the element (I, J), consider it as the final predictive value, and substitute it the (I, J) element resulted at the point 3.a (prediction on the *test set*).

(c) We repeat the operations *(a) − > (b)* for other nine times, using a different *test set* every time, and other different nine sub-datasets as *tranining set*.

4. We create the final $\widetilde{A}$ output reconstructed matrix by joining all the predicted elements of the ten training sets.

This operation makes the prediction of every single annotation much more independent from the others, and avoid the possible *overfitting*, that occurs when a statistical model describes random error or noise instead of the underlying relationship.

Input and output matrix comparison The entries of the reconstructed matrix $\widetilde{A}$ are real valued. We introduce a threshold τ such that, if $\widetilde{A}(i, j) > \tau$, then gene i is predicted to be annotated to term j. Subject to the original values assumed by the matrix A, the following cases may befall:

- If $A(i, j) = 1$ and $\widetilde{A}(i, j) > \tau$, the annotation of gene i to term j is confirmed; this case is denoted as an *annotation confirmed* (AC), with respect to the original $A(i, j)$. This annotation type can be considered similar to a True Positive (TP).

- If $A(i, j) = 0$ and $\widetilde{A}(i, j) > \tau$, a new annotation is suggested; this case is denoted as an *annotation predicted* (AP), with respect to the original $A(i, j)$. This annotation type can be considered similar to a False Positive (FP). These annotations are those that in the end are inserted in the likely predicted annotation lists, useful for biologists and physicians.

- If $A(i, j) = 1$ and $\widetilde{A}(i, j) \leq \tau$, an existing annotation is suggested to be semantically inconsistent with the available data; this case is denoted as an *annotation to be reviewed* (AR), with respect to the original $A(i, j)$. This annotation type can be considered similar to a False Negative (FP).

- If $A(i, j) = 0$ and $\widetilde{A}(i, j) \leq \tau$, the annotation is not present in the original annotation set and it is not suggested by the analysis; this case is denoted

as a *non existing annotation confirmed* (NAC), with respect to the original $A(i, j)$. This annotation type can be considered similar to a True Negative (TN).

We summarize these four categories in Table 2.

Table 2: The four annotion categories based on the database analyzed condition versus the software prediction condition.

	in the analyzed database	
	presents	absents
suggested presents by our software	**AC**	**AP**
suggested absents by our software	**AR**	**NAC**

As made by Khatri et al. in [54], we chose the rate of τ as the value that minimizes the number of presumed errors (APs + ARs), as explained in the next paragraph. Alternatively, τ value could be chosen on the basis of how many best annotations one would like to consider (e.g. best 10, best 100, etc).
As we stated before, since scientists and biologists discover new biomolecolar annotations every day, and also review and correct the old ones, the annotation databanks cannot be considered definitive. Their (current or future) datasets cannot be used as the true *gold standard* to appraise the correctness of our predictions.
The only item of information that we can count on is that newer checked annotation datasets mean better likelihood of correctness. That is why, for the previously introduced annotation types, we do not use the classical names like TP, TN, FP, FN, but instead use names related to the annotations, instead.

We depict the complete workflow of the prediction algorithm in Fig. 9.
An important and interesting difference between our method and that of Khatri et al. [54] is the choice of the predicted annotations. Khatri et al. [54] took all the predicted annotations above a certain threshold τ as equally correct, our method provides the predicted annotations in order of accuracy. In fact, the value $\widetilde{A}(i, j)$ represents the likelihood that the gene i is annotated to function j. The higher it is, the more likely the annotation exist.

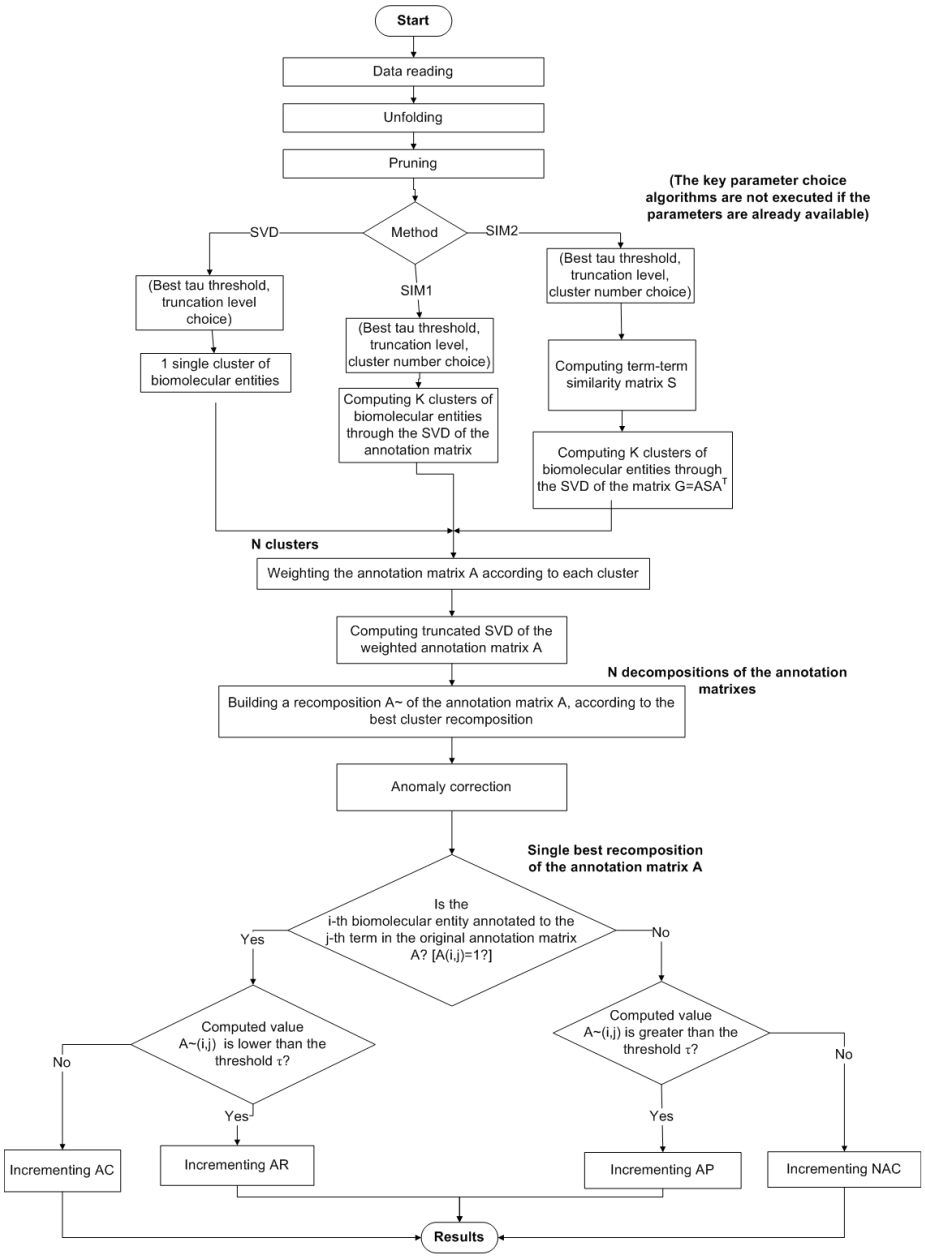

Figure 9: The flow chart describing our complete algorithm prediction procedure.

5.6 Key parameter choice

In the described algorithms (SVD, SIM1, and SIM2) there are some key parameters that can strongly influence the result of the prediction. We describe them in this subsection.

SVD truncation level In the Truncated SVD algorithm described in Section 5.2, a key role is played by a parameter: the truncation level.
How to choose the best SVD k truncation level remains an open challenge, because this choice can strongly influence the resulting matrix $\widetilde{A}$, and greatly affects the quality of the obtained solution.
In the past, other scientists have dealt with this issue in various ways, as we reported in the *Related works* Section 2.5. In [54] [55] [56], Khatri and colleagues use the heuristically set value $k = 500$.

To face this problem, we developed a new algorithm [19] based on Area Under the Curve (AUC) of the Receiver Operating Charactersitc (ROC) curve.
ROC curves are one of the validation analysis method we use to validate our tests. As we describe in Section 7.1, they are an important indicator of the similarity between the input matrix and the reconstructed prediction output matrix.
Our general rule is to find the truncation level that generates the ROC curve with the maximum AUC. The simple way to get it would be to performe the Truncated SVD for all the possible truncation level, in the interval $[1, p]$, where $p = min(m, n)$, m is the number of genes-rows, and n is the number of terms-columns. Unfortunately, this strategy would be too expensive for our resources.

So, we designed and implemented a new algorithm for the choice of the best truncation level, based on two main goals: minimizing the truncation level (without loss of important singular values), and maximizing the ROC area under the curve (AUC).
For the reconstruction of the matrix A, we want to avoid using the Σ matrix singular values that augment the time costs but not the quality of the $\widetilde{A}$ matrix. On the other hand, the most important Σ matrix singular values are those present in the initial positions. So, minimizing the truncation, while optimizing the quality of the predictions, remains an important goal.
Since an AC is an annotation present in input and predicted confirmed in output, and a NAC is an annotation absent in input and confirmed absent in output, having a high number of them means having many confirmations. With many ACs and NACs, the $ARrate$ (in equations (15)) tends to zero and the $ACrate$ tends to one. This corresponds to having a large AUC.
For these reasons, the best prediction performances corresponds to larger AUCs, which can be found through a classical optimization. A computationally inefficient way to choose the best truncation for the SVD would be to perform the SVD with all possible truncations, and then choose the one that provides the largest AUC. The possible truncation levels range from 1 to $p = min(m, n)$.

Therefore, we developed a novel algorithm to automatically select the best SVD truncation based on the optimization of the ACrate vs. APrate ROC curve, that we described in the [19].
The steps of this algorithm are the following:

1. We heuristically choose an integer variable $step = 10 \cdot N$, where N is the number of the singular values of the Σ matrix.

2. for i=1...10, we apply the SVD with truncation level $k = i \cdot step$, compute the ROC curve (with ten-fold cross validation, as explained in Section

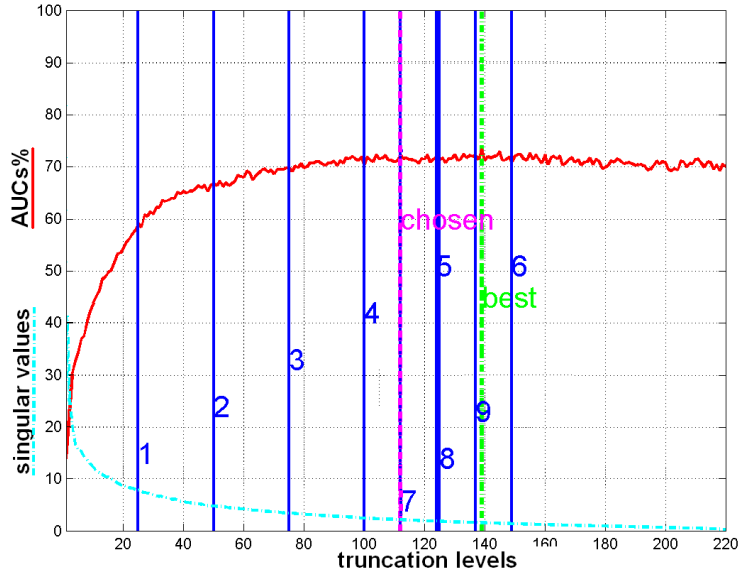

Figure 10: Illustration of the algorithm behavior for the dataset Gallus gallus BP. The upper continuous red line represents every AUC value (in percentage, on the y axis) for any truncation (on the x axis). The vertical lines represent all the truncations chosen by the algorithm for this dataset; the numbers indicate the order in which they were computed. The truncation chosen by the algorithm as best is the dotted line and labeled *"chosen"*, while the maximum AUC is the dotted green line and labeledl *"best"*. The lower dotted cyan plot represents the A matrix singular values.

5.5), and then compute the AUC.

(a) If the AUCs of all the three subsequent samples decrease, or the AUC differences of the last three singular values are lower than $\gamma = 10\%$, we select the truncation having the higher AUC as *best* and exit.

(b) Otherwise, we keep computing the AUCs for all the subsequent sampled singular values.

3. After the first sampling is complete, we iterate the search in the new sub-interval, by using a new sampling interval equal to half of the previous. The new sub-interval starts where the global minimum AUC truncation level is. We name this operation *zoom*.

4. We continue this iteration until in the considered sub-interval there are no more singular values to compute a new AUC, or until the maximum number of sub-interval computation ($numZoom = 4$, heuristically set) is reached, or until the two new truncations computed generate AUCs smaller than the maximum AUC previously computed.

Once the search is finished, we take as optimal truncation the one that corresponds to the maximum AUC among those computed during the search.
As a general indicator, we report an illustration of the behavior of the algorithm for the dataset Gallus gallus Biological Process in Fig. 10.

We describe and discuss the results of the tests made to evaluate this algorithm in Section 8.

Prediction likelihood threshold Once we compute the reconstructed output matrix with the methods SVD, SIM1, or SIM2, we use it to compare each element with its corresponding tuple in the input matrix.
The *annotations predicted* (AP) are the annotations absent in the input dataset and predicted present by our software; and the *annotations to be reviewed* are the annotations present in the input dataset but suggested absent by our software.
Since APs and ARs can be considered as *presumed errors*, we chose the rate of τ as the value that minimizes their sum (APs + ARs), as made by Khatri et al. in [54].
We design the following function:

$$f(\tau) = |AP|_\tau + |AR|_\tau \tag{14}$$

where $|AP|_\tau$ ($|AR|$) are the numbers of APs (ARs) computed by our algorithm when using τ as likelihood threshold. Once we compute this function, we then select the τ value that minimizes it, and consider it then best tau threshold for that dataset.
As a general indicator, we report in Fig. 11 the image of the $AR + AP$ function for the datasets of the annotations made by Cellular Component functions of the Bos taurus (cattle) genes.

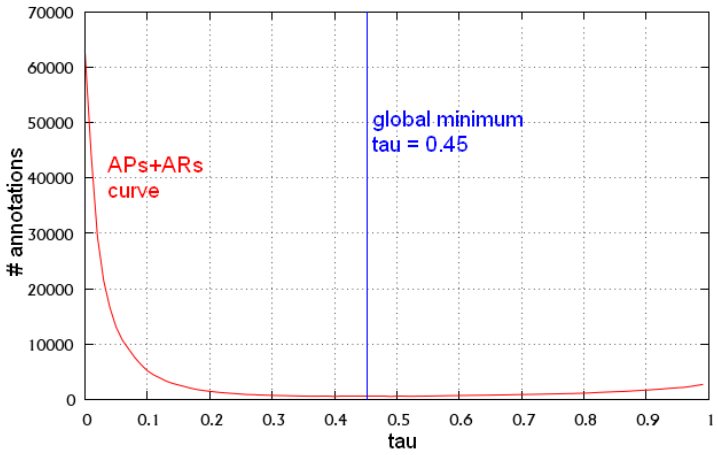

Figure 11: Depiction of the $ARs + APs$ function for the datasets of the annotations made by Cellular Component functions of the Bos taurus (cattle) genes. The red line is the function in Equation (14), while the blue line represents the global minimum, $\tau = 0.45$

Number of gene clusters In the Semantically IMproved methods (SIM1 and SIM2), an important role is played by the cluster number parameter.
Since the algorithm chooses the output matrix annotation profile that minimizes the difference from the original profile among all the possible cluster profiles (as explained in Section 5.3), the higher the cluster number is, the more similar the

output matrix results to the input matrix.

An high number of cluster would raise some problems. First, the reconstructed output matrix $\widetilde{A}$ would be too similar to the input matrix, and would minimize the number of new annotations, eliminating both the noise (annotaions present in input but suggested absent in output: annotation to be reviewed, ARs) and the important annotations predicted (APs). With an identical reconstruction of the output matrix, we would not have a list of predicted annotations, that is the main goal of our application.
Another issue would be raised by the time. A high number of clusters means multiplying the duration time of a cluster reconstruction.
For these reason, we consider prefer to select a low number of clusters, that is between $C = 2$ and $C = 1\% \cdot p$, where p is the maximum between gene number and term number.

As made for the SVD truncation level in Section 5.6, we develop an algorithm based on the Area Under the Curve (AUC) of the Receiver Operating Characteristic (ROC) curves to find the best cluster number. As usual, the ROC is computed with ten-fold cross validation, as explained in Section 5.5.

Since choosing the cluster number $C = 1$ means put all the genes into only one cluster, it is the same of running the SVD method. For this reason, we start our algorithm from $C = 2$.
The steps of the algorithm are the following.

for $i = 2...(1\% \cdot p)$:

1. We run the SIM algorithm with $C = i$ clusters, and we save its ROC AUC percentage (AUC_i).

2. After the third sampling (when $i = 5$), the algorithm states if a first local maximum AUC is present in the AUC list. We denote this AUC with flm.

3. After the fifth sampling (when $i = 7$), the algorithm states if a second local maximum AUC is present in the AUC list. We denote this AUC with slm.

   (a) Once the algorithm finds the two local maxima, we compute the difference between them:

      i. if slm is greater than flm, we take slm as the best area, and return slm_{index} as the best cluster number.

      ii. otherwise, if slm is more than 1/3 lower than flm, we select flm as the best area, and return flm_{index} as the best cluster number.

      iii. otherwsie, the algorithm keeps on searching for another local maximum.

4. Otherwise, if the algorithm does not find two local maxima, the global maximum is selected as best AUC, and its cluster number is selected as best cluster number.

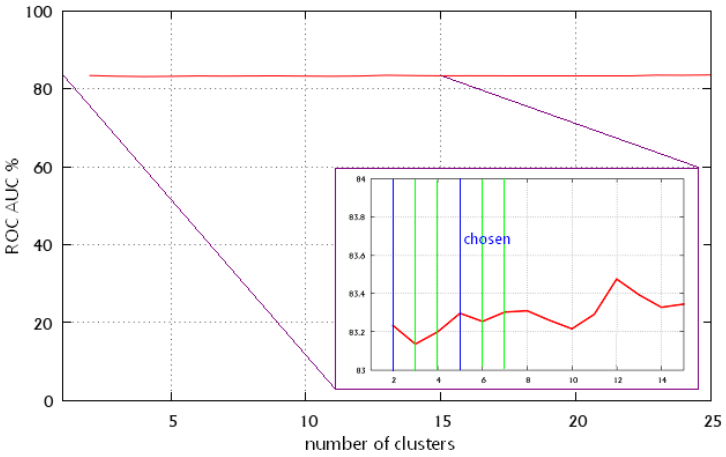

Figure 12: Depiction of the behaviour of the best cluster number algorithm applied to the Danio rerio BP dataset, with the SIM1 method. The red line represents the ROC AUC percentages generated by the SIM1 method when the number of clusters C varies. In the zoom, we see the trend between 2 and 15 clusters, and between 83% and 84%. The green lines highlight the cluster numbers chosen by the algorithm, and the blue lines highlight the local maxima found. Among them, the chosen cluster number has the label "chosen".

We design this algorithm with the intent to balance the effectiveness of having a high cluster number (that leads to a more precise reconstruction of the input matrix), maximizing the ROC AUC to minimize the number of presumed errors, and having a low cluster number (that makes the system save performance time). As a general indicator, we report in Fig. 12 the algorithm behavior when it selects the best cluster number for the SIM1 method applied to the Danio rerio BP dataset. We can see that the algorithm finds two different local maxima within the first positions (2 and 5), and selects the latter because produces a greater AUC (83.30%) than the former (83.23%).
In this case, the difference between AUC percentage is very little (less than 1%), but in other cases may be bigger ($\simeq$10%).
We list the cluster numbers chosen in the prediction results section, Section 9.2. We describe and discuss the results of the tests made to evaluate this algorithm in Section 8.

5.7 Computational complexity

The time complexity of SVD and SIM methods is strictly related to the complexity of the Singular Value Decomposition.
According to [90], the computational complexity of the Lanczos-based approach we have been using to approximate the actual SVD of the matrix A is:

$$O(k \cdot ((z \cdot u) + n))$$

where z is the number of non-zero elements of A, u is the complexity of the *unfolding* operation, k is the SVD truncation level and n is the number of the input features. The *unfolding* operation complexity depends on the number of annotations and on the ontological DAG depth; in the worst case it can be

estimated $O(n^2)$.

Given the dimension of the considered annotations matrices, the computational time is a critical point in the algorithm, and so we tried to optimize as much as possible the software implementation, in particular exploiting where possible the data parallelism.

6 Software

The design and the implementation of the software architecture were mainly made by a former member of our group, Roberto Sarati [8]. We departed from this working platform to add new functionalities and tools. So, this section does not add anything new within this thesis, but just describes the current software framework on which we have been working.

Since the amount of data is very large and the objectives and computation quite demanding, we had to pay particular attention to software performance and memory usage.
To satisfy the facility of the software to be modified and extended, we first chose the Java programming language for implementation. Java results to be very independent from platform and from operating system, that is a very important value for the software objectives. However, Java shows some limitations, too: it provides a high response time, and uses a lot of memory.
The response delay clearly results you comparing matrix operation rapidity in Java and C++ environments with native solutions that use very highly optimized mathematical kernels. The high memory usage problem is especially relevant for the virtual machine.

Given these issues, we decided to implement the mathematical core of our software in C++ programming language, using a multiplatform, multithreading, optimized mathematical kernel, such as AMD Core Math Library (ACML) [92]. This library provides a high level of optimization on generic processors, and is simple to use, given that it does not need to be compiled. In addition, ACML is freely available for both Linux/Unix, Microsoft Windows and Solaris systems. Another effective library we used for the mathematical core was SvdLibC [93]. The multithreading native part was developed by using OpenMP (Open Multi-Processing, OPM) [94] compiler directives which exploited by the mathematical kernels, independently from the operating system. The interaction between the native C++ code and Java code was though Java Native Interface (JNI) [95].

The GPDW database we use is managed by PostgreSQL [100] management system. We implemented the software module able GO term Directed Acyclic Graph (DAG) trees (Section 11) with the Graphviz package [101].

Configuration settings The data reading phase starts with the reading of the external *property.xml* file, where the user can define all the important configuration details, such as the name of the database versions, their address, their ports, the data reading SQL query, the database version comparison validation query, and others.
In the *property.xml*, there is a list of database connections available. Each connection tag has many fields: connection name, driver, url, date, username, password. At the moment, four connections are available, for four different GPDW versions: July 2009 version; October 2011 version; May 2012 version; and March 2013 version.

When launching the database comparison validation procedure (Section 7.2), the software asks to the user which database version he/she wants to use as

analyzed input database, and which database version as *updated* database version for the comparison. In our tests, we used the July 2009 version as *analyzed* database and the March 2013 version as *updated* database.

When, in the future, new versions of the GPDW database (or other databases) will be available, we just will have to insert a new connection tag into the *property.xml* file, and the software will automatically recognize the presence of a new database available for the analysis.

PostgreSQL There are many advantages in using PostgreSQL as database management system for data [100].
First of all, it is *open source*. There is no licensing cost associated to the product, and this make the system usable and reusable without any legal problem. And, in addition, the code is extensible. So, if one needs to customize or extend a PostgreSQL functionality, he/she is able to do it.
It is a system very reliable and stable. All the database administrator community agrees in stating that PostgreSQL rarely crashes. Beyond that, it is cross-platform.
Many database management experts consider PostgreSQL better than other options, such as MySQL [98].

Graphviz tool Graphviz is the short name of *Graph Visualization Software*, an open source package lanched by AT&T Labs Research for drawing graphs specified in *Dot language* scripts [101].
A *Dot language* is a plain text graph description language, where the user can declare all the nodes in the first part of the file, and then list all the links between nodes with the special character $->$
Once the Dot file is complete, it can be processed with Graphviz, that produces a PostScript file from it.

To create the term DAG trees depicted in Section 11, we designed the following procedure:

1. Once the prediction is complete, and we have the final predicted annotations AP_list, we select a gene from it.
2. We retrieve in the analyzed database all the terms to which it is annotated (directed and parental).
3. We write into a Dot file the DAG tree of the gene, made by all the terms and their relationships between them, as black-circled boxes.
4. We add the AP_list terms to the Dot file, as blue-hexagon boxes.
5. We execute the Graphviz compiler with this Dot file as input, and produce the final PostScript file.

In the end, the PostScript file contains the gene term DAG tree, differentiating the already present elements (black circles) from the new predicted elements (blue hexagons), as described in Section 2.2.

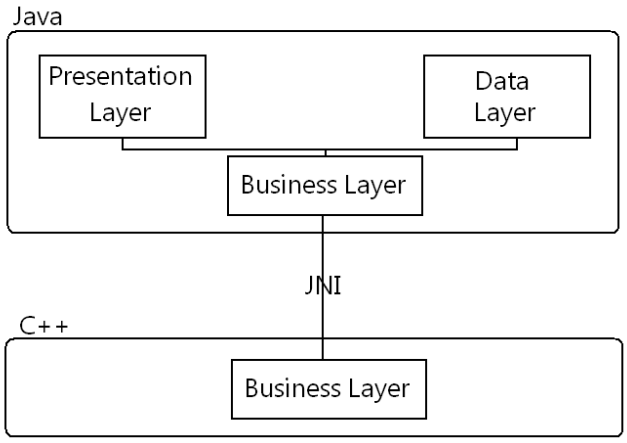

Figure 13: Illustration of the general structure of the application. The structure is based on the *model–view–controller* (MVC) pattern, where the *business layer* is divided into Java component and the C++ component, that communicate through Java Native Interface (JNI).

6.1 Architecture

As described in [8], R. Sarati implemented the application system with two main components: one developed in Java (whose goals are to elaborate the input choice of the user, reading the data for the analys, and finally show the results), and another component developed in C++ able to execute the mathematical operations of the analysis.

We report the architecture structure, built following the *model–view–controller* (MVC) pattern, in Fig. 13.

Java component The Java component has two main roles:

1. To manage the user interface.
2. To manage the data reading phase.

We implemented the user interface with the Swing framework. When running the application, a Java Swing window opens and waits for the user to click on button-commands, as shown in Fig. 14.
The advantages of using Java are manifold. It is *platform-independent*, and can be used in many different systems and integrated development environments (IDE). It is oriented to *distributed computing*, very *robust*, and puts a lot of emphasis on early checking for possible errors, with its strong compilers. Finally, Java is multithreaded, capable for a program to perform several tasks simultaneously within a program.
For the depiction of the ROC curve, we exploit Java JFreeChart library, an useful package able to draw charts and graphics [91].

Once the data reading phase is finished, the software sends a Java DataReader factory object to the C++ component, through the Java Native Interface (JNI) framework.

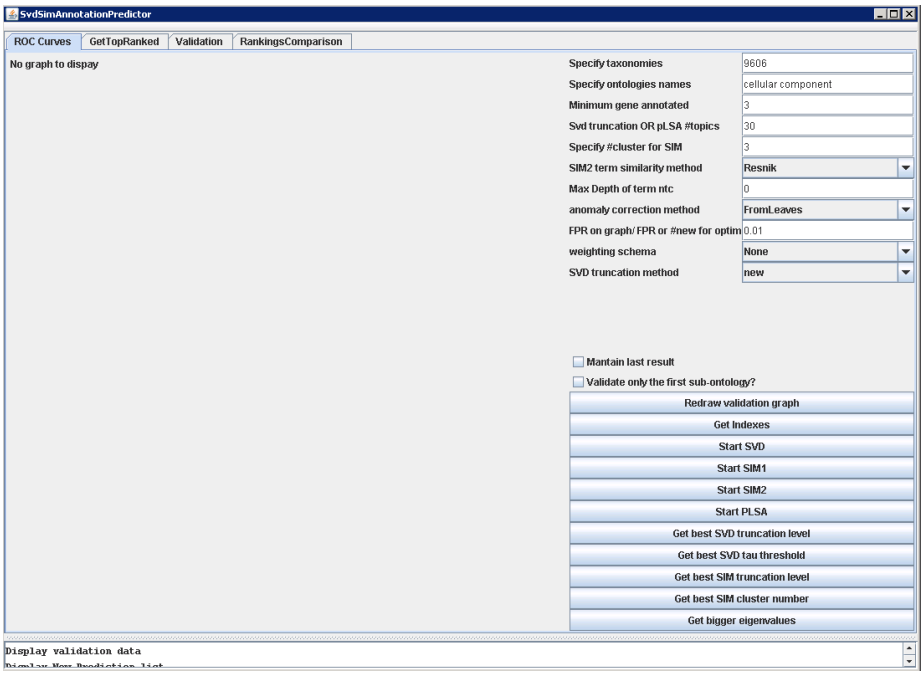

Figure 14: Screenshot of the initial user interface of the application.

C++ component and libraries We needed a way to implement a complex data processing efficiently, and we chose ACML and SvdLibC C++ libraries to manage this issue.
The AMD Core Math Library (ACML) implements a set of multithread functions optimized for *high performance computing*. It is formed by the following main components, relevant to our implementation:

- Complete implementation of the Basic Linear Algebra Subprograms (BLAS) [96]. BLAS is a *standard* application programming interface, supported by most of the mathematical libraries to execute vector and matrix operations. Implemented in different libraries, developed both by open source communities (GotoBLAS, ATLAS, etc) and processor inventors (ACML, Intel's MKL, IBM's ESSL, etc), BLAS provides different operators, mainly for: operations between vectors, between vectors and matrices, and between matrices.

- Complete implementation of the Linear Algebra PACKage (LAPACK) [99]. LAPACK is a set of functions, written in Fortran to make high level scientific calculations. It is mainly used to solve: linear simultaneous equations, systems with linear least squares solutions, factoring problems, and for eigenvalue and eigenvector investigations.

Moreover, the ACML could be compiled for 32-bit and 64-bit systems, on Linux/Unix, Microsoft Windows or Solaris systems. ACML takes great advantage of OpenMP [94] resources; this allows implementation with simple threading

models and a simplified debug. OpenMP also permits multithreading usage on any system.

SvdLibC is a C library based on the SvdPackC library [97], used mostly for SVD and truncated SVD operations. Its intent is to provide an easy to use interface and a set of functionalities to manage, and convert matrices.
The SvdPackC algorithm used in the software is *las2*, which proved to be very powerful for the SVD calculation despite of having some inaccuracies in the minor eigenvalues calculation.

Development process During the implementation we opted for an *iterative and incremental development* approach, that is a combination of both iterative design and incremental build model.
As shown in Fig. 15, this model consists in an initial planning phase followed by a continuous repetition of the core development phase (*planning, requirements analysis, design, implementation, testing, evaluation*), and finally the software deployment.

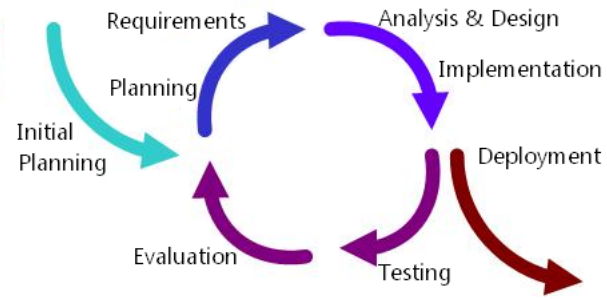

Figure 15: An illustration of the software engineering *iterative and incremental development model.* Image from [102].

This approach is based on the development through repeated cycles (*"iterative"*) and in little additional fractions at a time (*"incremental"*).
These two features permit the programmers exploit what they learned during the development of earlier parts of the system. Both the development and the use of the software make the developers learn new things and possible corrections. When facing a new software module, developers can start with a basic implementation of a subset of the software module, and iteratively enhance it until the full package is implemented. At each iteration, programmers may make design modifications and add novel functionalities.

The iteration approach needs the design, the re-design, the implementation, and the re-implementation to be simple, forthright, and modular, supporting re-design and re-implementation of each module at each stage.

In our software, we started from an initial simple version with just the ROC analysis functionality and a partial predicted annotation list production functionality. First, we tested and evaluated these parts, and we then improved them. Afterwards, we planned, designed, implemented, tested and evaluated the database version comparison component.

Later, we added the latter parts, always following the same development scheme. This approach resulted very suitable for our software, and it also will permit additional further extensions of new functionalities in the future.

6.2 Performances

As previously stated, the use of Java Native Interface code and extremely optimized mathematical libraries allowed much faster execution times. For the prediction of the annotation list, in the case the key parameters of the algorithms are already available (perhaps because just computed for another experiment) our software flow can be divided into six different performance steps.

The first step is the *reading* of initial annotations. As described in the previous sections, data are retrieved from the Java code software and then submitted to the native code. During this phase, the software retrieves all the annotations related to a certain organism, possibly deleting any duplicated annotations, that may be present because of different *evidence codes*.
The second step relates to *annotation unfolding* (described in Section 5.1). This part is executed by an iterative algorithm on the ontology controlled vocabularies native memory structure. We decided that the software would be used to manage already-unfolded data, because this would have drastically increased the database working load.

The third step is the computation of the prediction method (Truncated SVD, SIM1, or SIM2) that is implemented using the Lanczos algorithm in the SvdLibC library. This reduces occupied memory for the initial annotation matrix A, and allows direct computation of the already-truncated resulting matrices. It rebuilds the original matrix, by extracting the $B \in \mathbb{N}$ best annotations from it. This extraction is optimized by the implementation of a data structure (a memory buffer containing the best annotations, over a certain threshold).
The fourth step is the *anomaly correction* (described in Section 5.5), applied to any predicted term of the DAG tree.
The fifth and sixth step are dedicated to the saving of the predicted annotations: first into the Java user interface, then the writing into an XML file.

After this basic flow the user can decide how to move on with its analysis: he/she can choose to check the existence of the predicted annotations on the updated database version (as described in Section 7.2), or to take advantage of the XML annotation list to manually start a literature analysis (as described in Section 7.3), or decide to analyzed the prediction through the *novelty* indicator tool (described in Section 11.2).

Tests were executed on a personal computer with Intel Xeon E5320 1.86Ghz two-processors, 32GB RAM, a Dell PERC 5/i SCSI disk with 256Mb cache, and a Microsoft Windows Server 2008 R2 64-bit operating system.

As a general indicator, in Table 3 we show comparative test performances of the methods applied to the Bos taurus CC dataset (one of the smallest available, as described in Table 1).
As you cas see in Table 3, the most onerous step is prediction, because it has

Table 3: Performance tests time for the application of the three methods to the dataset Bos taurus - Cellular Component, when the key parameters (SVD truncation level, SIM cluster number, likelihood threshold) are already know before the computation, perhaps retrieved by a previous test. *ms*: milliseconds.

Execution performance Dataset: Bos taurus, Cellular Component $k = 25$, $\tau = 0.45$, $C = 3$						
	SVD		SIM1		SIM2	
	ms	%	ms	%	ms	%
Data reading	1,045	17.55	1,049	15.47	1,079	15.07
Unfolding	93	1.56	102	1.50	90	1.26
Prediction	1,743	29.26	2,311	34.08	2,470	34.49
Anomaly correction	2,689	45.15	2,937	43.31	2849	39.78
UI annotation list saving	194	3.26	190	2.80	286	3.99
XML writing	192	3.22	192	2.83	387	5.40
Total	5,956	100.00	6,781	100.00	7,161	100.00

to rebuild the initial matrix and order all the obtained values to complete the ranking operation. SIM methods show similar performance times, with a little increment corcerning SIM2, because of the computation of functional similarity between terms in S matrix. Concerning *Truncated SVD execution*, the performance time is proportional to the number of clusters used in SIM methods (in this case, 3 clusters).

Data reading, Unfolding, and *Annotation list saving and writing* phases need similar time for all the methods, while the *unfolding* phase times are slighty different, because of the different output lists made by the three procedures.
The most expensive method SIM2, that is also the method with the highest number of operations, needs 7,161 ms, that are equal to 7 seconds and 161 ms.

Table 4: Performance test time for the application of the three methods to the dataset Bos tarurus - Cellular Component. *ms*: milliseconds.

Execution performance Dataset: Bos taurus, Cellular Component						
	SVD		SIM1		SIM2	
	ms	%	ms	%	ms	%
Data reading	1,045	2.43	1,049	1.07	1,079	1.12
Unfolding and pruning	96	0.22	105	0.11	93	0.10
Best truncation choice	36,884	85.64	36,871	37.44	36,853	38.14
Best cluster number choice			54,568	55.41	54,568	56.48
Best threshold choice	224	0.52	249	0.25	255	0.26
Prediction	1,743	4.05	2,311	2.35	247	0.26
Anomaly correction	2,689	6.24	2,937	2.98	2,849	2.95
UI annotation list saving	194	0.45	190	0.19	286	0.30
XML file writing	192	0.45	192	0.19	387	0.40
Total	43,067	100.00	98,472	100.00	96,617	100.00

In Table 4 we can see the performance time of the complete execution of the software, with the automated procedures able to detect the best key parameters. The time dedicated to the choice of the best parameters represents the highest percentage: for SVD, the best truncation level choice occupies the 85.64% of

the total computation; the best truncation choice and the best cluster number represent the 37.44+55.41=92.85% of time for SIM1, and 38.14+56.18=94.32% of time for SIM2.
This is due to the repetition several times of the predictive method, of the ten-fold cross validation, and of the ROC depiction, to choose the best key parameters for the methods.

The performance time strongly increase when the annotation dataset size increases, while step percentages remain the same as described in Table 3 and Table 4. For example, the performance of our SVD method with fixed parameters ($k = 1$, $\tau = 0.43$) to the complete human Gene Ontology (CC+MF+BP) annotations takes 11,989,799 ms, equal to 3 hours 19 minutes 49 seconds 799 milliseconds. An improved restoration of the software code, in order to save time, is surely needed for the next future.

7 Validation

In this section we describe the manifold *validation* procedures we designed and implemented to test the effectiveness of our computational prediction methods: Receiver Operating Characteristics curve analysis in Subsection 7.1, the comparison between different version of the same database in Subsection 7.2; the check based on the literature review in Subsection 7.3.

7.1 Receiver Operating Characteristic (ROC) curve analysis

A *Receiver Operating Characteristic (ROC)* curve is a graphical plot which depicts the performance of a binary classifier system while its discrimination threshold is varied [68].
Differently from its original definition, we do not use *TPrate* and *FPrate*, but *ACrate* and *APrate* as previously defined in Section 5.5. Our ROC curves depicts the trade-off between the *ACrate* and the *APrate*, where:

$$ACrate = \frac{AC}{AC + AR} \qquad APrate = \frac{AP}{AP + NAC} \tag{15}$$

for all the possible values of τ. Notice that, in statistical terms, $ACrate = Sensitivity$ and $APrate = 1 - Specificity$. Our ROC curves are built with the $ACrate$ on the y axis and with the $APrate$ on the x axis.
In our tests, we considered only the $APrate$ in the normalized interval $[0, 1]\%$, in order to evaluate the best predicted annotations (APs) having the highest likelihood score, because the more NACs we have, the closer the APrate is to zero.

This ROC curve analysis is an efficient tool to understand the dissimilarity between the input annotations and the output annotations. A ROC curve showing a high Area Under the Curve (AUC) corresponds having many ACs (annotations present in input and confirmed present in output) and many NAC (annotations absent in input and confirmed absent in ouput). This means that the input matrix is very similar to the output matrix, and the output annotation profiles strongly refletct the input ones.
On the contrary, a low AUC means a lot of differences between the input and the output annotations.

Obviously, we are aware of the difficulty related to this ROC curve indicator: since the input dataset and the output dataset are made of the same annotations, if we simply compared the input matrix with itself (without any reconstruction phase), we would obtain the maximum AUC value 100%. This value would seem to mean an optimal prediction, but in fact would give no useful information about new predicted annotations.
We consider ROC curve analysis a good dissimilarity indicator, but we know that is less useful and efficient than other validation methods, such as the database version comparison (Subsection 7.2) and the literature review check (Subsection 7.3).

Given an output matrix $\widetilde{A}$, we consider it coming from a good reconstruction if its ROC AUC is greater than $\omega = 2/3 = 66.67\%$. This heuristically fixed value means that at least 2/3 of the output matrix has been correctly reconstructed.

7.2 Database version comparison

Another validation procedure we designed and implemented is the tally of the AP annotations predicted by our software that are found confirmed in a new version of the analyzed database.

To do this, we involve different SQL queries on different GPDW versions (see [86] and Section 4). In our tests, we use a GPDW version of July 2009 as analyzed database, and a GPDW version of March 2013 as updated database for the comparison.
Our databases use the PostgreSQL database management system [100], that unfortunately does not allow to execute SQL queries from involving a JOIN operation between tables present in different databases.

As software output, we obtain the list of the predicted annotations APs, made by a gene OID code, a term OID code, and a real value representing the prediction likelihood.

Procedure Our procedure behaves this way:

1. We save the AP_list into a table in the analyzed database, having columns *geneOID*, *termOID*, and the likelihood *value*.

2. Since the OID codes are unique IDs only in the analyzed database, we first have to enrich the AP list with additional fields to univocally identify genes and terms also in the updated database.
   We execute an SQL query on the analyzed database to enrich the AP list table, list by adding additional columns such as *gene_source_id*, *gene_source_name*, *term_source_id*, *term_source_name*.
   The *gene_source_name* field is the name of the source databank of the gene (e.g. Entrez Gene), *gene_source_id* is the unique ID of the gene inside that source databank, *term_source_name* is the name of the source databank of the term (e.g. Gene Ontology), and *term_source_id* is the unique ID of the gene inside that source databank.
   The *gene_source_name* and *gene_source_id* fields together univocally identify a gene; and the *term_source_name* and *term_source_id* fields together univocally identify a feature term.
   We create a new AP_list_enriched table with the AP annotations and these fields.

3. We read this AP_list_enriched table from the analyzed database, and create and copy it into the new updated database.

4. We execute an SQL query that retrieves all indirect annotations contained in the updated database, and put them into a new UnfoldedAnnotations table.
   The direct annotations are found in the *gene2biological_function_feature*

table, while the indirect are found with a JOIN operation between this table and the *gene2biological_function_feature_unfolded* table.

5. We execute an SQL query that counts how many direct annotations of AP_list_enriched table are found in the *gene2biological_function_feature* table, by joining the univocally fields: *gene_source_id*, *gene_source_name*, *term_source_id*, *term_source_name*.

6. We execute an SQL query that counts how many parental annotations of AP_list_enriched table are found in the UnfoldedAnnotations table, by joining the univocally fields: *gene_source_id*, *gene_source_name*, *term_source_id*, *term_source_name*.

7. Finally, we also count how many of confirmed annotations have evidence *IEA* or *ND*.

This procedure results very effective, but it has to face with some specific issues of the GPDW database we use. One of these issues is the presence of annotations that have multiple tuples into the *gene2biological_function_feature* table, showing different evidence codes.
We report a flow chart of this procedure in Fig. 16.

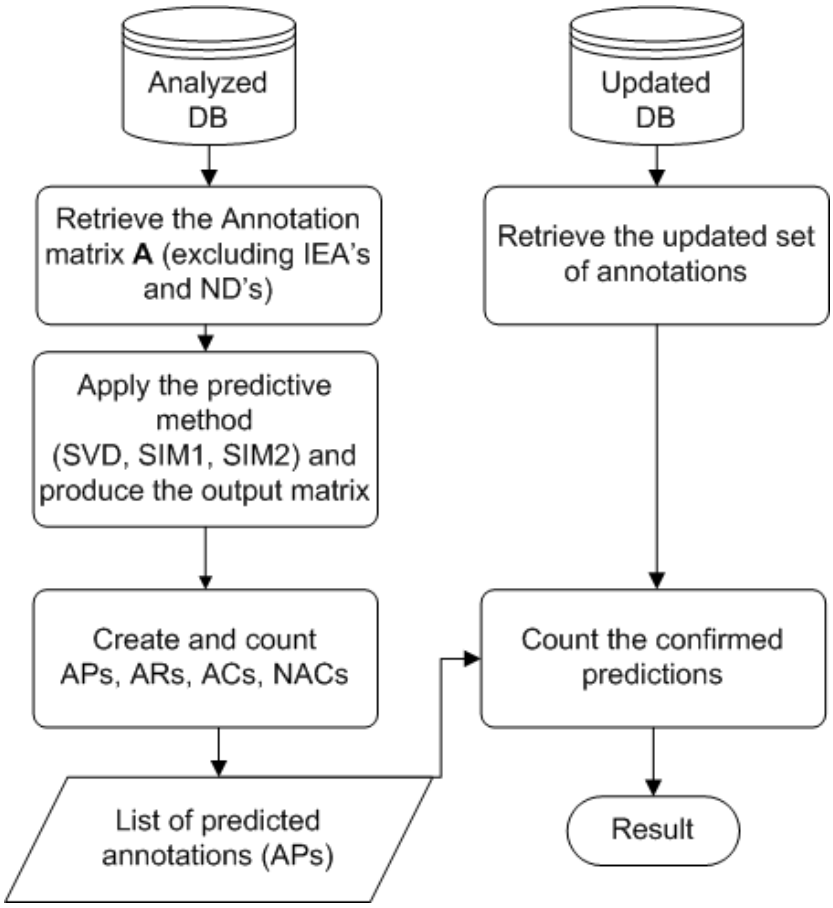

Figure 16: The flow chart of the validation procedure based on the database version comparison.

Annotations with different evidences In the GPDW databases, an annotation can be related to multiple tuples, having different evidence code. For example, in the GPDW July 2009 version, the annotation between the gene SLC25A4 and the term *endoplasmatic reticulum membrane* (GO:0005789) have one tuple with evidence code EXP (Inferred from Experiment), and another one having evidence code IEA (Inferred from Electronic Annotation). The evidence code EXP means that there is a publication that states the existence of that annotation proofed by a biological experiment, while the IEA code means that someone else stated the existence of that annotation on the basis of some computational tool (see Section 2.2 for the GO term evidence code explanation).

To count the IEA and ND annotations found confirmed in our database, we have to consider only the annotations without duplicate tuples, showing different IEA codes.
We reached the goal by using the following pseudo-SQL query:

```
SELECT *
FROM ap_list_enriched AS ann
INNER JOIN gene2biological_function_feature AS g2bff
 [...]
WHERE evidence NOT IN ('IEA', 'ND')

UNION

SELECT *
FROM ap_list_enriched AS ps
INNER JOIN gene2biological_function_feature AS g2bff
 [...]
WHERE evidence = 'IEA'
AND NOT EXISTS (

        SELECT 1
        FROM ap_list_enriched AS psA
        INNER JOIN gene2biological_function_feature AS g2bffA
         [...]
        WHERE psA = ps
        AND evidence <> 'IEA'
)

UNION

SELECT *
FROM ap_list_enriched AS ps
INNER JOIN gene2biological_function_feature AS g2bff
 [...]
WHERE evidence = 'ND'
AND NOT EXISTS (

        SELECT 1
        FROM ap_list_enriched AS psB
        INNER JOIN gene2biological_function_feature AS g2bffB
         [...]
        WHERE psB = ps
        AND evidence <> 'ND'
)
```

This query can be read in three parts:

- In the first part, from SELECT to UNION, we run a simple query to retrieve all the annotations that have evidence different from IEA and ND.
- In the second part, we select all the annotations that have an IEA evidence code (WHERE evidence = 'IEA'), but, among them, we take only those that do not have (AND NOT EXISTS) any other tuple showing evidence code different from IEA (evidence <> 'IEA').
- In the third part, we select all the annotations that have an ND evidence

code (WHERE evidence = 'ND'), but, among them, we take only those that do not have (AND NOT EXISTS) any other tuple showing evidence code different from ND (evidence <> 'ND').

This way, we count only IEA and ND annotations having no duplicates.
Although very reliable, this method anyway cannot be considered as definitive, because new annotations predicted by us might not be inserted into the databank GPDW, yet.

In addition to the number of predicted annotations we find on the updated database, we also check if some of the predicted annotations are present in the analyzed database with the IEA or ND evidence.
Since, as we stated before (in Section 5.1), we exclude all the annotations having evidence code IEA (Inferred from Electronic Annotation) and ND (No biological Data available) (we described evidences in Section 2.2), we expect that some of them may be predicted by our software, too.
If an annotation predicted by our software is found present in the analyzed database, it must have evidence IEA or ND. If not, an error occurred.

7.3 Literature and web tool based analysis

The GPDW data warehouse curators fill the databases with data coming from different source, such as Entrez Gene, Gene Ontology, among others. These sources mainly contain data coming from validated experiments, whose results are published in the literature.
Since there are many biology research groups working independently all over the world, and many different journals where to publish results, some validated annotations published in the literature may not be added yet to the annotation databanks, or anyway many differences and inconsistencies may happen.
For these reason, a literature review in search of confirmation about the existence of the annotations predicted by our methods could be a very effective analysis.

For our publication analysis based on literature review, we mainly take advantage of an online paper repositiories, PubMed [69], and for the web tools exploitation AmiGO [70] and GeneCards [71].

In absence of any efficient aumated tool, we perform this search manually. Starting at the PubMed website homepage, we search for publications containing both the gene symbol (or Entrez Gene ID) and the feature term name.

For example, we want to assess the existence of the annotation <ACYP1, *pyrophosphatase activity*>. The Entrez Gene ID of ACYP1 is 97, while the GO term ID of "pyrophosphatase activity" is GO:0016462.
First, we search for the gene ID 97 in the Gene search section on PubMed, and retrieve the complete name of the gene:
"ACYP1 acylphosphatase 1, erythrocyte (common) type [*Homo sapiens* (human)]"
We then run a paper search on PubMed by using the complete name main part: "ACYP1 acylphosphatase 1". PubMed saves this search in the search History

list (search#1).

Second, we search for papers containing "pyrophosphatase activity" with the classical PubMed search tool. Again, PubMed saves this search in the search History list (search#2).
Finally, by clicking on "advanced search", we are able to run a new search joining the previous two: [search#1] AND [search#2]. If any publications containing both these terms were present, it would be returned by this PubMed query.

Beyond the literature analysis, we can also take advantage of some available web tools able to query online databases in search of information we need.
On the AmiGO website homepage, we insert in the search form the symbol of the gene we want to search for, we set the proper filters (species: Homo sapiens; ontology: Molecular Function, that is the ontology of the pyrophosphatase activity), and run a Gene Product Search.
As output, we get a list of gene products. Each of these items has a link to all its annotations ("view associations"). We then search in this association list if the term "pyrophosphatase activity" is present among them. If present, it shows also the link to the journal publication indicating its evidence.

When using GeneCards, we insert in the website homepage search form the name of the gene symbol we want to analyze (i.e. SLC1A6), and as output we get a sort of "identity card" of the gene. The resulting webpage contains all the gene aliases, summaries, genomic views, proteins, protein domains and families, pathways and interactions, drugs, compounds, and its Gene Ontology associations.
In the list of the annotations linked to it, we search for the term *pyrophosphatase activity* term, or for its GO ID (GO:0016462). Again, If present, it shows also the link to the journal publication indicating its evidence.
Although GeneCards has the same data source (Entrez Gene) of GPDW, this tool may provide the existence of annotations just inserted into Entrez Gene and not added to GPDW yet.

All these three resources result very useful and effective for the predicted annotation review. Among them, the PubMed search is the most reliable and complete, because it can contain articles and publications about annotations that still have to be inserted into AmiGO or GeneCards. Anyway, this two tools result easier to use, and quick for the analysis.
This procedure is very useful to find out new annotations that have not been inserted yet into the GPDW data warehouse. Anyway, the number of predicted annotations found in the literature and in the web tools The absence of an annotation X from the literature may also due to the presence of a publication that states the existence of an annotation descendant Y of the considered annotation.
This may lead to the following situation: biologists discover the existence of the annotation Y, and publish a paper about it. Experts insert the annotation Y into the data bank of confirmed annotations, and, since X is a parent of Y, they insert X into the database, too. Since the insertion of X is caused by the insertion of Y, a publication about X may be absent.

8 Key parameter choice tests and analysis

This section describes the results of the tests made to evaluate the effectiveness of the key parameter choice algorithms described in Section 5.6: SVD truncation choice, and the SIM cluster number.

8.1 SVD truncation level

We tested the effectiveness and efficiency of our algorithms on nine datasets, downloaded from the GPDW data warehouse [86]. The nine different annotation datasets are made by genes of three different organisms (Bos taurus (Bt); Danio rerio (Dr); Gallus gallus (Gg)) and their Gene Ontolgy (GO) annotated terms regarding the biological function features (Cellular Component: CC; Molecular Function: MF; Biological Process: BP), whose quantitative characteristics are described in Table 1.

In Table 5, we report the results obtained in applying our algorithm (described in Section 5.6) to our nine considered datasets. In the Area Diff% column, we show the percentage difference between the AUC computed with the truncation chosen by the algorithm, and the maximum AUC among all the possible truncation. In the #AUCs column, we report the number of AUCs computed before stopping the algorithm.
One may notice how the best truncation *chosen* by the algorithm appears near to the global *best* truncation, that corresponds to the global maximum AUC.
We reported the dimensions of the nine datasets and the results of our algorithm performance in Table 5. One can notice that in two cases (Danio rerio BP and Gallus gallus MF) the algorithm selects the truncation that corresponds to the overall maximum AUC as the global *best*. In many other cases, the difference between the *chosen* AUC and the *best* one is very small: less than 1% (Bos taurus MF, Bos taurus BP, Danio rerio MF, Gallus gallus CC). In two cases where the *chosen* area is not very close from the *best* (Bos taurus CC and Gallus

Table 5: Number of genes *(#gs)*, features *(#fs)* and annotations *(#as)* in the July 2009 GPDW version. Percentage difference between the best area chosen by the algorithm and the maximum area of the dataset (*Area Diff%*), number of SVDs and AUCs computed to select the global best area (*#AUCs*), average time for an SVD and AUC computation (*Time*).

dataset	#gs	#fs	#as	Area Diff%	#AUCs	Time (ms)
Bt CC	497	234	7,658	2.59	6	71,445
Bt MF	234	422	3,574	0.59	10	18,232
Bt BP	512	1,023	18,167	0.93	11	57,416
Dr CC	430	131	4,844	1.18	11	16,653
Dr MF	699	261	4,861	0.24	10	23,693
Dr BP	1,528	1,176	38,624	0.00	10	1,380,400
Gg CC	260	309	3,450	0.33	12	18,531
Gg MF	148	225	1,944	0.00	12	86,679
Gg BP	478	509	8,731	1.06	9	162,730

gallus BP), this is balanced by a smaller number of iterations (#AUCs column in Table 5). Except for one dataset (Danio rerio CC), where we have many iterations and the penultimate largest area, all the dataset tested show that our algorithm hit the target of choosing an AUC near to the maximum one, and maximized the truncation value.

In the Section 9.2 tables we report the prediction results made for several datasets, when varying the truncation level. The SVD method application results with the truncation level chosen by our algorithm always outperform the results produced by the truncation level used by Khatri et al. in [54].

8.2 SIM cluster number

We tested the effectiveness and efficiency of our algorithm (described in Section 5.6) for the best cluster number choice on several datasets. We report the here the tests made for the Homo sapiens datasets downloaded from GPDW [86]. The different annotation datasets are made by genes of Homo sapiens and their Gene Ontolgy (GO) annotated terms regarding the biological function features (Cellular Component: CC; Molecular Function: MF; Biological Process: BP), whose quantitative characteristics are described in Table 1.

Table 6: Comparison between the results of the SIM1 method applied to the the Homo sapiens datasets, when increasing the cluster number. The τ threshold minimizes the sum $APs + ARs$. C: the number of clusters for SIM1. APs: the number of annotations predicted; $anDb$: the number and percentage of predicted annotations found in the November 2009 GPDW version; $IEA/ND\ anDb\%$: percentage of IEAs or NDs among these annotations; $upDb$ ($upDb\%$): number of predicted annotations found in the March 2013 updated db (percentage over the predicted ones). *chosen*: if that cluster number has been chosen by our algorithm.

dataset	k	τ	C	APs	anDb	IEA/ND anDb%	upDb	upDb%	chosen
Homo sapiens, Cellular Component - CC									
CC	378	0.49	**2**	8	0		4	50.00	√
CC	378	0.49	**5**	2	0		0	0.00	
CC	378	0.49	**8**	0	0		0	0.00	
Homo sapiens, Molecular Function - MF									
MF	607	0.48	**2**	51	2	100	4	7.84	
MF	607	0.48	**5**	13	0		1	7.69	√
MF	607	0.48	**8**	5	0		0	0.00	
MF	607	0.48	**12**	5	0		0	0.00	
MF	607	0.48	**15**	5	0		0	0.00	
Homo sapiens, Biological Process - BP									
BP	1,413	0.45	**2**	35	1	100	10	28.57	√
BP	1,413	0.45	**3**	11	0		1	9.09	
BP	1,413	0.45	**5**	2	0		0	0.00	

We report the results of our tests in Table 6. For each test, the cluster number *chosen* by our best cluster number algorithm (described in Section 5.6) is reported in the last column.

We can notice that, as expected, when the cluster number increase, the number of APs decrease. This is due because the output matrix reconstruction becomes more similar to the input one, as the cluster number increases.

In addition, if the cluster number augments, the execution time of the method strongly augments. This happens because, as described in Section 5.4, the algorithm compares all the output gene profiles produced by each cluster number with the input gene profile. So, a cluster number too high would lead to two main flaws: an excessive similarity level between the output matrix and the input matrix, that would eliminate many possible predicted annotations possibly confirmed in the updated database version, and too high time costs.

In Table 6 tests, our algorithm chooses the cluster number that leads to the maximum percentage of predicted annotations confirmed in the updated database version for the CC and BP datasets, while it chooses the second best for the MF dataset.

9 Prediction method validation results and discussion

We first run prediction complete tests for the datasets of annotations made by genes of Homo sapiens and feature terms of the GO Cellular Component, Molecular Function and Biological Process, whose dimensions are listed in Table 1, by using a single sub-ontology in input and in output.

We also report some results for other organism datasets, to find common or different trends and patterns. We also report some result for the multi-ontology analysis, where we run the prediction method on a dataset made of annotations of different sub-ontology, and then consider in output only single sub-ontology annotations.

9.1 ROC analysis

Here we report the ROC cuves we depicted for the Homo sapiens Cellular Component dataset Fig. 17, for the Homo sapiens Molecular Function dataset Fig. 18, and for the Homo sapiens Biological Process dataset Fig. 19.

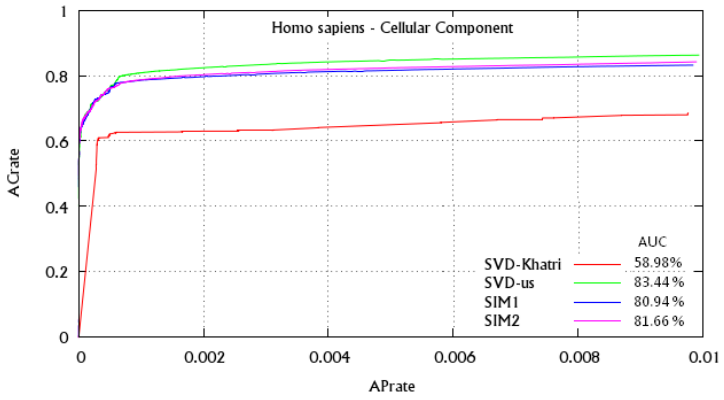

Figure 17: ROC curves for the Homo sapiens CC dataset. SVD-Khatri has $k = 500$; SVD-us, SIM1, SIM2 have $k = 378$; SIM1 and SIM2 use $C = 2$, and SIM2 uses Resnik measure.

As one can notice in Fig. 17, in Fig. 18, and in Fig. 19, the ROC AUCs slightly decrease when we use our SVD, SIM1, and SIM2 methods. This means that the reconstructed output matrices get more different from the input one, when using SIM1 and SIM2 methods, and so we have more AP and more NAC annotations.
Anyway, all the ROC AUCs are greater than $\omega = 66.67\%$, that is the minimum "reconstruction reliability" threshold we consider for our reconstructions, as shown in Table 7.
Since the SIM1 and SIM2 methods are focused on choosing the gene row that minimize the *Euclidean norm* between the clustered predicted gene profile row and the input matrix profile row, one can expect that the ROC curve built by SIM1 or SIM2 methods is always greater than the ROC curve made with the SVD method when using the same parameters. This may not be true sometimes, because SIM1 and SIM2 methods do not choose the reconstructed gene

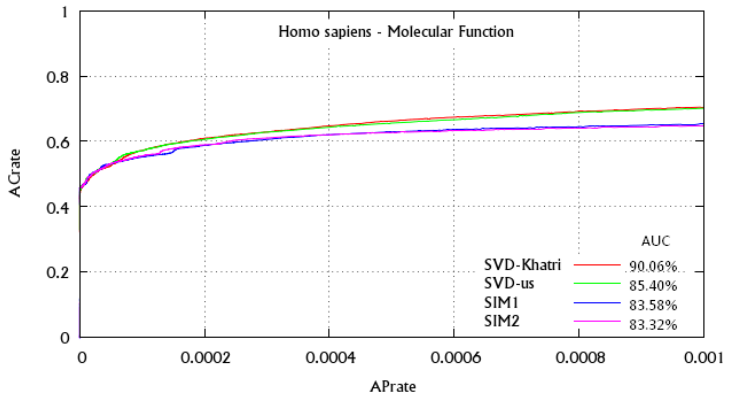

Figure 18: ROC curves for the Homo sapiens CC dataset. SVD-Khatri has $k = 500$; SVD-us, SIM1 and SIM2 have $k = 607$, SIM1 and SIM2 uses $C = 5$, and SIM2 uses Resnik measure.

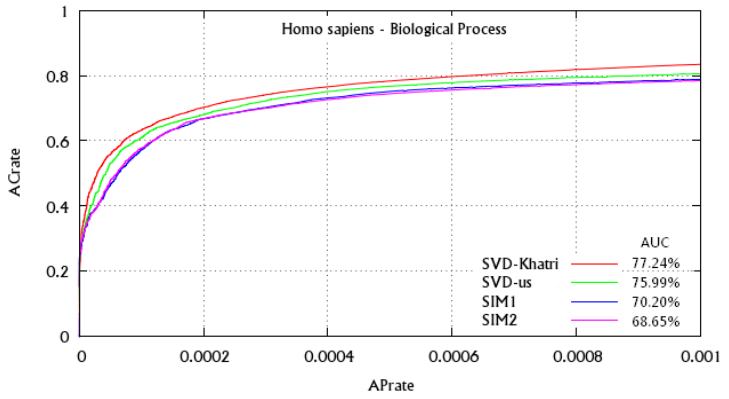

Figure 19: ROC curves for the Homo sapiens CC dataset. SVD-Khatri has $k = 500$; SVD-us, SIM1 and SIM2 have $k = 1,413$; SIM1 has $C = 5$, and SIM2 has $C = 2$ and uses Resnik measure.

Table 7: ROC AUC areas for the three Homo sapiens datasets produced by the four different methods. The AUC area percentage is always greater than the matrix reconstruction reliability threshold ω that we heuristically fixed at 66.67%, except for SVD-Khatri method applied to the CC dataset.

	Homo sapiens		
method	CC	MF	BP
SVD-Khatri	58.98%	90.06%	77.24%
SVD-us	83.44%	85.40%	75.99%
SIM1	80.94%	83.58%	70.20%
SIM2	81.66%	83.32%	68.65%

profile row most similar to the corresponding original input matrix gene profile row, but the row most similar to the profile during the *ten fold cross validation*. As described in Section 5.1, during this phase the gene row undergoes the re-

moval and the re-insertion of each single annotation, before appying the predictive method to the gene row. The *Euclidean norm* is computed between the predictive gene profile and this modified gene profile, and not with the original profile.
This explains why sometimes the AUC of the ROC generated by SIM1 and SIM2 may be lower than the SVD ROC curve computed with the same parameters.

Complete ontology When using all the Gene Ontology annotations of an organism, the ROC curve and the AUC follow trends similar to the sub-ontology case.

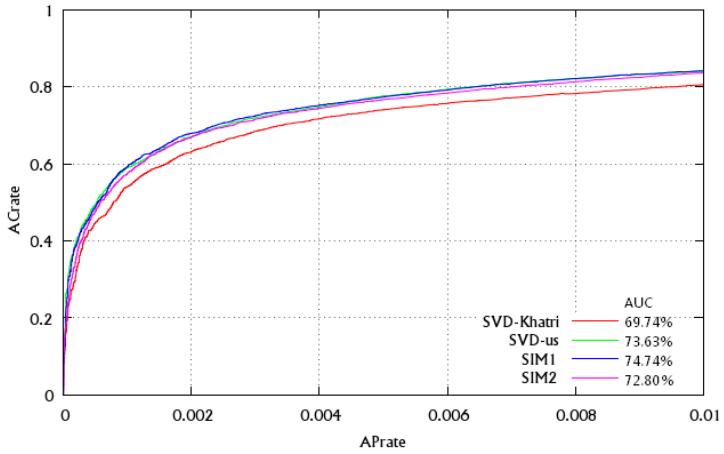

Figure 20: ROC curves for the Bos taurus total Gene Ontology (CC + MF + BP) annotations. SVD-Khatri has $k = 500$; SVD-us, SIM1 and SIM2 has $k = 149$; SIM1 and SIM2 has $C = 2$; SIM2 uses Resnik measure.

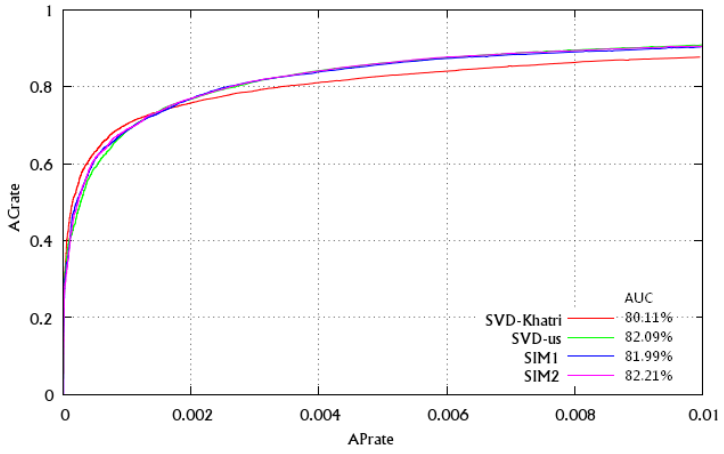

Figure 21: ROC curves for the Danio rerio total Gene Ontology (CC + MF + BP) annotations. SVD-Khatri has $k = 500$; SVD-us, SIM1 and SIM2 has $k = 158$; SIM1 and SIM2 has $C = 8$; SIM2 uses Resnik measure.

For example, we report in Fig. 20 the ROC curves produced by the methods applied to the dataset of the annotations between Bos taurus (cattle) genes and all the Gene Ontology terms (CC + MF + BP), and in Fig. 21 the ROC curves produced by the methods applied to the dataset of the annotations between Danio rerio (zebrafish) genes and all the Gene Ontology terms (CC + MF + BP). As we can see in both Fig. 20 and 21, our methods (SVD-us, SIM1, SIM2) follow a similar trend and outperform the SVD-Khatri method.
All the AUC percentage result greater than the fixed threshold $\omega = 2/3 = 66.67\%$, demonstrating a good reconstruction reliability.

Multiple sub-ontology So far, in the previous tables, we have shown the results when having annotations of an only sub-ontology in the input matrix, compared to the reconstructed output matrix, made of annotations of just one sub-ontology (CC vs. CC, MF vs. MF, BP vs. BP).
To increase the information about correlation between annotation, it is also useful to apply our methods to a matrix made of annotations from different sub-ontologies, and then consider in output just the annotations of a single sub-ontology. For example, having an input matrix made of MF and CC annotations, and then considering the MF annotations in output, could lead to interesting results.

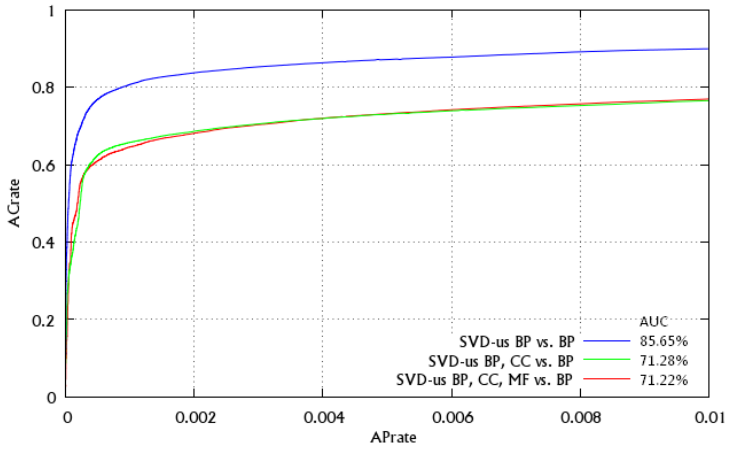

Figure 22: ROC curves for the Homo sapiens BP multiple sub-ontology execution of our SVD mehtod, with $k = 1,413$

For the ROC curve analysis, we report an example in Fig. 22. There we show the ROC curve and the AUC percentage for the application of the SVD-us method with the single sub-ontology BP annotations in input and output (blue line), with BP and CC annotations in input and only the BP annotations in output (green line), and with BP, CC, MF annotations in input and only the BP annotations in output (red line).
We can see in this Figure 22 that the more sub-ontologies we add in input, the lower the ROC AUC becomes. This happens because the output matrix, in which we consider just the BP annotations, becomes more different from the input matrix, when we add further sub-ontologies.

The ROC curve measures the similarity between the input matrix and the output matirx, and adding new input sub-ontologies diminish this similarity level. Despite the dissimilarity, the curves anyway show good levels of reconstruction reliability: all the three ROC AUC are greater than our fixed threshold $\omega = 2/3 = 66.67\%$.
The addition of a sub-ontology in input, in some cases, may lead to better results in the database validation.

Results and discussion From the ROC curves shown in the previous Figures, we can notice that generally our methods SVD-us, SIM1 and SIM2 outperform the SVD-Khatri methods when applied to small datasets, such as Bos taurus CC+MF+BP (Fig. 20) and Danio rerio CC+MF+BP (Fig. 21).
On the contrary, when applied to large datasets such as Homo sapiens single sub-ontology, the SVD method with Khatri truncation level outperforms our methods, on two cases: MF (Fig. 18) and BP (Fig. 19).
This states that our methods produce an output reconstruction matrix more similar to the input matrix when applied to small datasets. Anyway, our methods produce ROC AUC always greater than the fixed reliability threshold $\omega = 2/3 = 66.67\%$, and almost always outperform the SVD-Khatri in the database validation phase, that we show in the next Section.

9.2 Database version comparison

Among all the validation procedures described, the search of the predicted annotations on the updated database is the most important and reliable, because tells directly how many annotations were well predicted. However, as we already said, since the data banks are continuously updated with new annotations, a newer database version is never *the* definitive version.
So, while the predicted annotations found in the newer database version can be considered correctly predicted, all the predicted annotations not found in the new database version cannot be considered *wrong*, because they may be added to the database in the future.

We run the prediction methods SVD, SIM1, SIM2 on several datasets, and then we applied the database comparison analysis validation, as described in Secion 7.2.

We compared the results of the SVD method as used by Khatri et al. [54] (with truncation level $k = 500$, when $m \geq 500$ or $n \geq 500$; or with $k = 10\% \cdot min(m, n)$, otherwise), the results of the SVD method with the truncation level chosen by our algorithm, the results of the SIM1 method with the truncation level and the cluster number chosen by our algorithms, and the results of the SIM2 method with the truncation level and the cluster number chosen by our algorithms and using the Resnik similarity measure.
In Table 8 we report the results for the Homo sapiens datasets, with a single sub-ontology dataset in input and a single sub-ontology dataset in output in the first three cases, and with the complete Gene Ontology (CC + MF + BP) in the last case.
By analyzing the results in Table 8 for the single sub-ontology analysis, one can notice many matters:

Table 8: Comparison between the results of the Truncated SVD with Khatri et al. [54] truncation level (*SVD-Khatri*), Truncated SVD with our truncation level (*SVD-us*), SIM1, SIM2 methods for the **Homo sapiens** datasets. The τ threshold minimizes the sum $APs + ARs$. C: the number of clusters for SIM1 and SIM2. SIM2 uses Resnik's similarity method. *APs*: the number of annotations predicted; *anDb*: the number and percentage of predicted annotations found in the November 2009 GPDW version; *IEA/ND anDb%*: percentage of IEAs or NDs among these annotations; *upDb* (*upDb%*): number of predicted annotations found in the March 2013 updated db (percentage over the predicted ones). In **bold** we write the most important percentage: the number of APs found on the updated database.

method	k	τ	C	APs	anDb	IEA/ND anDb%	upDb	upDb%
Homo sapiens, Cellular Component - CC								
(ROC curves in Fig. 17)								
SVD-Khatri	500	0.45		0	0		0	**0.00**
SVD-us	378	0.49		8	0		4	**50.00**
SIM1	378	0.49	2	8	0		4	**50.00**
SIM2	378	0.49	2	8	0		4	**50.00**
Homo sapiens, Molecular Function - MF								
(ROC curves in Fig. 18)								
SVD-Khatri	500	0.48		108	0		4	**5.56**
SVD-us	607	0.48		81	2	100	5	**6.17**
SIM1	607	0.48	5	13	0		1	**7.69**
SIM2	607	0.48	5	30	0		3	**10.00**
Homo sapiens, Biological Process - BP								
(ROC curves in Fig. 19)								
SVD-Khatri	500	0.48		358	1	100	48	**13.51**
SVD-us	1,413	0.45		64	2	100	12	**18.75**
SIM1	1,413	0.45	2	35	1	100	10	**28.57**
SIM2	1,413	0.45	5	14	0		8	**57.14**
Homo sapiens, CC+MF+BP								
SVD-Khatri	500	0.45		794	196	100	234	**29.47**
SVD-us	1,905	0.43		112	3	100	51	**45.54**
SIM1	1,905	0.43	2	116	3	100	45	**38.79**
SIM2	1,905	0.43	2	111	3	100	49	**44.14**

- Our SVD method always outperforms the Khatri SVD method with fixed truncation, except for the CC dataset where they have the same results. The percentage of annotations found confirmed in the new database version (last column) is greater for the MF and the BP datasets, and equal for the CC term dataset.
- Our SIM1 method always outperforms the SVD methods, except for the CC dataset, where they have the same results, and for the CC+MF+BP dataset, where the SVD-us outperforms all the other methods. In the CC dataset case, the percentage of annotations found confirmed in the new database version (last column) is greater for the MF and the BP datasets, and equal for CC.

- Our SIM2 method always outperforms the the SIM1 and SVD methods, except for CC, where they have the same results. The percentage of annotations found confirmed in the new database version (last column) is greater for the MF and the BP datasets, and equal for CC.
- Our SVD method also predicts 2 annotations that are found as IEA or ND in the analyzed database, for the MF and the BP terms, while the SIM methods just find 1 annotation for BP. This shows an interesting property of the SVD method that behaves better than the SIM techniques.
- For the complete Gene Ontology dataset (CC + MF + BP), we notice an increase on the number of validated predicted annotations, that are much more than the ones predicted by the single sub-ontology experiments. In addition, we notice that SVD-us method outperforms SIM1 and SIM2, too.

Other datasets In addition to the analysis made for Homo sapiens, we report here the database comparison analysis made for two other datasets: Bos taurus (cattle) annotations, and Danio rerio (zebrafish) annotations.
In Table 9 we read the results for the Bos taurus datasets. As for the Homo sapiens datasets, here we notice that the percentage of annotations predicted and found confirmed in the new database version (*upDb%* column) is increasing, as we move from the SVD method by Khatri et al. to our SVD, SIM1, and SIM2 methods.
The only exception are the results of the method SIM2 dataset made of Molecular Function terms: no annotation predicted is found in the update database. This may be due to a too precise reconstruction of the input matrix made through the output matrix. It is possible that the reconstruction removed many AP annotations because wrongly considered them noise.

As for the Homo sapiens datasets (Table 8), also in the Bos taurus SVD methods predict many more annotations found as IEA/ND in the analyzed database.
In Table 10 we read the results for the Danio rerio datasets.

We first notice that all the methods prediction make no annotations found predicted in the updated database, neither present as IEA/ND in the analyzed database. This is probably due to the extremely low number of annotations predicted by the methods: 21 by *SVD-Khatri*, and only 3-2-2 by our SVD-SIM1-SIM2 methods.
With so low numbers of annotations, it is quite unlikely that some of these few annotations can be found in the updated database version.
As reported in Table 1, the annotation set in the updated database is made of 6,395 elements. So, for example, the likelihood of finding one of the 2 annotations predicted in the updated database, is just $P = 2/6395 = 0.031\%$. This may explain the lack of annotations found confirmed.

The other two sub-ontology (MF and BP) results show the same trend of the previously shown results for Homo sapiens and Bos taurus. The percentages of annotations predicted and found in the updated database increase when moving from the SVD-Khatri method to SIM2.

Table 9: Comparison between the results of the Truncated SVD with Khatri et al. [54] truncation level (*SVD-Khatri*), Truncated SVD with our truncation level (*SVD-us*), SIM1, SIM2 methods for the **Bos taurus** annotation datasets. The τ threshold minimizes the sum $APs+ARs$. C: the number of clusters for SIM1 and SIM2. For the *SVD-Khatri* method, since the Bos taurus dataset matrix dimension are lower than 500, we cannot use $k = 500$ as in [54], and instead we use its value proportional to the dataset: $k = 10\% \cdot min(\#gene, \#features)$. SIM2 uses Resnik's similarity method. *APs*: the number of annotations predicted; *anDb*: the number and percentage of predicted annotations found in the November 2009 GPDW version; *IEA/ND anDb%*: percentage of IEAs or NDs among these annotations; *upDb* (*upDb%*): number of predicted annotations found in the March 2013 updated db (percentage over the predicted ones). In **bold** we write the most important percentage: the number of APs found on the updated database.

method	k	τ	C	APs	anDb	IEA/ND anDb%	upDb	upDb%
Bos taurus, Cellular Component - CC								
SVD-Khatri	23	0.42		84	2	100	4	**4.76**
SVD-us	25	0.45		57	2	100	4	**7.02**
SIM1	25	0.45	2	73	0		27	**36.99**
SIM2	25	0.45	2	68	0		27	**39.71**
Bos taurus, Molecular Function - MF								
SVD-Khatri	23	0.37		161	6	100	4	**2.48**
SVD-us	74	0.46		12	1	100	1	**8.33**
SIM1	74	0.46	2	8	0		1	**12.50**
SIM2	74	0.46	5	13	0			**0.00**
Bos taurus, Biological Process - BP								
SVD-Khatri	51	0.40		230	1	100	4	**1.74**
SVD-us	126	0.44		27	1	100	2	**7.41**
SIM1	126	0.44	5	48	0		7	**14.58**
SIM2	126	0.44	2	50	0		8	**16.00**
Bos taurus, CC + MF + BP (ROC curves in Fig. 20)								
SVD-Khatri	500	0.40		57	0		0	**0.00**
SVD-us	149	0.40		109	5	100	27	**24.77**
SIM1	149	0.40	2	162	6	100	45	**27.78**
SIM2	149	0.40	2	150	5	100	32	**21.33**

In the Biological Process dataset, we notice that both the SVD methods lead to very low percentages of annotations predicted and confirmed: 0% for SVD-Khatri and just 0.70% for our SVD-method.
Differently from the previously described Homo sapiens and Bos taurus result tables, here the SVD methods do not predict any annotation found as IEA/ND in the analyzed database
In the last lower part of the table, we see the results for all the three sub-ontology annotations together into a single dataset. Differently from the other cases, here the results of SVD-Khatri method outperforms our SVD, SIM1, SIM2 that anyway show a general improvement trend. Since the SVD-Khatri method predicted just 11 annotations, the percentage of the confirmed annotations strongly increases when an additional annotation is found in the updated

Table 10: Comparison between the results of the Truncated SVD with Khatri et al. [54] truncation level (*SVD-Khatri*), Truncated SVD with our truncation level (*SVD-us*), SIM1, SIM2 methods for the **Danio rerio** annotation datasets. The τ threshold minimizes the sum $APs+ARs$. C: the number of clusters for SIM1 and SIM2. For the *SVD-Khatri* method, since the Danio rerio dataset matrix dimension are lower than 500, we cannot use $k = 500$ as in [54], and instead we use its value proportional to the dataset: $k = 10\% \cdot min(\#gene, \#features)$. SIM2 uses Resnik's similarity method. *APs*: the number of annotations predicted; *anDb*: the number and percentage of predicted annotations found in the November 2009 GPDW version; *IEA/ND anDb%*: percentage of IEAs or NDs among these annotations; *upDb* (*upDb%*): number of predicted annotations found in the March 2013 updated db (percentage over the predicted ones). In **bold** we write the most important percentage: the number of APs found on the updated database.

method	k	τ	C	APs	anDb	IEA/ND anDb%	upDb	upDb%
Danio rerio, Cellular Component - CC								
SVD-Khatri	13	0.52		21	0		0	**0.00**
SVD-us	55	0.48		3	0		0	**0.00**
SIM1	55	0.48	5	2	0		0	**0.00**
SIM2	55	0.48	2	2	0		0	**0.00**
Danio rerio, Molecular Function - MF								
SVD-Khatri	26	0.50		40	1	100	2	**5.00**
SVD-us	88	0.50		21	1	100	3	**14.29**
SIM1	88	0.50	2	22	1	100	4	**18.18**
SIM2	88	0.50	2	13	2	100	5	**38.46**
Danio rerio, Biological Process - BP								
SVD-Khatri	117	0.45		82	0		0	**0.00**
SVD-us	189	0.44		143	0		1	**0.70**
SIM1	189	0.44	2	234	1	100	19	**8.12**
SIM2	189	0.44	2	227	1	100	16	**7.05**
Danio rerio, CC + MF + BP (ROC curves in Fig. 21)								
SVD-Khatri	500	0.44		11	0		3	**27.27**
SVD-us	158	0.44		248	3	100	34	**13.71**
SIM1	158	0.44	8	291	8	100	43	**14.78**
SIM2	158	0.44	8	290	5	100	67	**23.10**

database. In this case, just 3 annotations found in the updated database make SVD-Khatri method outperform the others, that predicted 248, 291 and 290 annotations.

Multiple sub-ontology Again, in the previous tables, we have shown the results when having annotations of a single sub-ontology in the input matrix, compared to the reconstructed output matrix, made of annotations of the same sub-ontology (CC vs. CC, MF vs. MF, BP vs. BP).
As made for the ROC curves, to increase the information about correlation between annotation, we can apply our methods to a matrix made of annotations from different sub-ontologies, and then consider in output just the annotations of an only sub-ontology. For example, having an input matrix made of MF and CC annotations, and then considering the MF annotations in output, could lead

to new interesting results.

Table 11: Comparison between the results of the Truncated SVD with our truncation level (*SVD-us*), SIM1, SIM2 methods when using a single sub-ontology (CC vs. CC) and when using multiple sub-ontology (CC, MF vs. CC) The τ threshold minimizes the sum $APs + ARs$. C: the number of clusters for SIM1 and SIM2. SIM2 uses Resnik's similarity method. *APs*: the number of annotations predicted; *anDb*: the number and percentage of predicted annotations found in the November 2009 GPDW version; *IEA/ND anDb%*: percentage of IEAs or NDs among these annotations; *upDb* (*upDb%*): number of predicted annotations found in the March 2013 updated db (percentage over the predicted ones). In **bold** we write the most important percentage: the number of APs found on the updated database.

method	k	τ	C	APs	AnDb	IEA/ND anDb%	upDb	upDb%
CC vs. CC								
SVD-us	378	0.48		8	2	100	4	**50.00**
SIM1	378	0.48	5	8	0		4	**50.00**
SIM2	378	0.48	5	8	0		4	**50.00**
CC, MF vs. CC								
SVD-us	378	0.48		55	3	100	44	**80.00**
SIM1	378	0.48	5	18	1	100	13	**72.22**
SIM2	378	0.48	5	11	0		4	**36.36**
CC, MF, BP vs. CC								
SVD-us	378	0.48		29	3	100	3	**10.34**
SIM1	378	0.48	5	35	16	100	20	**57.14**
SIM2	378	0.48	5	204	76	100	80	**39.22**

In Table 11, we can see the results that partially confirm our hypothesis. We re-run the tests for our methods (SVD, SIM1, SIM2) reported in Table 8, and we can see in Table 11 middle part (CC, MF vs. CC) that the percentage of confirmed predicted annotations found confirmed in the updated database strongly increases.
For our SVD method, it goes from 50% to 80%, that means a gain of +30%. For the SIM1 method, the percentage goes from 50% to 72.22% (+22.22%). For the SIM2 method, instead, we have the same number of annotation confirmed (4) but the percentage decreases from 50% to 36.36% (-13.64%).

These results show that the additional information provided by the matrix co-occurrences of a second sub-ontology produces the presence of many additional annotations in the reconstructed output matrix, and generally enhance the prediction.

On the contrary, in the lower part of Table 11, we can see the results of the prediction made having annotations of CC, MF and BP terms in input to predict only CC annotations. Here we can notice that the results worsen. On three cases, only SIM1 results outperform the its single sub-ontology, while the others are clearly worse.
This may due to an excessive presence of noise, added by the two additional sub-ontology. Perhaps, the information added by the two sub-ontologies added

too many co-occurrences that are related to some relationship between MF and BP that are not valid for CC.

Result discussion We can draw some conclusions about the results we just showed in the previous tables. We analyzed the results of the application of the methods to the Homo sapiens annotations of the same sub-ontologies in input and in output (Table 8), to the Bos taurus annotations of the same sub-ontology in input and in output (Table 9), to the Danio rerio annotations of the same sub-ontology in input and in output (Table 10), to the Homo sapiens annotations of multiple sub-ontology in input and only the CC sub-ontology in output (Table 11).

We can notice that the results generally get better when moving from the SVD method with Khatri fixed values, to our SVD method, to our new methods SIM1 and SIM2. In addition, the percentage of confirmed annotations on the updated database version increase when the dataset size increases: the results of the Homo sapiens annotation datasets (Table 8) show higher percentages than the results for the Danio rerio annotation datasets (Table 10) and the Bos taurus datasets (Table 9).
In fact, as shown in Table 1, the Homo sapiens annotations are about 15 times the amounts of the Danio rerio and Bos taurus. The more annotations are present, the more information is provided to the methods, and the better the prediction results are, according to our hypothesis.
This is true also for the prediction made with two sub-ontology in input and one sub-ontology in output ("CC, MF vs. CC" Table 11), but not for the case of the three sub-ontologies ("CC, MF, BP vs. CC" Table 11). As we already said, this may due to an excessive "noise" added by the two additional sub-ontologies, that may suggest the existence of associations not possibly related to the Cellular Component terms.

Anyway, our enhanced SIM2 method almost always outperforms the other methods.
As one may notice analyzing the result tables that we reported here, there is no direct correlation between the percentage of predicted annotations found on the updated database and the ROC trends and AUCs reported in the previous Section. Best ROCs do not lead to best database prediction. The ROC curve validation becomes particularly significant when a newer updated database version is not available: in that case, the ROC analysis is able to state which are the methods that work better.

9.3 Literature and web tool based analysis

Once we had the lists of the annotations predicted by our methods, we searched for confirmation of their existence in the literature or/or through web tools, as described in Section 7.3.

Among the predictions made for Homo sapiens with our SVD method with our best truncation level for each single sub-ontology, we found the annotations in Table 12. On the total of 151 annotation predicted (CC: 8, MF: 108, BP:

64), 8 annotations were found on revised and published scientific papers on GeneCards.org or to AmiGO, that is equal to 5.30%

Table 12: The Homo sapiens annotations predicted by our Truncated SVD method (Table 8) and found in the literature. If the annotation has been added to the latest Gene Ontology version, its evidence code is reported. In **bold** the annotations not found in the updated database comparison analysis.

SVD with truncation level chosen by our algorithm Homo sapiens				
ontology	gene symbol	term ID	term name	evidence
MF	SLC1A6	GO:0005313	L-glutamate transmembrane transporter activity.	IEA
MF	HDAC6	GO:0004407	Histone deacetylase activity.	IEA
MF	POR	GO:0004128	Cytochrome-b5 reductase activity.	IEA
MF	NT5M	GO:0008253	5'-Nucleotidase activity.	EXP
BP	ITGA6	GO:0007155	Cell adhesion.	IEA
BP	**ITGA6**	**GO:0007160**	**Cell-matrix adhesion.**	**IEA**
BP	CPA2	GO:0006508	Proteolysis.	IEA
BP	AHR	GO:0006805	Xenobiotic metabolic process.	TAS

Table 13: The Homo sapiens annotations predicted by our SIM1 method (Table 8) and found in the literature. If the annotation has been added to the latest Gene Ontology version, its evidence code is reported.

SIM1 with truncation level and cluster number chosen by our algorithms Homo sapiens				
ontology	gene symbol	term ID	term name	evidence
MF	POR	GO:0004128	Cytochrome-b5 reductase activity.	IEA
BP	CPA2	GO:0006508	Proteolysis.	IEA

Among the predictions made for Homo sapiens with our SIM1 method with our best truncation level and our best cluster number, we found the annotations in Table 13. On the total of 56 annotation predicted (CC: 8, MF: 18, BP: 35), 2 annotations were found on revised and published scientific papers on GeneCards.org or to AmiGO, that is equal to 3.57%

Among the predictions made for Homo sapiens with our SIM2 method with our best truncation level, our best cluster number, and the Resnik semantic similarity measure we found the annotations in Table 14. On the total of 52 annotation predicted (CC: 8, MF: 30, BP: 14), 4 annotations were found on revised and published scientific papers on GeneCards.org or to AmiGO, that is equal to 7.69%

We find a predicted annotations, not found on the updated database version,

Table 14: The Homo sapiens annotations predicted by our SIM2 method (Table 8) and found in the literature. If the annotation has been added to the latest Gene Ontology version, its evidence code is reported.

SIM2 with truncation level and cluster number chosen by our algorithms, Homo sapiens by using the Resnik measure				
ontology	gene symbol	term ID	term name	evidence
MF	POR	GO:0004128	Cytochrome-b5 reductase activity.	IEA
MF	NT5M	GO:0008253	5'-Nucleotidase activity.	EXP
MF	FNTA	GO:0004661	Protein geranylgeranyl--transferase activity.	TAS
BP	CPA2	GO:0006508	Proteolysis.	IEA

through the literature analysis: <ITGA6, *Cell-matrix adhesion*>. We reported it in **bold** in Table 12.
In the literature and web tools analysis, we do not found any publications for the eight Cellular Component annotations found by our algorithms.

As we can see in this example, on the total of 151 annotations for the Homo sapiens datasets predicted by our SVD method, 21 of them were validated in the updated database version, we found only 1 additional annotation in the literature and web tools. Compared to the long time costs to perform this research, this result may seem very constraining. This is why we consider more useful and reliable the two other validation procedures (*ROC curve analysis* and *Database comparison validation*).
Anyway, although very time-consuming, this literature and web tools analysis is able to provide further confirmation to the correctness of the annotation prediction. This analysis represents another additional step in the validation procedure, after the ROC analysis and the database comparison.

Figures and tables recap In the following table we review the list of the Figures and Tables used for the different datasets and method procedures.

Table 15: Validation figures and tables recap.

Dataset	ROC curve	DB validation	Literature validation
Homo sapiens CC	Fig. 17	Table 8	Tables 12 13 14
Homo sapiens MF	Fig. 18	Table 8	Tables 12 13 14
Homo sapiens BP	Fig. 19	Table 8	Tables 12 13 14
Homo sapiens CC+MF+BP		Table 8	
Homo sapiens CC, MF vs. CC		Table 11	
Homo sapiens BP, MF vs. BP	Fig. 22		
Homo sapiens BP, MF, CC vs. BP	Fig. 22		
Bos taurus CC		Table 9	
Bos taurus MF		Table 9	
Bos taurus BP		Table 9	
Bos taurus CC+MF+BP	Fig. 20	Table 9	
Danio rerio CC		Table 10	
Danio rerio MF		Table 10	
Danio rerio BP		Table 10	
Danio rero CC+MF+BP	Fig. 21	Table 10	

10 Annotation list correlation

Another interesting concept to understand how much a list may vary when one of its key parameter change is its similarity correlation with other lists. Comparing two lists produced when varying an important parameter can actually provide some interesting information about how that parameter is able to influence the output results.
In order to find and understand the similarity and correlation levels of two lists of predicted biomolecular annotations, we took advantage of two metrics:

- the *Spearman's rank coefficient* [103], or *Spearman* ρ (Subsection 10.1);
- the *Kendall distance* [104], or *Kendall* τ (Subsection 10.2);

We developed so some significant variants to adapt these metrics to our domain and case studies, descibed in Subsection 10.1.1, in Subsection 10.1.2, in Subsection 10.2.1, in Subsection 10.2.2.
We implemented these measures to understand how much the annotation lists vary when the key-parameters (such as SVD truncation level) of our predictive statistical methods varies.

10.1 Spearman coefficient

The *Spearman's rank correlation coefficient* [103] is measure of statistical dependence between two lists, or two variables. It was named by Charles Spearman (1863 - 1945) and it is often denoted with the Greek letter ρ (pronounced: *"rho"*).
Differently from other measures such as Kendall distance, it is also able to express the *negative* correlation between two lists, that is for example when they have the same elements in the opposite ranking.
If two lists are identical (that is they have the same elements in the same order), the Spearman coefficient is +1, and denotes the maximum *positive* correlation between the lists. On the contrary, when the two lists have the same elements but in exactly the opposite order, the Spearman coefficient is -1, and denotes the maximum *negative* correlation. Finally, if the elements order in the two lists strongly deverges, the Spearman coefficient is 0, and denotes the minimum correlation and maximum diversity.

The keypoint of the coefficient is the *rank* (difference) between the position of each element in the first list and the position of the same element in the second list. For example, if the element i occupies the 7th position in the first list, and occupies the 19th position in the second list, its rank is $|7 - 19| = 12$.

Given an element i, we denote x_i its position in the first list, y_i its position in the second list, $d_i = |x_i - y_i|$, and n the size of the two lists. The final normalized Spearman ρ value is given by:

$$\rho = 1 - \frac{6 \cdot \sum d_i^2}{n(n^2 - 1)} \tag{16}$$

For example, the Spearman coefficient computed for ListX and ListY in Table 16, with $n = 5$, would be:

$\rho = 1 - (6 \cdot 18)/(5 \cdot 24) = 1 - 18/120 = 1 - 0.85 = 0.25$
This value, near to zero, means that ListX and ListY have a little correlation.

Table 16: Example of the Spearman's correlation coefficient usage.

ListX	x_i	ListY	y_i	d_i	d_i^2
a	1	c	4	3	9
b	2	b	2	0	0
c	3	e	1	2	4
d	4	a	5	1	1
e	5	d	3	2	4

The Spearman correlation coefficient is often defined as *nonparametric*. This can have two meanings. First, the fact that a perfect Spearman correlation results when ListX and ListY are related by any monotonic function. Second, its exact sampling distribution can be obtained without requiring knowledge (i.e., knowing the parameters) of the joint probability distribution of ListX and ListY.

Although being very useful, the classical Spearman's rank correlation coefficient is not able to manage efficiently lists having different sizes and/or different elements. To overcome this problem, we design and introduce a new coefficient variant, based on the addition of penalty values to the absent elements of the two lists.

10.1.1 Weighted Spearman coefficient

Previously defined coefficient are able to work correctly only if the two lists compared contain the same elements and have the same sizes. Obviosuly, it is not always true: sometimes two lists may contain different elements and/or have different size. In this cases, a strategy to manage this issue is adding the first list elements missing in the second list to the queue of the second list, and viceversa.
This behavior gives wrong results when ListX and ListY have very different size: for example, if ListX contains 4 elements and ListY contains 800 elements, the Spearman correlation coefficient may state that they are quite similar. This is obviously wrong.

To manage this problem, we introduce a new weighted Spearman coefficient featuring penalty distance weights (w) for the elements absent from a list, as described by Ciceri et al. in [108] paper.
If the element i is present in one list and absent from the other, it weight is the maximum penalty $w_i = 1$.
Otherwise, if the element is present in both lists, it gets a weight proportional to the distance between its position in ListX x_i and its position in ListY y_i.
The two weights are summarized in the 17 formula.

$$w_i = \begin{cases} 1, & \text{Element } i \text{ absence} \\ 1 - \frac{1}{|x_i - y_i| + 1}, & \text{Element } i \text{ presence} \end{cases} \tag{17}$$

For the present elements, we choose this weight function to make the weight increase as the distance between elements increases (as shown in the Fig. 23 image plot).

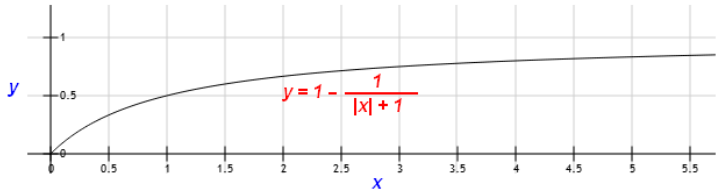

Figure 23: Graphical plot of the trend of the function of the weight presence in Equation 17. The higher the position distance is (on the x axes), the higher the weight results (on the y axes).

If ListX contains n elements, ListY contains m elements, and ListX and ListY contain a total amount of q different elements, our Weighted Spearman coefficient value is given by the formula 18:

$$\rho_w = \frac{\sum_{i=1}^{q} w_i}{q} \tag{18}$$

If ρ_w gets a value near to zero, that means a high correlation, because very few penalties are present and/or elements have similar positions in the two lists. On the contrary, ρ_w gets a value near to one, this means a low correlation between the two lists, because many penalties are presents or elements have very distant positions.
If the two lists have no common elements ($q = n + m$), $\rho_w = 1$.

10.1.2 Extended Spearman coefficient

The Weighted Spearman coefficient resulted very powerful and useful for the general list analysis, but shows a flaw for our context: it considers all the absent elements equal.
In our case, if an annotation is not present in an *annotation predicted* list (AP, the annotations present in the output matrix with likelihood greater than τ threshold), it is likely that is present in the *no annotation confirmed* list (NAC, the annotations present in the output matrix with likelihood less or equal than τ threshold).

In the Weighted Spearman coefficient, an annotation absent from the AP list but present in the NAC list would get penalty 1.0, like an annotation absent from both lists. So, we have to design and create a new coefficient more suitable to our domain, where an annotation-element absent from the AP list but present in the NAC list gets a lower penalty than a annotation absent from the AP list and from the NAC list.

If an element-annotation is present in both lists, it gets the same weight as in the Weighted Spearman coefficient (Subsection 10.1.1, formula 17).
If an element is present in the first AP list, absent from the second AP list, but present in the second NAC list, it gets a stronger weight.

If the element-annotation is present in one list but absent from the other, it gets the same weight as in the Weighted Spearman coefficient (Subsection 10.1.1, formula 17).

We introduce the new penalty by changing the position-weight of the element present in the NAC list:
$z = y + m \cdot 2$
We express the new Extended Spearman Coefficient in the following formula 19:

$$v_i = \begin{cases} 1, & i \text{ absence from AP list and NAC list} \\ 1 - \frac{1}{|x_i - z_i| + 1}, & i \text{ presence 1st AP list and 2nd NAC list} \\ 1 - \frac{1}{|x_i - y_i| + 1}, & i \text{ presence in the AP lists} \end{cases} \tag{19}$$

With this new feature, an element present in the fist AP list and in the second NAC list gets a weight-penalty greater than the weight of an element present in both the AP lists, but lower to the penalty of an element absent from the AP lists and the NAC lists.
The general Extended Spearman coefficient formula is the same as in 18:

$$\rho_e = \frac{\sum_{i=1}^{q} v_i}{q} \tag{20}$$

As for the Weighted Spearman coefficient, if two lists are identical, their Extended Spearman coefficient is equal to zero. If the lists are totally different, with many different elements and many position difference, their their Extended Spearman coefficient is near to one.

10.2 Kendall distance

The *Kendall tau rank distance* [104] is a metric that counts the number of pairwise disagreements between two ranking lists, that is also the number of bubble-sort swaps needed to order one list in the same list of another. Named after Maurice Kendall (1907 - 1983), it is often denoted with the Greek letter τ (pron.: *"tau"*).
Given two lists of the same size and containing the same elements, the more bubble-sort swaps you need to make the elements of the second list in the same order of the first list, the more dissimilar the two lists are, the larger the distance is, and the higher the Kendall distance τ results.
If the two lists are identical, with the same elements in the same order, no bubble-sort swaps are needed, and so the Kendall distance results zero.

Given two lists List1 and List2 having size n, an element i, we denote γ_{1i} its position in the first list, γ_{2i} its position in the second list, γ_1 the general List1 ranking, and γ_2 the general List2 ranking. The number of bubble-sort swaps k is given by:

$$k(\gamma 1, \gamma 2) = |(i,j) : i < j, (\gamma_{1i} < \gamma_{1j} \ \& \ \gamma_{2i} > \gamma_{2j}) \mid (\gamma_{1i} > \gamma_{1j} \ \& \ \gamma_{2i} < \gamma_{2j})| \tag{21}$$

The formula in 21 may be better explained by this pseudo-code section:

```
foreach element i of List1
```

```
foreach element j of List2
 if (position_of_i_inList1 < position_of_j_inList1 &
  position_of_i_inList2 > position_of_j_inList2)
    k++;
```

After the check of every element of List1 against every element of List2, the variable k contains the total number of bubble-sort swaps.

The normalized Kendall distance is given by:

$$\tau = \frac{k}{n(n-1)/2} \tag{22}$$

If the two lists are identical, no swaps are needed and so $\tau = 0$. If one list is the reverse of the other, $n(n-1)/2$ bubble-sort swaps are needed, and $\tau = 1$. Notice that, differently from the Spearman correlation coefficient (Section 10.1), the Kendall distance does not express the negative correlation between the lists. In addition, the Kendall distance is not focused on the position difference of elements, but genuinely on the number of different orders of two elements in the two lists.

For example, we can consider the lists ListX and ListsY already used in 16, now in the following Kendall distance example Tab. 17.

Table 17: Example of the Kendall's correlation measure usage. The first column "pair" indicates the pair of elements analyzed. The second and third columns "ListX positions" and "ListY positions" indicates the position of the pair elements in ListX and ListY, and the sign of relations. For example, *(a,b) 1 < 2 in ListX* means that the element a has position 1 in the ListX, and the element b has position 2 in the ListX.

pair	ListX positions	ListY positions	Bubble-sort swap needed?
(a, b)	$1 < 2$	$4 > 2$	**yes**
(a, c)	$1 < 3$	$4 > 1$	**yes**
(a, d)	$1 < 4$	$4 < 5$	
(a, e)	$1 < 5$	$4 > 3$	**yes**
(b, c)	$2 < 3$	$2 > 1$	**yes**
(b, d)	$2 < 4$	$2 < 5$	
(b, e)	$2 < 5$	$2 < 3$	
(c, d)	$3 < 4$	$1 < 5$	
(c, e)	$3 < 5$	$1 < 3$	
(d, e)	$4 < 5$	$5 > 3$	**yes**

In the Tab. 17 example, the number k of bubble-sort swaps needed to rank ListY in the same order of ListX (or viceversa), is 5.
With $k = 5$ and $n = 5$, we can apply the formula in 22 and get:
$\tau = 5/((5 \cdot 4)/2) = 5/10 = 0.5$
Since the number of possible bubble-sort swaps was 10, and 5 swaps were needed, the Kendall distance of this example results exactly the half between zero and one, and means a precise average correlation between the two lists.

One may notice that the previously defined Spearman correlation coefficient stated in Section 10.1 that ListX and ListY have *little* correlation, while the Kendall distance now assert that they have *average* correlation. This ostensible inconsistency is due to the different type of correlation gathered by the two coefficients: the Spearman measure pays more attention to the distance between positions, while the Kendall measure focuses on the number of different order of pair of elements in the lists.

10.2.1 Weighted Kendall distance

Similarly for what concerns the Spearman coefficient (Subsection 10.1.1), a flaw of the classical normalized Kendall distance coefficient is that it works properly only when the two lists have the same size and the same elements.
Similarly to what we do for the Spearman coefficient, we introduce weights and penalties to consider the elements that are present in one list but absent from the other, as described by Ciceri et al. in [108] paper.
In case of absence from one of the two lists, we add the maximum weight to the total weight value; otherwise, we add a log value proportional to the position of the element, in order to give more weight to a bubble-sort swap made in the first positions, respect to a bubble-sort swap made in the last positions.

$$w_i = \begin{cases} 0.5 & \text{Element } i \text{ absence} \\ \frac{1}{log(i+2)} - \frac{1}{log(i+3)} & \text{Element } i \text{ presence} \end{cases} \tag{23}$$

For the present elements, we choose this weight function to make the weight higher if the element is found in the first position, and lower if it is in the last (as shown in Fig. 24). We consider bubble-sort swap in the first positions more important.

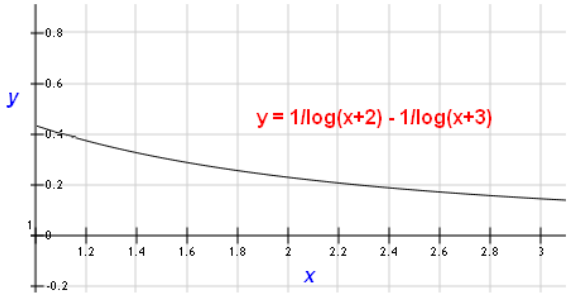

Figure 24: Graphical plot of the trend of the function of the weight presence in Equation 23. The lower the position is (on the x axes), the higher the weight results (on the y axes).

If List1 contains n elements, List2 contains m elements, ListX and ListY contain a total amount of q different elements, and k is the classical normalized Kendall distance described in Subsection 10.2, our Weighted Kendall coefficient value is given by the formula:

$$\tau_w = \frac{\sum_{i=1}^{q} w_i}{k} \tag{24}$$

If on k bubble-sort swaps, every time an element of the two were absent, the

total sum of 0.5 penalties would be $k \cdot 2$, that would mean:
$\tau_w = (k \cdot 0.5 \cdot 2)/k = k/k = 1$
So, $\tau_w = 1$ means minimum correlation and maximum diversity between the two lists. If the two lists were identical and no bubble-sort swaps were needed, w_i would be zero for all the elements, and consequently $\tau_w = 0$.

10.2.2 Extended Kendall distance

As we stated for the Weighted Spearman coefficient, we need a Kendall measure variant that considers also the possibility of having an annotation-element absent from an AP list but present in its related NAC list.
As we introduced the Extended Spearman coefficient in Subsection 10.1.2, we introduce now a new Extended Kendall distance where an element i absent from an AP list but present in its NAC list, and having likelihood value h gets a weight equal to $0.5 - h$.

$$v_i = \begin{cases} 0.5 & i \text{ absence from AP list and NAC list} \\ 0.5 - h & i \text{ presence in 1st AP list and 2nd NAC list} \\ \frac{1}{log(i+2)} - \frac{1}{log(i+3)} & i \text{ presence in AP list} \end{cases} \quad (25)$$

And consequently:

$$\tau_e = \frac{\sum_{i=1}^{q} v_i}{k} \quad (26)$$

The prediction likelihood real value decrease along the position. Annotations in the AP list have values ranked from the maximum to the minimum, in the interval $[1, threshold)$. Annotations in the NAC list are in the interval $[threshold, 0]$. Since the likelihood is proportional to the ranking position, we used it for the new weights.
As for the Weighted Kendall coefficient (Subsection 10.2.1), τ_e gets near to one when the two lists are very diffent, and gets near to zero when the two lists are very similar.

10.3 Results and discussion

We report here the tests made for our Extended Spearman coefficient and Extended Kendall distance on lists of annotations produced with the SVD method on a small dataset (annotations of Homo sapiens genes and Cellular Component terms), when varying the truncation level.

In Table 18 we report the quantitative amounts of AP and NAC annotations for the different truncation level. We selected the truncation levels chosen by our best truncation level algorithm (described in Section 5.2), from the best value chosen ($k = 378$, leading to *List0*).
We can notice in Table 18 that also a little change in the truncation level may lead to strongly different amounts of APs and NACs.

In Table 19 we read the values of the Extended Kendall distances and the Extended Spearman coefficients between the nine considered lists.

Table 18: Amounts of AP and NAC annotations for Homo sapiens CC when varying the SVD truncation level k. The best k is 378, and generates *List0*. The other truncation levels are those selected by our best truncation choice algorithm before finding the best. The likelihood threshold is fixed $\tau = 0.49$

SVD when varying truncation k Homo sapiens CC				
	k	AP	NAC	ROC AUC
List0	**378**	8	4,458,751	83.49%
List1	**402**	2	4,458,757	53.64%
List2	**390**	7	4,458,752	53.58%
List3	**291**	19	4,458,740	53.14%
List4	**349**	8	4,458,751	52.97%
List5	**233**	48	4,458,711	51.82%
List6	**175**	78	4,458,681	48.80%
List7	**117**	86	4,458,673	45.11%
List8	**59**	95	4,458,664	38.82%

By analyzing the order of values of the Extended Spearman coefficients (gray cells), we do not notice any interested trends or patterns. Moving from the first lists to the last, the values seem randomly increasing and decreasing without a specific trend.
The Extended Kendall distance values, similarly, increase as the the list distance increase, getting the values near to 1.0 for almost all the list couples, except for the first ones (*List0* vs. *List1*, *List0* vs. *List2*, *List1* vs. *List2*).

Since the Spearman coefficient is mainly focused on the difference between positions in the two list of the same element, these values mean that the ROC AUC size does not influence directly this position dissimilarity. On the contrary, for the Kendall distance measuring based on the number of bubble-sort swaps needed to make the two list identical (that means measuring the different position number), we notice a strong increase in the distance after the first lists comparison (from List0 to List2).
From the *List3* comparison, all the values get near to 1, showing the values near to the maximum dissimilarity. This means that there is a relation between the ROC AUC and the list orders just for the lists coming from SVD generating very high AUC percentages and having very low AP numbers.

In Table 19, the comparison having higher Extended Spearman coefficient ($\rho = 0.885$) is between *List8* ($k = 59$, $AUC = 38.82\%$), and *List5* ($k = 233$, $AUC = 51.82\%$). The comparison having lowest Extended Spearman coefficient ($\rho = 0.01$) is between *List1* ($k = 402$, $AUC = 53.32\%$), and *List2* ($k = 390$, $AUC = 53.58\%$).
From this last case, similar truncation levels and similar AUC percentages seem to lead to a low Spearman coefficient, and high similarity, but this is not true, for example, for the comparison between *List2* and *List3*.

For what concerns the Extended Kendall distance, that focuses on the bubble-

sort swaps needed to make two lists identical, in Table 19 we see that comparison showing minimum value and maximum correlation ($\tau = 0.491$) is between *List0* ($k = 378$, $AUC = 83.49\%$) and *List2* ($k = 390$, $AUC = 53.58\%$).
As one may notice, all the comparison involving the lists from *List3* to *List8* have Extended Kendall distances near or greater than 0.9

Table 19: The Extended Spearman coefficient values (gray cells) and the Extended Kendall distance values (white cells) resulting from the comparison of the nine annotation lists coming from the application of the SVD method to the Homo sapiens CC dataset, when varying the truncation level. The Extended Spearman coefficient values are in the gray cells, in the upper-right part of the table; the Extended Kendall distance values are in the white cells, in the lower-left part of the table. Intervals: 0 = maximum correlation; 1 = minimum correlation. All the lists are ordered from the one generating the maximum **ROC AUC percentage** (*List0*, $AUC = 83.49\%$) to the one generating the minimum AUC (*List8*, $AUC = 38.82\%$). We report the list amounts and parameters in Table 18.

		Extended Spearman values								
		List0	List1	List2	List3	List4	List5	List6	List7	List8
E.	List0		0.129	0.07	0.301	0.224	0.593	0.841	0.443	0.501
	List1	0.621		0.01	0.535	0.512	0.708	0.838	0.585	0.539
K	List2	0.491	0.558		0.324	0.245	0.656	0.872	0.499	0.509
e	List3	0.984	0.962	0.978		0.57	0.158	0.362	0.249	0.772
n	List4	0.868	0.894	0.913	0.935		0.416	0.302	0.306	0.885
d	List5	0.995	0.995	0.995	0.952	0.984		0.283	0.069	0.805
a	List6	0.999	0.999	0.999	0.988	0.998	0.936		0.427	0.628
l	List7	0.996	0.998	0.997	0.997	0.998	0.990	0.946		0.522
l	List8	0.998	0.998	0.999	0.999	0.999	0.996	0.984	0.982	

SVD truncation patterns We can notice something more interesting if we sort the lists on the basis of the truncation level, as made in Table 20.
While the Extended Spearman coefficients give no additional clue on specific trends, the Extended Kendall distances show that the higher the SVD truncation level difference between two lists gets, the higher is the Extended Kendall distance, and so the less similar are the lists.

Apart from the value of the comparison between *List0* and *List1*, all the other lists show value trends that increase when the distance between truncation levels increase. *List8*, for example, shows almost the maximum dissimilarity values with all the compared other lists. This is an interesting trend.

ROC AUC patterns Beyond its importance in inferring new knowledge about variation of the annotation lists, when varying the SVD truncation level, this item of information does not give any particular avail to our validation process.
Instead, interesting trends can be found by comparing the ROC AUC percentages and the Extended Kendall distance and Extended Spearman coefficient.
We searched for interesting trends between the ROC AUC and the our similarity coefficients, and we found them. As general indicators, we show the case for

Table 20: The Extended Spearman coefficient values (gray cells) and the Extended Kendall distance values (white cells) resulting from the comparison of the nine annotation lists coming from the application of the SVD method to the Homo sapiens CC dataset, when varying the truncation level. The Extended Spearman coefficient values are in the gray cells, in the upper-right part of the table; the Extended Kendall distance values are in the white cells, in the lower-left part of the table. Intervals: 0 = maximum correlation; 1 = minimum correlation. All the lists are ordered from the one produced with the greater **SVD truncation level** (*List1*, $k = 402$) to the one produced with the lowest level (*List8*, $k = 59$). We report the list amounts and parameters in Table 18.

		Extended Spearman values								
		List1	List0	List2	List4	List3	List5	List6	List7	List8
E.	List1		0.129	0.01	0.512	0.535	0.708	0.838	0.585	0.539
	List0	0.621		0.07	0.224	0.301	0.593	0.841	0.443	0.501
K	List2	0.558	0.491		0.245	0.324	0.656	0.872	0.499	0.509
e	List4	0.894	0.868	0.913		0.935	0.416	0.302	0.306	0.885
n	List3	0.962	0.984	0.978	0.57		0.158	0.362	0.249	0.772
d	List5	0.995	0.995	0.995	0.984	0.952		0.283	0.069	0.805
a	List6	0.999	0.999	0.999	0.998	0.988	0.936		0.427	0.628
l	List7	0.998	0.996	0.997	0.998	0.997	0.99	0.946		0.522
l	List8	0.998	0.998	0.999	0.999	0.999	0.996	0.984	0.982	

the Bos taurus MF dataset and the Gallus gallus MF dataset, in Table 21 and in Table 22.
As one may notice in this cases, lists generated by prediction that produced similar AUC (List0, List1, List2), have similar low Extended Spearman coefficients.
This means that lists from predictions having similar AUC percentages have element difference very low. Elements present in these lists have similar ranking, similar positions. This is the meaning of the Extended Spearman coefficients.

Since, as we saw, there is a correlation between ROC AUC and list similarity, this may be helpful in finding the best predicted annotations. In fact, our methods are able to produce ROC with maximum AUC, and the AUCs have an oscillatory trend. Since these oscillations produce low changes to the ROC AUC percentages, these will slightly influence the final results. Our methods, based on the optimization of the ROC AUCs, results quite robust. There is no need to find the overall best SVD truncation and the SIM cluster number., but is sufficient to find parameters that are near to the best ones.
The next software steps will anyway select the predicted annotations considered the best ones.
In conclusion, we can state that our Extended Spearman coefficient and Extended Kendall distance measures resulted very useful to understand the similarity (and dissimilarity) ratio between two lists. With this statistical tools, we can compute the level of "similarity" and "dissimilarity" between two lists, basing these concepts on the element position difference (Spearman) or on the number of elements having different orders (Kendall).
We see a relationship between the decrease of the truncation level difference and

Table 21: Amounts of AP and NAC annotations for Gallus gallus MF and Bos taurus MF when varying the SVD truncation level k. The best k is 378, and generates *List0*. The other truncation levels are those selected by our best truncation choice algorithm before finding the best. The likelihood threshold is fixed $\tau = 0.50$

SVD when varying truncation k				
	k	AP	NAC	ROC AUC
Gallus gallus, Molecular Function				
List0	**40**	9	39,340	75.38%
List1	**53**	5	39,344	74.33%
List2	**27**	10	39,339	73.69%
List3	**14**	11	39,338	67.25%
List4	**1**	164	39,185	35.98%
Bos taurus, Molecular Function				
List0	**70**	11	120,318	74.74%
List1	**93**	8	120,321	74.59%
List2	**47**	11	120,318	73.57%
List3	**24**	32	120,297	68.59%
List4	**1**	369	119,960	35.21%

Table 22: The Extended Spearman coefficient values (gray cells) and the Extended Kendall distance values (white cells) resulting from the comparison of the annotation lists coming from the application of the SVD method to the Gallus gallus MF dataset, when varying the truncation level. The Extended Spearman coefficient values are in the gray cells, in the upper-right part of the table; the Extended Kendall distance values are in the white cells, in the lower-left part of the table. Intervals: 0 = maximum correlation; 1 = minimum correlation. All the lists are ordered from the one produced with the greater SVD truncation level to the one produced with the lowest level. In **bold** we report the cases showing low values for lists having similar AUC.

		Extended Spearman values				
		list0	list1	list2	list3	list4
Gallus gallus, Molecular Function						
E	list0		**0.370**	**0.200**	0.620	0.85
x	list1	0.790		**0.330**	0.780	0.94
t.	list2	0.640	0.790		0.180	0.74
	list3	0.980	0.960	0.950		0.78
K.	list4	1.000	1.000	1.000	1.000	
Bos taurus, Molecular Function						
E	list0		**0.192**	**0.166**	0.364	0.794
x	list1	0.687		**0.279**	0.425	0.888
t.	list2	0.812	0.822		0.128	0.772
	list3	0.980	0.980	0.950		0.773
K.	list4	0.999	0.999	0.999	1.000	

the increase of similarity between two lists: the more the list truncation level divergence increase, the more dissimilar (in terms of bubble-sort swaps needed to make them identical) they are. For example, in Table 20 we can notice the

vertical white column values of List1, going from 0.621 (when comparing it to List0, having the highest truncation level) to 0.998 (when comparing it with List8, having the lower truncation level).

We also observe a relationship between the ROC AUCs and the similarity between two lists: the nearer two AUC percentages are, the more similar their output predicted annotation lists are. In this case the similarity is expressed through the Extended Kendall distance, that measures the difference between ranked position of an element present in both the analyzed lists.

As we stated before, these statistical tools can be very useful, in the validation pipeline, because they can state similarity levels between lists when varying the key parameters.

11 Biological relevance

As one can intuitively understand, not all the predictions are equally novel. If a prediction suggests the presence of more tree nodes, independent from the other nodes, we consider it *more novel* than another one predicting tree leaves or nodes linked to other nodes.

For example, for a given gene, we can run an annotation prediction X that suggests the existence of just 1 term that is a leaf into the term DAG tree, made by 100 nodes.
Then, we can run another prediction Y that suggests the existence of an entire tree branch, made by 10 terms, and completely independent from the other tree branches, into the DAG tree made only by 15 nodes.
One can intuitively comprehend that the Y prediction is decisively *more novel* than the X prediction, because predicts the presence of new Gene Ontology sub-areas to explore.

Defining the concept of *novelty* is, in this case, a key point. There are multiple elements to consider when analyzing the predicted terms of a gene:

1. The percentage of nodes suggested by the prediction. For example, if a prediction process suggests a number of nodes that is just the 1% of the previously present nodes in the DAG tree, it is less interesting than a prediction that suggests a number of nodes equal to the 75% of the already present node numbers.

2. Are the predicted terms mainly leaves or internal nodes in the DAG tree? A leaf means just a more details specialization into the ontology sub-area, while an internal node adds more newness.

3. Do the suggested terms form an independent branch in the term DAG tree? If the predicted terms are linked just with the DAG tree root (Cellular Component, Molecular Function, or Biological Process), and not linked with any other node, that means that our prediction suggests the involvement of the analyzed gene into a new ontology sub-area, that may lead to important and novel scientific research directions.

To state the novelty of a prediction made on a gene, we developed two tools: a visual DAG tree functionality, named DagViewer, and a term semantic similarity measure, the *novelty indicator*, based on Schlicker measure [18].

11.1 DAG tree viewer

The DAG viewer tool is an useful instrument that allows the user and the biologist to understand the importanc of a prediction on a gene through visualization. This software modules behaves as follows:

1. Once the user select the gene whose term DAG tree he/she wants to see, it takes the list of the AP of that gene.

2. It seeks for the annotations already present in the analyzed database.

3. It seeks in the database all the relationships child-parent between terms of the already present annotations.

4. It depicts the DAG tree of the already present terms where each already present term is a black-circled box.

5. It seeks in the database all the relationships child-parent between terms of the new predicted annotations.

6. It depicts the DAG tree of the predicted terms where each term is a blue-hexagon box.

We provide some example of the use of this visualization tool: the Molecular Function term prediction DAG tree made using our SVD method for the Homo sapiens gene RARA in Fig. 27; the Cellular Component term prediction DAG tree made using our SVD method for the Homo sapiens gene DPM1 in Fig. 26; and the Biological Process term prediction DAG tree made using our SVD method for the Homo sapiens gene PPME1 in Fig. 25
In Fig. 25 one can notice that the prediction suggests the existence of a new complete branch, where some nodes are linked to already present nodes. This graph suggests to the biologist to further explore the Biological Process area of organelle features, chromosome features, and histone features, that may be already tied with existing features of the gene PPME1. Even though these predicted terms are not independent from the others, their amount and their height in the tree make this prediction quite interesting and important.
In Fig. 26 one can notice that the prediction suggests the existence of a single leaf node, at the bottom of the tree. This graph represents a prediction of minimum interest, because it suggests the existence of only a super-specialized term in the term feature list.
In Fig. 27 one can notice that the prediction suggests the existence of a complete independent separate branch in the tree. All the predicted terms are not linked with previously present terms, except to the tree root. This prediction represents a case of maximum importance and relevance, because suggests to the biologist the existence of a never-explored sub-area of the molecular function domain to be connected with the gene RARA.

11.2 Novelty indicator

In addition to the visualization tool, we also introduce a statistical indicator of the *importance* of a term prediction made on a gene.

All the importance components that we described before are considered by the $\text{GOscore}_{\text{BM}}$ term semantic similarity measure, introduced by Schlicker et al. [105] in the FunSimMat database.

Schlicker and colleagues define this coefficient as follows. Given two genes q and p, we denote with GO^q the set of all the terms annotated to the gene q, and with GO^p the set of all the terms annotated to the gene p.

The general idea of this measure is to build a matrix where to insert the binary semantic similarity measure between each term of the first set and each element

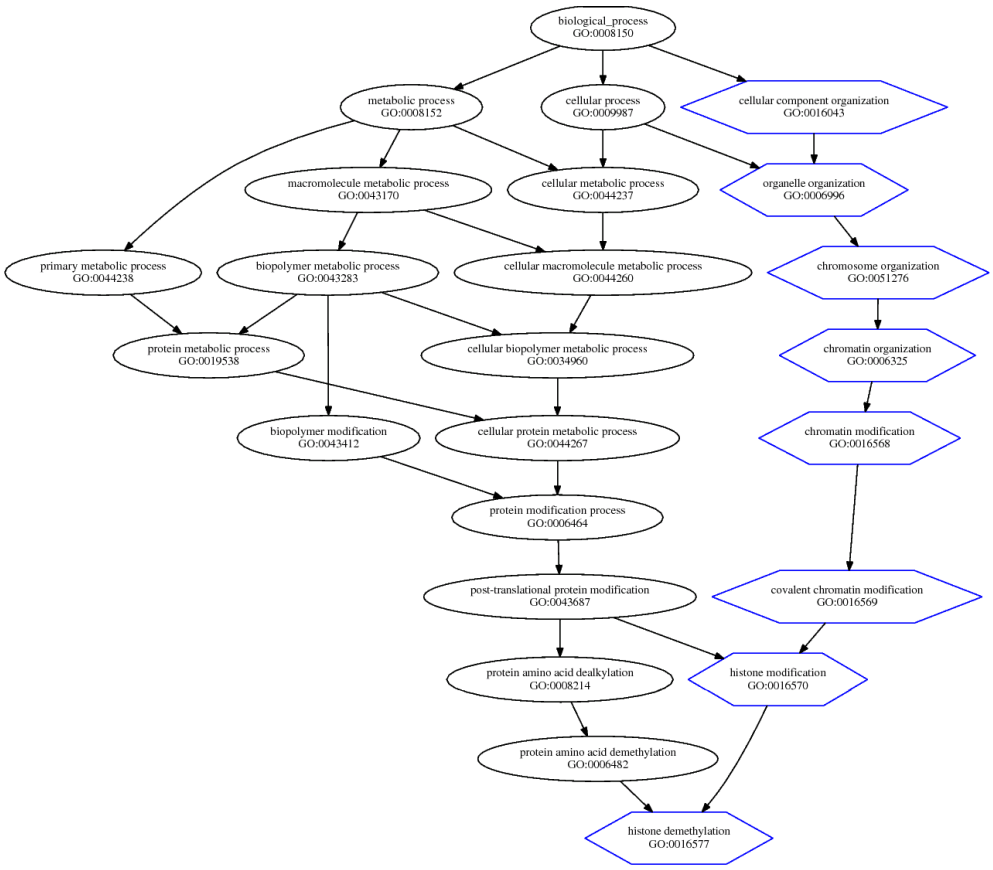

Figure 25: DAG tree of the Biological Process terms predicted for the Homo sapiens gene PPME1 (Entrez Gene ID: 51400) with the SVD method with our optimized parameters. The blue hexagon elements are the predicted terms, while the black circle elements are the terms already present in the database before the prediction.

of the second set. Then, we consider the maxima of each row and each column, and we compute the average of them.

We build the S matrix as follows: with $i = 1..N$ rows and $j = 1...M$ columns, each element s_{ij} is made computing the Resnik similarity measure (as defined in Section 5.4) between the i_{th} element of GO^q and the j_{th} element of GO^p.

$$\begin{aligned} &\forall i \in \{1, ..., N\},\ \forall j \in \{1, ..., M\} : \\ &s_{ij} = ResnikSimilarity(\, GO_i^p,\, GO_j^q \,) \end{aligned} \tag{27}$$

Within this $S \in \mathbb{R}^{M \times N}$ matrix, we define two other operators. rowScore is the average of the row maxima, and columnScore is the average of the column

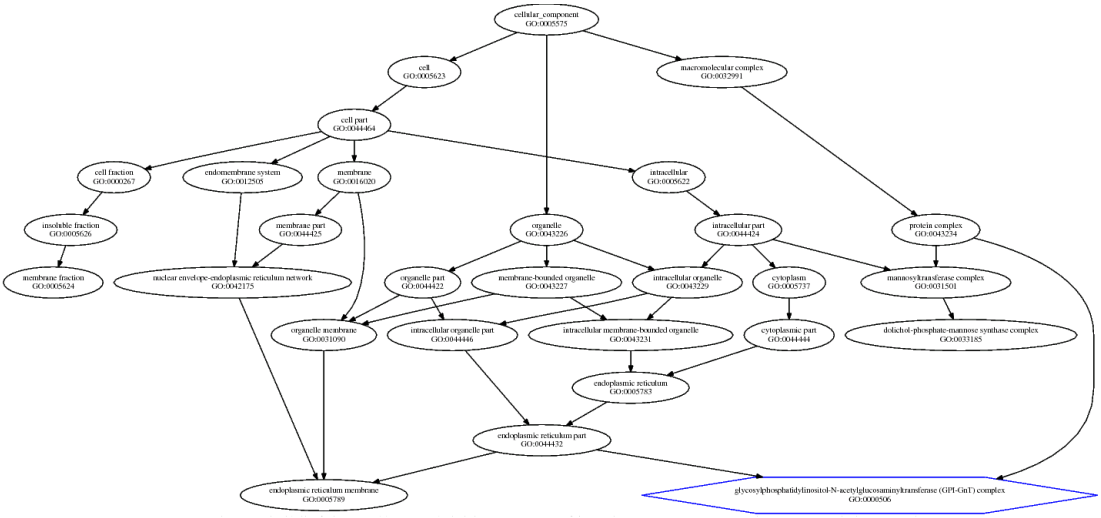

Figure 26: DAG tree of the Cellular Component terms predicted for the Homo sapiens gene DPM1 (Entrez Gene ID: 8813) with the SVD method with our optimized parameters. The blue hexagon elements are the predicted terms, while the black circle elements are the terms already present in the database before the prediction.

masima. s_i is the i_{th} row, and s_j is the j_{th} column:

$$\begin{aligned} \forall i \in \{1,...,N\}, \forall j \in \{1,...,M\} : \\ maxRowScore_i = max(s_{ij}) \\ \forall i \in \{1,...,N\} : rowMaximaSum = \sum maxRowScore_i \end{aligned}$$

$$\begin{aligned} \forall j \in \{1,...,M\}, \forall i \in \{1,...,N\} : \\ maxColumnScore_j = max(s_{ij}) \\ \forall j \in \{1,...,M\} : columnMaximaSum = \sum maxColumnScore_j \end{aligned}$$

$$\begin{aligned} rowScore &= rowMaximaSum \ / \ N \\ columnScore &= columnMaximaSum \ / \ M \end{aligned} \tag{28}$$

The Schlicker measure is defined as:

$$GOscore_{BM}(p,q) = max(rowScore(p,q), columnScore(p,q)) \tag{29}$$

We use this coefficient to measure the level of *newness* of the prediction. For a given gene G, we compare the set of terms associated to it before the prediction (G_{before}) with the set of terms associated to it by the prediction (G_{after}).
The more the two sets are different, the lower the $GOscore_{BM}$ is. If the two sets had no common elements (that is impossible in our case, because we use terms from an ontology), the $GOscore_{BM}$ would be zero.

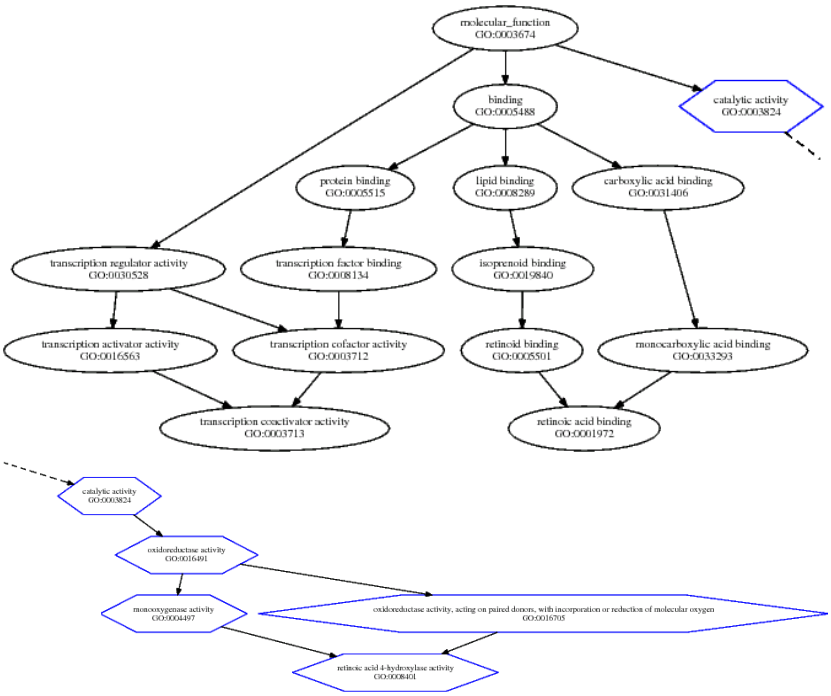

Figure 27: DAG tree of the Molecular Function terms predicted for the Homo sapiens gene RARA (Entrez Gene ID: 5914) with the SVD method with our optimized parameters. For lack of paper room, we have to split the figure in two parts: the lower part is the right branch of the upper part. The blue hexagon elements are the predicted terms, while the black circle elements are the terms already present in the database before the prediction.

Since we use the Resnik semantic similarity measure as central coefficient [14], there is not upper bound.

11.3 Results and discussion

We tested this $\text{GOscore}_{\text{BM}}$ measure on all the prediction made for the Homo sapiens annotations with the methods SVD-us, SIM1 and SIM2 that lead to the results descibed in Table 8.

The $\text{GOscore}_{\text{BM}}$ measure resulted quite proportional to the visualization analysis. Gene predictions with more (independent) blue hexagon nodes have lower $\text{GOscore}_{\text{BM}}$ coefficients.

We report here the list of the five gene term predictions having the lowest $\text{GOscore}_{\text{BM}}$ measures, for all the experiments on the Homo sapiens datasets, in Table 23.
The tree graph for gene P2RY14 in Fig. 28, having the second lowest Schlicker measure ($GOscore_{\text{BM}} = 0.087$), represents one of the most important and interesting term prediction tree produced by our methods. As one can notice in in Fig. , the predicted terms (blue hexagons) form a complete new independent branch of the tree.

Table 23: The five genes having best Schlicker measures for their term predictions, produced by the SVD, SIM1, SIM2 methods applied to the Homo sapiens and GO datasets. SVD-us stands for SVD method with our optimized parameters. CC: Cellular Component; MF: Molecular Function; BP: Biological process. $\#pt$: number of predicted terms for that gene.

method	ontology	gene symbol	#pt	$\mathbf{GOscore_{BM}}$	graph
SVD-us	MF	RARA	5	**0.074**	Fig. 27
SVD-us	MF	P2RY14	18	**0.087**	Fig. 28
SIM1	BP	MMP23B	1	**0.087**	Fig. 29
SIM1	MF	CCR2	2	**0.134**	Fig. 30
SIM2	MF	NT5M	1	**0.154**	Fig. 31

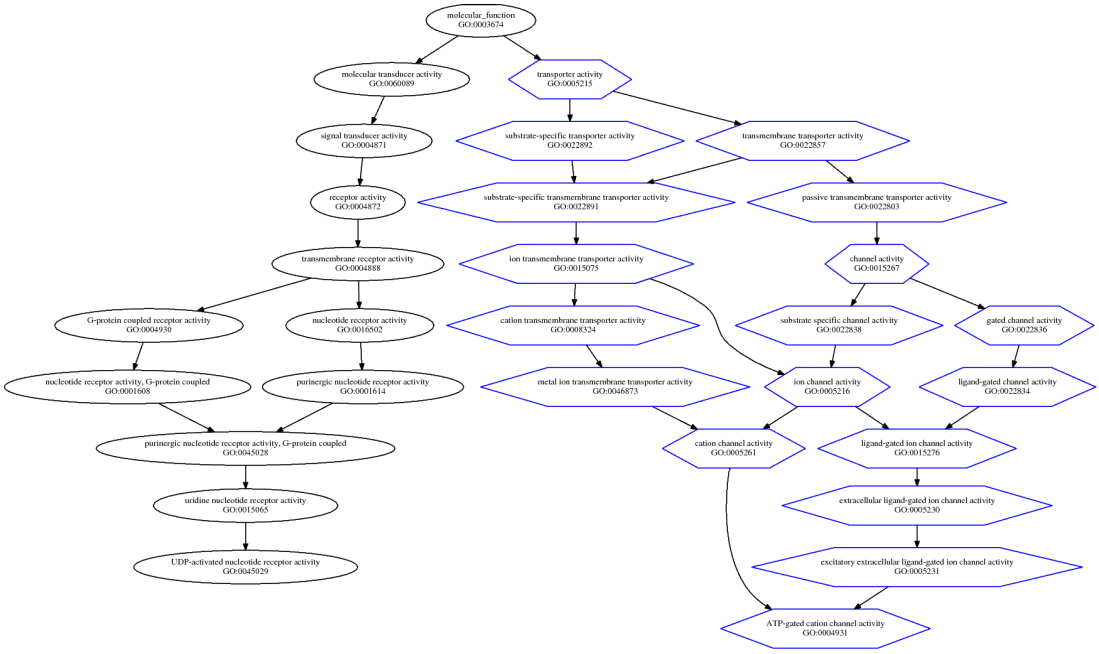

Figure 28: DAG tree of the Molecular Function terms predicted for the Homo sapiens gene P2RY14 (Entrez Gene ID: 9934) with the SVD method with our optimized parameters. $GOscore_{\mathrm{BM}} = 0.087$

All the new blue terms are connected only to the root of the tree, the Molecular Function node.
In addition, the number of predicted terms is also very high: 18 new terms, when the already present terms (black circle nodes) are just 12. This means +150% terms more.

These three factors (a lot of predicted terms, linked only to the root, that form an independent branch) make the Schicker measure $\mathrm{GOscore_{BM}}$ very close to zero, meaning a very *novelty important* prediction.

This prediction is second most important prediction made by our methods. The first is the prediction made for the RARA gene, depicted in Fig. 27, where the situation is similar: a complete independent branch of 5 new terms lead to the $\text{GOscore}_{\text{BM}}$ equal to 0.074. Five new terms and 14 "old" terms means a +35.71% increase

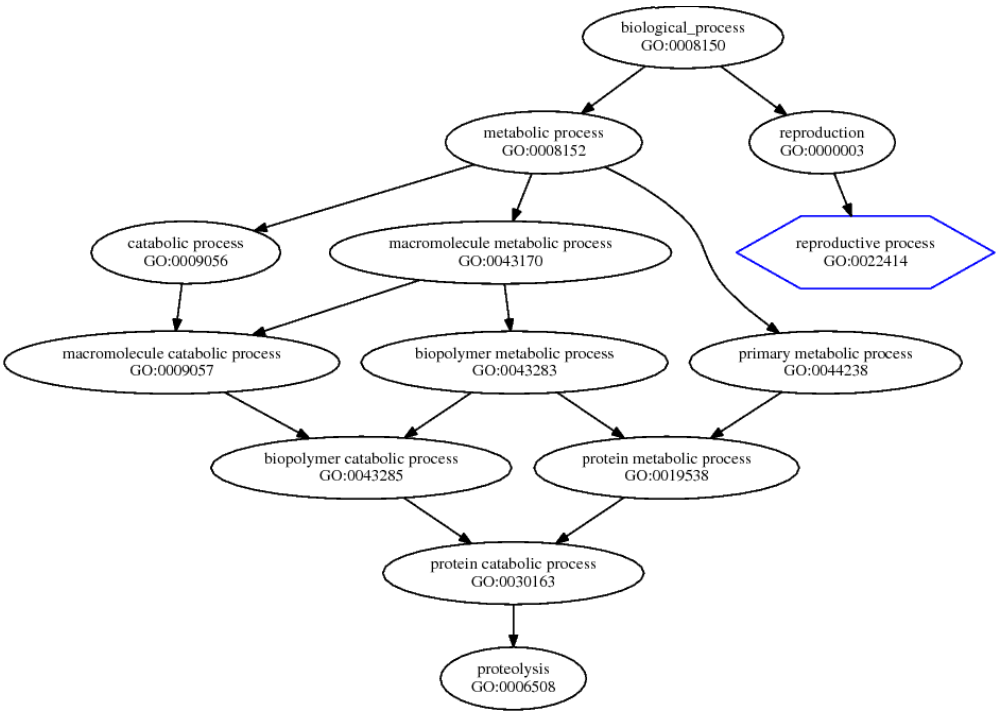

Figure 29: DAG tree of the Biological Process terms predicted for the Homo sapiens gene MMP23B (Entrez Gene ID: 8510) with the SIM1 method with our optimized parameters. The blue hexagon elements are the predicted terms, while the black circle elements are the terms already present in the database before the prediction. $GOscore_{\text{BM}} = 0.087$

In Fig. 29 we see the depiction of the graph tree of the predicted terms for the gene MMP23B. Also in this case we have a very low Schicker measure ($GOscore_{\text{BM}} = 0.087$), mainly due to the position of the new predicted term. In fact, the only predicted term "reproductive process" (blue hexagon) is just two-links far from the root, and it is independent from all the other nodes. In this deep tree, the high level of the predicted term leads it to a low Schlicker value.

We can analyze the term graph tree of the gene CCR2 in Fig. 30. In this case, the low Schlicker measure ($GOscore_{\text{BM}} = 0.134$) is due to multiple factors: the ratio between the new predicted terms and the already present nodes (2 new nodes on 8, meaning +25% terms), and the height of the first predicted term "citokine activity".

The other predicted node, "chemokine activity", is not independent from the other nodes: it is linked to "chemokine receptor binding", on another tree branch. This makes the Schlicker measure of this prediction greater than the

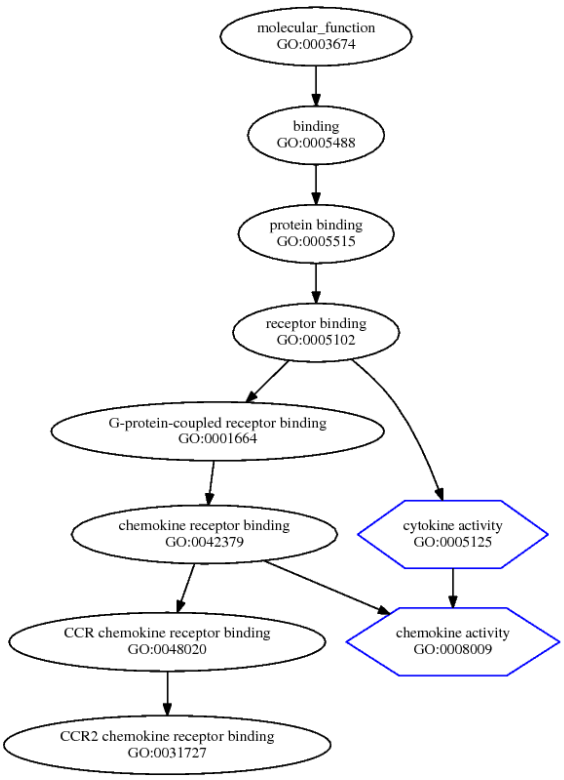

Figure 30: DAG tree of the Molecular Function terms predicted for the Homo sapiens gene CCR2 (Entrez Gene ID: 729230) with the SIM1 method with our optimized parameters. The blue hexagon elements are the predicted terms, while the black circle elements are the terms already present in the database before the prediction. $GOscore_{\mathrm{BM}} = 0.134$

previous ones, that contained independent terms.
In the last of the five images, we see the graph tree of the predicted terms for the gene NT5M (Fig. 31). The predicted blue term is a leaf in the tree, connected only with the previous leaf in the only tree branch existing. Its Schlicker measure is greater than the previous ones, but anyway the 5th among all the predictions: its low value is mainly due to the low number of elements of this tree (just 7). The addition of this predicted term represents +14.29% more terms to the tree.

The novelty indicator and the prediction We also can take advantage of the DagViewer novelty indicator tool to enhance the reliability of our prediction. We can depict in the DAG tree images the nodes-terms found within the annotation set in the updated database version, and then use this item of information as indicator of the (right) direction taken by our prediction, if present.
For example, in Fig. 32, we can see the term DAG tree for the gene ROP1N1B, predicted by our SVD method. We have inserted a green rectangle inside each term that is part of an annotation found in the updated database. As one may intuitively understand, this illustration states that the term-leaf of this tree may be a likely correct annotation.

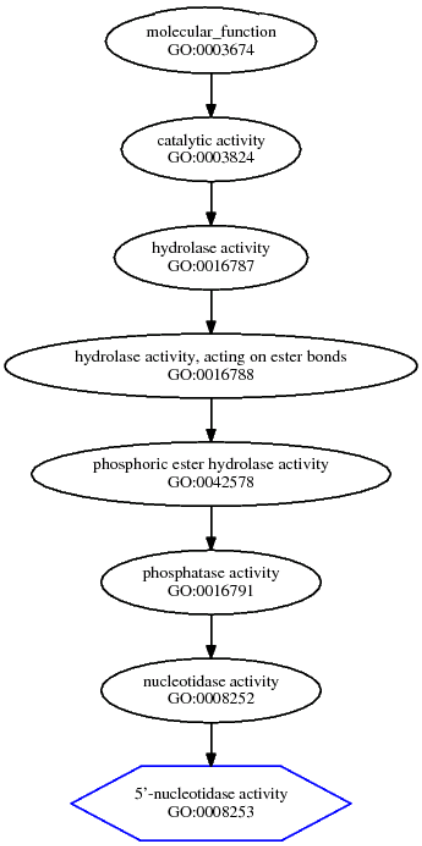

Figure 31: DAG tree of the Molecular Function terms predicted for the Homo sapiens gene NT5M (Entrez Gene ID: 56953) with the SIM2 method with our optimized parameters, and the Resnik measure. The blue hexagon elements are the predicted terms, while the black circle elements are the terms already present in the database before the prediction. $GOscore_{\mathrm{BM}} = 0.154$

Since the annotation-leaf <ROPN1B, *microtubule based flagellum*> has a parent already present in the analyzed database (<ROPN1B, *flagellum*>), and a parent predicted by our method and found in the updated database version (<ROPN1B, *intracellular membrane-bounded organelle*>), we consider it probably correct.
Having a parent already present in the analyzed database version means a strong link with the available knowledge, while having a parent predicted and found confirmed in the updated database means that our prediction goes in the right direction. These two conditions suggest us to trust the correctness of the annotation-leaf.

Unfortunately, the predicted annotation DAG trees having this two conditions are not many. We report the list of genes having a prediction with these conditions in the *predVal* column of the next Table 26.
We can also take advantage of this information to rank the terms in order of distance from the nearest validated predicted term. For example, in Fig. 32,

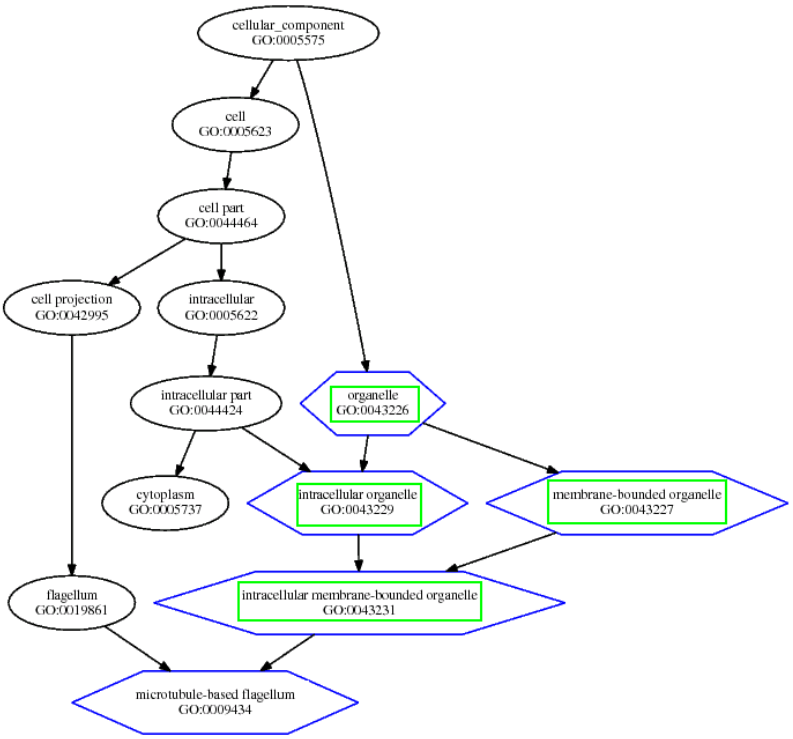

Figure 32: DAG tree of the Molecular Function terms predicted for the Homo sapiens gene ROPN1B (Entrez Gene ID: 152015) with the SVD method with our optimized parameters. The blue hexagon elements are the predicted terms, the green rectangle elements are found confirmed in the updated database version, and the black circle elements are the terms already present in the database before the prediction. $GOscore_{\mathrm{BM}} = 0.335$

the term-leaf *microtubule based flagellum* has distance one (edge) from the first predicted validated term-node.
Another interesting quantity is the number of predicted annotations found confirmed in the updated database for a gene. For example, in Fig. 32, we have 4 annotations over 5 (that is the 80%) that were found confirmed in the updated database version. We consider more reliable the predictions that find a number of predicted terms in the updated database higher than the 50%. This item of information is reported in the "*conf.* $\geq 50\%$ " column of the next Table 26.

Table 24: Final list of annotations predicted by our methods, not found in the updated version of the database neither in the literature o through the knowledge web tools. The "likelihood ranking " columns report the predicted annotation list ranking percentile in which the predicted annotation befalls by the methods SVD with our optimized parameters, SIM1, SIM2. For example, the annotation <CHST14, *chondroitin sulfate proteoglycan biosynthetic process*> resulted ranked 36th on the 64 AP predicted by the methods. This position befalls into the first 56.25% percentile. The "confirmed %" columns report the number of predicted terms found confirmed in the updated database for that gene. For example, the methods predicted other terms for the gene CHST14, beyond the term *chondroitin sulfate proteoglycan biosynthetic process*. Among these predicted terms, we found the 58.33% of them in the updated database version. The last "val. edge distance" columns report the distance of the term from the nearest predicted and validated term. For example, the term *microtubule-based flagellum* predicted for the gene ROPN1B, as shown in Figure 32, is linked to the predicted and validated term *intracellular membrane-bounded organelle* through a single link, that leads to the value 1 in the ROPN1B last columns. In the term name column, the codes [CC], [MF], [BF] states if the term belong to the Cellular Component [CC], to the Molecular Function [MF], or to the Biological Process [BP] sub-ontology.

gene	term	SVD	SIM1	SIM2	SVD	SIM1	SIM2	SVD	SIM1	SIM2
		likelihood ranking %			confirmed %			val. edge distance		
CHST14	Chondroitin sulfate proteoglycan biosynthetic process. [BP] (GO:0050650)	56.25	34.29	57.14	58.33	58.33	58.33	1	1	1
CHST14	Biopolymer biosynthetic process. [BP] (GO:0043284)	56.25	34.29	57.14	58.33	58.33	58.33	1	1	1
ROPN1B	Microtubule -based flagellum. [CC] (GO:0009434)	50.00	50.00	50.00	80.00	80.00	80.00	1	1	1
PPME1	Organelle organization. [BP] (GO:0006996)	3.13	5.71	7.14	12.50	12.50	12.50	1	1	1
CHST14	Dermatan sulfate proteoglycan biosynthetic process. [BP] (GO:0050651)	56.25	34.29	57.14	58.33	58.33	58.33	2	2	2
PPME1	Chromosome organization. [BP] (GO:0051276)	3.13	5.71	7.14	12.50	12.50	12.50	2	2	2
CPA2	Proteolysis involved in cellular protein catabolic process. [BP] (GO:0051603)		31.43	50.00		50.00	50.00		1	1
CNOT2	Positive regulation of cellular metabolic process. [BP] (GO:0031325)	28.13			12.50			1	1	1

In conclusion, we can consider three main indexes to rank the quality of our predictions:

- The likelihood score ranking percentile. Given a predicted annotation, we take the its position in the prediction ranking, and we state which percentile it befalls. Obviously, best predictions lead to lower percentages, that augment the likelihood of the annotations to be correct. We report these percentages in the "likelihood ranking %" columns of Table 24.
- The number of terms found in the annotation dataset of the updated database version, on the total of the predicted terms for that gene. Given the list of the predicted terms for a certain gene, we take the number of terms found in the updated database, and we compute the percentage on the number of total predicted terms for that gene. This item of information is very useful to us because it states if the prediction goes in the right direction. An annotation predicted with no tree terms found in the updated database may be correct as well, but we do not have any confirmation about the prediction quality. We report these percentages in the "confirmed %" columns of Table 24.
- The validated edge distance. Given a predicted term (non validated in the updated database), we search in the DAG tree how many edges sever this term-node from the nearest predicted term validated in the updated database. The lower is this link number, the more likely the predicted term is correct. Obviously, this indicator is influenced by the structure of the ontology, and may penalize the terms that have few ancestors.

An excellent feature of the procedure that computes these quantities the automation: all the conditions reported in the previous list are computed through a computerized pipeline of operations. Once the user understands how to manage the system, he/she is able to run all the procedure steps and get a final list of predicted annotations as output, as the one in Table 24.
Among the previously described correctness conditions, if an updated database version were not available, just the second one (percentages of terms found in the updated dataset) would become unreachable. The third one (edge distance from the first validated node) would be easily replaceable with the edge distance from the first node already present in the analyzed database. This way, an efficient validation that takes advantage only of the analyzed database version would be set.

Table 25: Comparison between the "validated edge distance" indexes for the three methods. The column predVal states how many terms have a parent predicted and validated; the AP CC (MF) (BP) is the number of predicted annotations for the CC (MF) (BP) dataset; predVal% states the percentage of predVal terms on the total of predicted terms.

method	predVal	AP CC	AP MF	AP BP	tot	predVal%
SVD-us	7	8	81	64	153	4.61%
SIM1	5	8	13	35	56	8.92%
SIM2	5	8	30	14	52	**9.62%**

In Table 25, we can notice that the percentage of terms having "validated edge distance" equal to 1 is the highest for the SIM2 method, and that SIM1 outperforms SVD-us.
This gives us an additional clue on the quality of our new methods with respect to the classical Truncated SVD method.
Also with this lastly introduced prediction correctness index ("validated edge distance"), we can see an improvement in the results of our new methods SIM1 and SIM2 with respect to SVD.
By considering always the predictions made for the Homo sapiens single subontology in Table 8, we counted how many predicted terms have the "validated edge distance" equal to 1, that is the number of terms that have at least a parent that is predicted and found confirmed in the updated database. We then generated a final table of predicted annotations that satisfy these conditions in Table 26.

11.4 Final predicted annotations

On the basis of the prediction algorithms, validation methods, similarity measures and novelty indicator described in the previous sections, we can possibly state a list of predicted annotations that we consider the most likelihood, between the ones we suggested.
In the Introduction we promised a list of predicted annotations that would be usable by biologists and physicians to address their experiments, and here we report this directory for the Homo sapiens annotations.
We report this list in Table 26, that recalls the quantities of Table 24 with the proper quality thresholds.

We inserted in this list the predicted annotations, not found in the updated version of the database, that respect these conditions:

- Predicted by one (or more) our methods (SVD, SIM1, SIM2), applied to the Homo sapiens annotation datset of single sub-ontologies, described in Section 9, that means predicted by many different algorithms;
- Ranked within the first 10% positions in the output AP list (column "first 1%" in Table 26), that means being one of the most likely predicted annotation;
- Predicted by one (or more) our methods (SVD, SIM1, SIM2) generating a ROC AUC greater than our fixed threshold $\omega = 2/3 = 66.67\%$ (this condition is true for all the genes and therefore not reported in the table);
- Found within a prediction DAG tree having the novelty indicator value very low (column "$\text{GOscore}_{\text{BM}} < 1$" in Table 26), that indicates a very novel prediction for that gene;
- Having, among all the gene predicted annotations, the percentage of them found confirmed in the updated database is greater than 50% ("conf. $\geq$ 50%" column in Table 26);
- Having an annotation term parent already present in the database, and an annotation parent predicted and found confirmed in the updated database ("predVal" column in Table 26).

The predicted annotation list reported in Table 26 is the final biological relevance we can provide to physicians and biologists to address their experiments about human genes. A concrete application of our methods, that we hope may improve and quicken the discovery of new cures, new therapies, and new knowledge about gene functions.
The annotations reported in Table 26 are sorted by number of conditions satisfied, from the ones that respect more conditions, to the ones that respect less conditions.

By observing this Table 26, we can notice that two annotations predicted for the gene CHST14 (<CHST14, *chondroitin sulfate proteoglycan biosynthetic process*> and <CHST14, *byopolimer biosynthetic process*>) are suggested by all the thee methods (SVD-SIM1-SIM2), are part of a prediction where more than a half of the predicted terms were found in the updated database, and have at

Table 26: Final list of annotations predicted by our methods, not found in the updated version of the database neither in the literature or through the knowledge web tools. The *predicted by SVD-SIM1-SIM2* states which method(s) predicted that annotation; the *first* 10% column states if its position in the likelihood ranking is in the first 10% of all the predicted annotations (APs), for all the three prediction methdos; the "*conf.* $\geq$ 50%" column states if, among all the predicted annotation terms for the considered genes, the percentage of them found confirmed in the updated database is greater than 50%; the *predVal* column states if the predicted annotation term node has all the parent nodes present in the database and at least one parent predicted and found confirmed in the new updated database version. In the term name column, the codes [CC], [MF], [BF] states if the term belong to the Cellular Component [CC], to the Molecular Function [MF], or to the Biological Process [BP] sub-ontology.

gene symbol	term	predicted by			first	conf.	pred
		SVD	SIM1	SIM2	10%	$\geq$ 50%	Val
PPME1	Organelle organization. [BP] (GO:0006996)	√	√	√	√		√
CHST14	Chondroitin sulfate proteoglycan biosynthetic process. [BP] (GO:0050651)	√	√	√		√	√
CHST14	Biopolymer biosynthetic process. [BP] (GO:0043284)	√	√	√		√	√
ROPN1B	Microtubule-based flagellum. [CC] (GO:0009434)	√	√	√			√
CHST14	Dermatan sulfate proteoglycan biosynthetic process. [BP] (GO:005051)	√	√	√		√	
CPA2	Proteolysis involved in cellular protein catabolic process. [BP] (GO:0051603)		√	√		√	√
PPME1	Chromosome organization. [BP] (GO:0051276)	√	√		√		
CNOT2	Positive regulation of cellular metabolic process. [BP] (GO:0031325)	√					√

least a parent predicted and found in the updated database. Anyway, their prediction likelihood score is not part of the first 10% of the predicted annotations by all the three methods.
Two other annotations, <ROPN1B, *microtubule-based flagellum*> and <PPME1,

organelle organization>, were predicted by all our three methods and do not have a score likelihood ranked in the first 10% of all the predicted annotations. The first also has at least a parent predicted and found in the updated database.
The annotation <CHST14, *dermatan sulfate proteoglycan biosynthetic process*> was predicted by the three methods, and is part of a prediction in which more than half of the predicted terms were found in the updated database.
The annotation <CPA2, *proteolysis involved in cellular protein catabolic process*> was predicted only by SIM1 and SIM2 methods, but is part of a prediction in which more than half of the predicted terms were found in the updated database, and have at least a parent predicted and validated by the the updated database.
The annotation <CNOT2, *positive regulation of cellular metabolic process*> was predicted only by SVD, but its likelihood score is part of the first 10% among the predicted annotations, and has at least a parent predicted and validated by the the updated database. Finally, in this list of the most likely annotations, the gene function <PPME1, *chromosome organization*> was predicted by SVD and SIM1 methods, and its likelihood score is ranked within the first 10% among the predicted annotations.
Obviously, other annotations were predicted, but no one of them satisfied at least three of the six conditions reported in this Table 26.

Biological interpretation Once we produced the final list of our best predicted annotations, shown in Table 26, we searched for any biological evidence proofing the contingent correctness of these gene functions. Obviously, the annotations listed in the table were not found present in the updated database version, neither in the literature or through the web tools.
So, we turned to a biologist and searched on the internet for any possible available knowledge about our predicted gene functions.

For the predicted annotation <PPME1, *chromosome organization*> (DAG tree shown in Fig. 25), we searched for the gene PPME1 on the PubMed [69] website. We found in the gene webpage, the following summary: "This gene encodes a protein phosphatase methylesterase localized to the nucleus, influencing the *chromosome organization*, through the dephosphorylation of regulatory proteins of the ERK pathway".
This sentence clearly states the existence of this annotations, although it is not added to the GPDW data warehouse, or to AmiGo or GeneCards web tools.
Since the annotation <PPME1, *organelle organization*> (shown in Fig. 25) is a parent of the previous one, it consequently results correct, for the ontology structure.

We then checked the existence of the annotation <CHST14, *dermatan sulfate proteoglycan biosynthetic process*>. We searched for the CHST14 gene proteins on UniProt [109], where we found its function: "Catalyzes the transfer of sulfate to position 4 of the N-acetylgalactosamine (GalNAc) residue of dermatan sulfate. Plays a pivotal role in the formation of 4-0-sulfated IdoA blocks in dermatan sulfate.". From this we have information about the involvement in the dermatan sulfate, but no information about the involvement in the proteoglycan.

So, we scrutinized the concept of "dermatan sulfate" and found that it is a glycosaminoglycan. On the English Wikipedia, we found a link to a paper containing what we were searching: "Based on core disaccharide structures, GAGS are classified into four groups. Heparin/heparan sulfate (HSGAGs) and chondroitin/dermatan sulfate (CSGAGs) are synthesized in the golgi apparatus, where protein cores made in the rough endoplasmic reticulum are posttranslationally modified with O-linked glycosylations by glycosyltransferases forming a proteoglycan" (Sasisekharan et al. [110]).
This paper states the existence of the annotation <CHST14, *dermatan sulfate proteoglycan biosynthetic process*>, and consequently the existence of its parent <CHST14, *chondroitin sulfate proteoglycan biosynthetic process*>, and its ancestor <CHST14, *biopolymer biosynthetic process*>.
From this discovery, we can state that the protein CHST14 is involved not only in the chondroitin and dermatan sulfate, but also in their proteoglycan biosynthetic process.

For the annotation <ROPN1B, *microtubule-based flagellum*> (DAG tree shown in Fig. 32), we searched for the gene ROPN1B proteins on UniProt ([109]). On its page, we found the sentence: "Note: In the sperm tail, found in the principal piece and in the cytoplasmic droplet located at the distal end of the midpiece. Inner surface of the fibrous sheath".
We know that the human sperm tail is a microtubule-based flagellum, and this states the existence of this annotation. From this discovery, we can state that the ROPN1B protein is involved only in eukaryotic organisms, because the prokaryotic does not have microtubule-based flagelli.
For the annotation <CPA2, *proteolysis involved in cellular protein catabolic process*>, we found that the associated term and the term "vacuolar protein catabolic process" share a common ancestor ("proteolysis"). This term "vacuolar protein catabolic process" is more specific than the term of our annotation, and since the annotation <CPA2, *vacuolar protein catabolic process*> is already present in the data banks, our annotation results correct, although not yet inserted in the available resources.
A similar situation goes to the annotation <CNOT2, *positive regulation of cellular metabolic process*>. On UniProt [109], we found that the CNOT2 protein are related to the "positive regulation of cytoplasmic mRNA processing body assembly" term, although this term has not been added to the Gene Ontology nor to the GPDW, yet. This term is more specific than the one in our annotation, so our gene function results correct.

In conclusion, we found scientific evidence for the existence of all the eight most likely predicted annotations listed in Table 26, showing the reliability of our prediction methods.

12 Conclusions

Currently, the discovery of new biomolecular annotations is a time-consuming process, in biology. Biological experiments are long-lasting and expensive, and so are the *in vitro* validation procedures of biomolecular annotations that finally state if an annotation is biologically correct and reliable.
In this thesis, we described some computational methods that we explored to predict new gene functions, to verify their consistence with multiple validation procedures, to compare different annotation lists, and to comprehend the value of our predictions.

We started from an existing algorithm by Khatri et al. [54], and first enhanced it with automated procedures able to select the best key parameters for the SVD truncation level and for the likelihood threshold. Departing from a previous work developed in our research group [8], we moved to new algorithms, based on gene clustering and on term-term similarity weights.
Our tests showed that these methods are able to improve the reconstruction of the output annotation matrix. This led our prediction to increase the percentage of annotations later found confirmed by the annotations become subsequently available within updated versions of the public data bases, while decreasing the number of total predictions.

Together with these new techniques, we also provided some useful validation procedures. Since we do not have any facility to run *in vitro* experiments, we designed and implemented different validation methods based on ROC analysis, comparison with available versions of annotation databases, and literature and analysis.
Our validation procedures resulted very efficient in checking the correctness of our predicted annotations.

To better compare annotation lists, when varying the predictive methods or the input parameters, we introduced new coefficients, based on Spearman rank and on Kendall measure, which are able to detect *how much* a list differs from another one.
These new measures demonstrated very effective and able to reach the objectives which they were designed for, showing similarity values that vary proportionally to the difference between the lists.

Finally, we designed and developed software functionalities to analyze the *importance* level of our predictions. First, with a powerful DAG viewer visualization tool, able to depict the DAG tree of the (former and predicted) annotation terms of each gene. Afterwards, by implementing a *novelty indicator* measure based on the semantic similarity between the original gene and the new annotated gene, fit to classify each prediction from *little novel* to *very novel*.
The DagViewer with its DAG graphs that demonstrated to be very useful for understanding which predictions more relevant and interesting than others, can be easily interpreted also by non-expert readers.
The introduced novelty indicator values resulted consistent with the visualization tool results, and "translated" the concept of *prediction novelty* from qualitative visualizations to quantitative values.

After designing these modules, we integrated them all into a unique software, that may be extended in the future with new functionalities, too.

Beyond the general positive overview, after the test result analysis, we noticed some issues that may be solved and/or improved in the future.
For example, we can improve the choice of some heuristically fixed parameters. We decided to consider only the reconstructions leading to a ROC AUC lower than 100% and greater than $\omega = 2/3 = 66.67\%$, but this threshold was heuristically fixed by us. A new data-driven threshold value could be define in the future.

Another point to study better is the number of annotations to consider as APs. In this work, we considered the predicted annotations with likelihood greater than the τ threshold, where the chosen τ is the one that minimizes the sum of presumed errors, AP + AR.
An alternative to this is choosing the first best $B = 100$ (or 1,000 or 10,000 etc) annotations for every prediction, and then use them to compare all the AP list. We did not choose this way because the first B annotations of every reconstructed output matrix vary on the basis of the matrix dimensions, and so this comparison may lead to non-sense results.
To avoid this problem, another option may be to choose the first $b = 10\%$ (or 15%, or 20%, etc.) predicted annotations to create the AP list. This choice is better than the previous because is related on the matrix dimension, but anyway contains some issues. For example, it may exclude some annotations from the AP list just because of some thousandth difference.
A good way to try may be combining this three options: considering the first 10% predicted annotations having value greater than τ, only if the difference between each annotation likelihood and the next annotation likelihood is lower than 33.33%. This combination might effectively lead to choose the reconstructed matrix *most important* annotations into the AP list.

In the database version comparison analysis, we mainly focus on the number of predicted annotations found on the updated version of the database, and, to a smaller degree, on the number of predicted annotations found with evidence IEA or ND on the analyzed database.
An improvement to this analysis can be made by inserting some different scores on the basis of the evidence of the predicted annotations found in the databases. For example, a score equal to 1 to a predicted annotation found as IEA or ND in the analyzed database version, a score equal to 2 to a predicted annotation found as IEA or ND in the updated database version, and a score equal to 3 to a predicted annotation found with evidence code different from IEA and from ND in the updated database version. Then, the final total prediction score would be computed by summing up all these predicted annotation scores.

For the Spearman coefficient and the Kendall distance, the penalty functions are not definitive and may be improved in the future. We may study some ways to consider also the variation of these coefficients when considering only the first 10%, 20%, 50% of the lists.

The DAG tree *novelty indicator*, based on the Schlicker [18] measure, spans in the interval $[0, \infty)$, because funded on the Resnik [14] similarity measure. For maximum novelty it gives value 0, while for the minimum novelty it provides a higher value, in theory going to infinity.
The absence of the upper bound makes it impossible to normalize all the values into a $[0\%, 100\%]$ interval, and makes it difficult to hypothesize the prediction minimum novelty value.

Finally, we can summarize here the list of the main contributions of this thesis:

- the improvement of the algorithm from which we departed (Truncated SVD by Khatri et al. [54]) with the insertion of the aumated data-driven choice of the key parameters, such as SVD truncation and likelihood threshold;
- the design and the implementation of two new methods, SIM1 and SIM2, that enhance the previously available Truncated SVD through gene clustering and term-term similarity weights, that generally lead to better results;
- the interpretation of the predicted annotations sorted by correctness likelihood, instead of considering them equally likely, as made by Khatri and colleagues in [54]:
- the design of new similarity measures (Extended Spearman and Extended Kendall) to better understand the difference between lists when varying the key parameters, that take into account the annotations in the NAC list, in addition to the annotations in the AP list;
- our software and algorithms may also be applied to non-GeneOntology datasets, and also to non-annotation datasets, such as metabolic pathways;
- our software may be easily extended with new algorithms, that might take advantage of already available modules (such as ROC curve depiction, database validation, etc).
- the design of the "novelty indicator", as visualization tool and as statistical measure, able to better comprehend the biological relevance level, whose graphs may be interpreted by non-experts, too;
- additional predictive reliability conditions, that join the items of information come from the "novelty indicator" with the prediction likelihood score and the percentage of validated annotations in the updated database, able to state further reliability about the correctness of the predicted annotations;
- our system is able to work efficiently even if a new updated version of the database is not available. In this case, it can take advantage of the ROC curve analysis, of the novelty indicator and on the join of these tools to state which are the annotations most likely to be correct;
- a final list of human predicted annotations, that we provide to the scientific community with the highest level of reliability given by our validation tools.

And here we sum up the main limits of the present work:

- heuristically defined threshold for the ROC AUC minimum percentage of reliability;
- the performance time is too high for large datasets, such as Homo sapiens GO annotations;
- the validation through literature and web tools is not automated, but done manually;
- the validation made through the database version comparison does not consider the same input dataset into the newer database, but instead a different dataset in which some genes and features may differ from the original.

12.1 Future developments

For the possible future developments, we see some different research directions. A good alternative to our methods and to Probabilistic Latent Semantic Analysis (pLSA) [111] could be Latent Dirichlet Allocation (LDA) [112]. We think that this method could be suitable for our goals because, given the relationships gene-topic and topic-term, it is able to reconstruct a probabilistic output matrix expressing the probability that a certain gene is related to a certain term.

Generally speaking, the problem of the prediction of biomolecular annotations could be dealt with some novel *on-line machine learning* methods, such as this by Lazaric et al. [113]. Differently from the classical machine learning approach, the on-line learning techniques are able to manage sets changing during time. While the classical machine learning are powerful only if all the data are already present at the beginning of the algorithm exection, the on-line methods can make prediction over data arriving little by little.
In fact, on-line learning algorithms are often used for the economics stock exchange data analysis. In the biomolecular annotations field, data always vary during time. This is why we think that an on-line learning approach may lead to good results.

For what concerns the validation component, we would like to make the literature text mining phase computerized. At the moment, no literature analysis software module is available for our analysis, but we hope to be able to add a connection to a system like the one by Nuzzo et al. [114] in the future.
To gain more useful information for validation, we will also consider the use of the ROC curve analysis tool pROC [115], able to run statistical tests on the AUCs and compare different ROCs.

Finally, we would like to make the software publicly available through the internet, to make it usable by all the scientific community. For this, we would like to release a web application version of it, based on the JSP and REST technologies, and integrate it into the Search Computing (SeCo) [116] [117] platform.

13 Bibliography

References

[1] M. Y. Galperin, and G. R. Cochrane, "Nucleic Acids Research Annual Database Issue and the NAR Online Molecular Biology Database Collection in 2009." *Nucleic Acids Res.* 37, Database issue, pp. 1-4, 2009.

[2] EMBL Nucleotide Sequence Database Statistics, `http://www3.ebi.ac.uk/Services/DBStats/`

[3] D. W. Huang, B. T. Sherman, R.A. Lempicki, "Bioinformatics Enrichment Tools: Paths toward the Comprehensive Functional Analysis of Large Gene Lists". *Nucleic Acids Res.* 37, pp. 1-13, 2009.

[4] F. Al-Shahrour, P. Minguez, J. Tarraga, I. Medina, E. Alloza, D. Montaner, J. Dopazo, "FatiGO+: A Functional Profiling Tool for Genomic Data. Integration of Functional Annotation, Regulatory Motifs and Interaction Data with Microarray Experiments". *Nucleic Acids Res.* 35, Web Server issue, pp. 91- 96, 2007.

[5] D. W. Huang, B. T. Sherman, Q. Tan, J. Kir, D. Liu, D. Bryant, Y. Guo, R. Stephens, M. W. Baseler, H. C. Lane and R. A. Lempicki, "DAVID Bioinformatics Resources: Expanded Annotation Database and Novel Algorithms to Better Extract Biology from Large Gene Lists". *Nucleic Acids Res.* 35, Web Server issue, pp. 169-175, 2007.

[6] M. Masseroli, D. Martucci, and F. Pinciroli, "GFINDer: Genome Function INtegrated Discoverer through Dynamic Annotation, Statistical Analysis, and Mining". *Nucleic Acids Res.* 32, pp. 293-300, 2004.

[7] M. Masseroli, "Management and Analysis of Genomic Functional and Phenotypic Controlled Annotations to Support Biomedical Investigation and Practice." IEEE Trans. Inf. Technol. Biomed. 11, 376–385 (2007)

[8] R. Sarati, "Metodi e Architetture Software per la Predizione di Annotazioni Genomiche Funzionali e la Stima di Similarita' Semantica tra Geni e Proteine", Politecnico di Milano, 2009.

[9] G. H. Golub, and C. Reinsch, "Singular value decomposition and least squares solutions". *Numerische Mathematik* 14.5: pp. 403-420, 1970.

[10] P. Drineas, A. Frieze, R. Kannan, S. Vempala, and V. Vinay, "Clustering large graphs via the singular value decomposition", *Machine Learning* 56.1-3: pp. 9-33, 2004.

[11] Z. Wu, and M. Palmer, "Verbs semantics and lexical selection". *Proceedings of the 32nd annual meeting on Association for Computational Linguistics.* Association for Computational Linguistics, 1994.

[12] D. Lin, "An information-theoretic definition of similarity." *ICML*, Vol. 98, 1998.

[13] R. Rada, H. Mili, E. Bicknell, and M. Blettner, "Development and application of a metric on semantic nets". *IEEE Transactions on Systems, Man and Cybernetics*, 19.1: pp. 17-30, 1989.

[14] P. Resnik, "Using information content to evaluate semantic similarity in a taxonomy" *arXiv preprint*, cmp-lg/9511007, 1995.

[15] P. Resnik, "Semantic similarity in a taxonomy: An information-based measure and its application to problems of ambiguity in natural language". *arXiv preprint* arXiv:1105.5444, 2011.

[16] D. Lin, "An information-theoretic definition of similarity". ICML. Vol. 98, 1998.

[17] J. J. Jiang, and D. W. Conrath, "Semantic similarity based on corpus statistics and lexical taxonomy." arXiv preprint cmp-lg/9709008, 1997.

[18] A. Schlicker, F. S. Domingues, J. Rahnenfoehrer, and T. Lengauer, "A new measure for functional similarity of gene products based on Gene Ontology". *BMC Bioinformatics* 7.1: 302, 2006

[19] D. Chicco, M. Masseroli, "A Discrete Optimization Approach for SVD Best Truncation Choice based on ROC Curves", *Proceedings of BIBE 2013, the 13rd IEEE International Conference on Bioinformatics and Bioengineering*, IEEE, 2013.

[20] W. Sujansky, "Heterogeneous Database Integration in Biomedicine". *Journal of Biomedical Informatics* 34, pp. 285-298, 2001.

[21] T. Hernandez, and S. Kambhampati, "Integration of Biological Sources: Current Systems and Challenges ahead". *SIGMOD Record* 33, pp. 51-60, 2004.

[22] O. Bodenreider, "The Unified Medical Language System (UMLS): integrating biomedical terminology". *Nucleic Acids Research*, 32, pp. 267-270, 2004.

[23] The Gene Ontology Consortium, "Creating the Gene Ontology Resource: Design and Implementation", *Genome Res.*, vol. 11, pp. 14251433, 2001.

[24] M. Ashburner, C. A. Ball, J. A. Blake, D. Botstein, H. Butler, J. M. Cherry, A. P. Davis, K. Dolinski, S. S. Dwight, J. T. Eppig, M. A. Harris, D. P. Hill, L. Issel-Tarver, A. Kasarskis, S. Lewis, J. C. Matese, J. E. Richardson, M. Ringwald, G. M. Rubin, and G. Sherlock, "Gene Ontology: tool for the unification of biology". *Nature genetics* 25.1: pp. 25-29, 2000.

[25] J. Tanoue, M. Yoshikawa, and S. Uemura, "The GeneAround GO viewer" *Bioinformatics* 18.12: pp. 1705-1706, 2002.

[26] J. A. Blake, J. Chan, R. Kishore, P. Sternberg, and K. Van Auken, "Gene Ontology annotations and resources". *Nucleic Acids Research* 41.D1: pp. 530-535, 2013.

[27] D. Croft , G. O'Kelly, G. Wu, R. Haw, M. Gillespie, L. Matthews, M. Caudy, P. Garapati, et al, "Reactome: A database of reactions, pathways and biological processes". *Nucleic Acids Research* 39 (Database issue): pp. 691D697, 2010.

[28] A. Hamosh, A. Scott, J. Amberger, C. Bocchini, V. McKusick, "Online Mendelian Inheritance in Man (OMIM), a knowledgebase of human genes and genetic disorders". *Nucleic Acids Research* 33 (Database issue): pp. 514517, 2004.

[29] R. R. Brinkman et al. "Modeling biomedical experimental processes with OBI". *Journal of Biomedical Semantics* 1.Suppl 1: S7, 2010.

[30] P., De Matos, R. Alcantara, A. Dekker, M. Ennis, J. Hastings, K. Haug, I. Spiteri, S. Turner, et al. "Chemical Entities of Biological Interest: An update". *Nucleic Acids Research* 38 (Database issue): pp. 24954, 2009.

[31] N. Le Novre, "BioModels.net, tools and resources to support Computational Systems Biology". *Proceedings of the 4th Workshop on Computation of Biochemical Pathways and Genetic Networks*, Logos, Berlin, pp. 69-74, 2005.

[32] C. E. Lipscomb, "Medical subject headings (MeSH)". *Bulletin of the Medical Library Association* 88.3: 265, 2000.

[33] D. Binns, et al., "QuickGO: a web-based tool for Gene Ontology searching". *Bioinformatics* 25.22: pp. 3045-3046, 2009.

[34] P. D. Karp, "What we do not know about sequence analysis and sequence databases". *Bioinformatics* 14.9: pp. 753-754, 1998.

[35] Biology StackExchange.com, "How much does a biological experiment to validate a gene function cost?". `http://biology.stackexchange.com/questions/13988/how-much-does-a-biological-experiment-to-validate-a-gene-function-cost`, December 2013.

[36] G. Pandey, V. Kumar, and M. Steinbach,"Computational approaches for protein function prediction: A survey". Twin Cities: Department of Computer Science and Engineering, University of Minnesota, 2006.

[37] O. D. King, et al. "Predicting gene function from patterns of annotation." *Genome research* 13.5: pp. 896-904, 2003.

[38] Y. Tao, et al., "Information theory applied to the sparse gene ontology annotation network to predict novel gene function". *Bioinformatics* 23.13: pp. 529-538, 2007.

[39] X. Jiang, et al., "Combining hierarchical inference in ontologies with heterogeneous data sources improves gene function prediction". *Proceedings of BIBM 2008, the IEEE International Conference on Bioinformatics and Biomedicine*, IEEE, 2008.

[40] O. G. Troyanskaya, et al., "A Bayesian framework for combining heterogeneous data sources for gene function prediction (in Saccharomyces cerevisiae)." *Proceedings of the National Academy of Sciences* 100.14: pp. 8348-8353, 2003.

[41] B. V. Dasarathy, "Nearest Neighbor (NN) Norms:NN Pattern Classification Techniques", 1991.

[42] H. Mi, A. Muruganujan, and P. D. Thomas, "PANTHER in 2013: modeling the evolution of gene function, and other gene attributes, in the context of phylogenetic trees". *Nucleic acids research* 41.D1: pp. 377-386, 2013

[43] L. E. Baum, and T. Petrie, "Statistical inference for probabilistic functions of finite state Markov chains". *The annals of mathematical statistics* 37.6: pp. 1554-1563, 1966.

[44] X. Deng, and H. Ali, "A hidden markov model for gene function prediction from sequential expression data". *Proceedings of CSB 2004, the IEEE Computational Systems Bioinformatics Conference*, IEEE, 2004.

[45] C. Cortes, and V. Vapnik, "Support-vector networks." *Machine learning* 20.3: pp. 273-297, 1995.

[46] Z. Barutcuoglu, R. E. Schapire, and O. G. Troyanskaya, "Hierarchical multi-label prediction of gene function". *Bioinformatics* 22.7: pp. 830-836, 2006.

[47] F. Minneci, et al. "FFPred 2.0: Improved Homology-Independent Prediction of Gene Ontology Terms for Eukaryotic Protein Sequences". *PloS* one 8.5: e63754, 2013.

[48] N. Mitsakakis, et al. "Prediction of Drosophila melanogaster gene function using Support Vector Machines". *BioData Mining* 6: 8, 2013.

[49] J. Quackenbush, "Microarraysguilt by association." Science 302.5643: pp. 240-241, 2003.

[50] X. Li, et al. "Graph kernel-based learning for gene function prediction from gene interaction network". *Proceedings of BIBM 2007, the IEEE International Conference on Bioinformatics and Biomedicine*, IEEE, 2007.

[51] M. B. Eisen, et al. "Cluster analysis and display of genome-wide expression patterns". *Proceedings of the National Academy of Sciences* 95.25: pp. 14863-14868, 1998.

[52] P. Pavlidis, and J. Gillis, "Progress and challenges in the computational prediction of gene function using networks." *F1000Research* 1, 2012.

[53] D. Warde-Farley, et al. "The GeneMANIA prediction server: biological network integration for gene prioritization and predicting gene function". *Nucleic Acids Research* 38 suppl 2: pp. 214-220, 2010.

[54] P. Khatri, et al. "A semantic analysis of the annotations of the human genome". *Bioinformatics* 21.16: 3416-3421, pp. 2005.

[55] B. Done, et al. "Semantic analysis of genome annotations using weighting schemes". *Proceedings of CIBCB 2007 , the IEEE Symposium Computational Intelligence and Bioinformatics and Computational Biology*, IET, 2007.

[56] B. Done, et al. "Predicting novel human gene ontology annotations using semantic analysis." *IEEE/ACM Transactions on Computational Biology and Bioinformatics (TCBB)* 7.1: pp. 91-99, 2010

[57] M. Tagliasacchi, and M. Masseroli, "Anomaly-free Prediction of Gene Ontology Annotations using Bayesian Networks" *Proceedings of BIBE 2009, the Ninth IEEE International Conference on Bioinformatics and BioEngineering*. IEEE, 2009.

[58] V. N. M. Aradhya, F. Masulli, and S. Rovetta, "A novel approach for biclustering gene expression data using modular singular value decomposition." *Computational Intelligence Methods for Bioinformatics and Biostatistics*. Springer Berlin Heidelberg. pp. 254-265, 2010.

[59] R. Homayouni, et al. "Gene clustering by latent semantic indexing of MEDLINE abstracts". *Bioinformatics* 21.1: pp. 104-115, 2005.

[60] D. Chicco, M. Tagliasacchi, and M. Masseroli, "Genomic Annotation Prediction Based on Integrated Information". *Computational Intelligence Methods for Bioinformatics and Biostatistics*, Springer Berlin Heidelberg, pp. 238-252, 2012.

[61] P. Cremonesi, Y. Koren and R. Turrin, "Performance of recommender algorithms on top-n recommendation tasks". *Proceedings of the fourth ACM conference on Recommender systems*. ACM, 2010.

[62] S. Chiaverini, "Singularity-robust task-priority redundancy resolution for real-time kinematic control of robot manipulators". *Proceedings of the IEEE Transactions on Robotics and Automation* 13.3: pp. 398-410, 1997.

[63] M. Frank, and J. M. Buhmann, "Selecting the rank of truncated SVD by Maximum Approximation Capacity". *Proceedings of ISIT 2011, the IEEE International Symposium on Information Theory*, IEEE, 2011.

[64] C. R. Vogel, "Optimal choice of a truncation level for the truncated SVD solution of linear first kind integral equations when data are noisy". *SIAM journal on numerical analysis* 23.1: pp. 109-117, 1986.

[65] S. Isam, I. Kanaras, and I. Darwazeh, "A Truncated SVD approach for fixed complexity spectrally efficient FDM receivers". *Proceedings of WCNC 2011, the Wireless Communications and Networking Conference*, IEEE, 2011.

[66] K. Jbilou, L. Reichel, and H. Sadok, "Vector extrapolation enhanced TSVD for linear discrete ill-posed problems". *Numerical Algorithms* 51.2: pp. 195-208, 2009.

[67] P. C. Hansen "The discrete Picard condition for discrete ill-posed problems". *BIT Numerical Mathematics* 30.4: pp. 658-672, 1990.

[68] T. Fawcett, "ROC graphs: Notes and practical considerations for researchers". *Machine Learning* 31: pp. 1-38, 2004

[69] NCBI PubMed Health.National Library of Medicine (Bethesda, Marylan, USA). `http://www.ncbi.nlm.nih.gov/pubmed`

[70] S. Carbon, A. Ireland, C. J. Mungall, S. Shu, B. Marshall, S. Lewis, AmiGO Hub, Web Presence Working Group, "AmiGO: online access to ontology and annotation data". *Bioinformatics*, 25(2): pp. 288-9, 2009.

[71] M. Rebhan, et al. "GeneCards: a novel functional genomics compendium with automated data mining and query reformulation support." *Bioinformatics* 14.8: pp. 656-664, 1998.

[72] S. T. Dumais, "Improving the retrieval of information from external sources", *Behavior Research Methods, Instruments, and Computers* 23.2: pp. 229-236, 1991.

[73] R. Ostrovsky, and Y. Rabani, "Polynomial time approximation schemes for geometric K-clustering". *Proceedings of 41st Annual Symposium on Foundations of Computer Science*, IEEE, 2000.

[74] M. Charikar, et al., "A constant-factor approximation algorithm for the k-median problem". *Proceedings of the 31st annual ACM symposium on Theory of computing*. ACM, 1999.

[75] N. Diaz-Diaz, and J. S. Aguilar-Ruiz, "GO-based functional dissimilarity of gene sets". *BMC bioinformatics* 12.1: 360, 2011.

[76] X. Guo, et al. "Assessing semantic similarity measures for the characterization of human regulatory pathways". *Bioinformatics* 22.8: pp. 967-973, 2006.

[77] R. Fagin, R., Kumar, and D. Sivakumar, "Comparing top K lists". *SIAM Journal on Discrete Mathematics* 17.1: pp. 134-160, 2003.

[78] L. A. Goodman, and W. H. Kruskal, "Measures of association for cross classifications", *Journal of the American Statistical Association* 49.268: pp. 732-764, 1954.

[79] R. A. Laskowski, J. D. Watson, and J. M. Thornton, "Protein function prediction using local 3D templates". *Journal of Molecular Biology* 351.3: pp. 614-626, 2005.

[80] C. E. Jones, U. Baumann, and A. L. Brown, "Automated methods of predicting the function of biological sequences using GO and BLAST". *BMC Bioinformatics* 6.1: 272, 2005.

[81] S. F. Altschul, et al. "Basic local alignment search tool". *Journal of molecular biology* 215.3: pp. 403-410, 1990

[82] A. J. Perez, et al. "Gene annotation from scientific literature using mappings between keyword systems". *Bioinformatics* 20.13: pp. 2084-2091, 2004.

[83] M. A. Andrade, and A. Valencia, "Automatic extraction of keywords from scientific text: application to the knowledge domain of protein families". *Bioinformatics* 14.7: pp. 600-607, 1998.

[84] Chiang, J., et al., "GeneLibrarian: an effective gene-information summarization and visualization system". *BMC Bioinformatics* 7.1: 392, 2006.

[85] S. Raychaudhuri, et al. "Associating genes with gene ontology codes using a maximum entropy analysis of biomedical literature." *Genome Research* 12.1: pp. 203-214, 2002.

[86] A. Canakoglu, G. Ghisalberti, and M. Masseroli, "Integration of Biomolecular Interaction Data in a Genomic and Proteomic Data Warehouse to Support Biomedical Knowledge Discovery". *Computational Intelligence Methods for Bioinformatics and Biostatistics*. Springer Berlin Heidelberg. pp. 112-126, 2012.

[87] F. Pessina, M. Masseroli, and A. Canakoglu, "Visual composition of complex queries on an integrative Genomic and Proteomic Data Warehouse". *Engineering* (in press).

[88] M. Masseroli, M. Tagliasacchi, "Web resources for gene list analysis in biomedicine", In: Lazakidou, A., editor. *Web-based Applications in Health Care and Biomedicine*. Heidelberg, D: Springer, Annals of Information Systems Series, vol. 7, pp. 117–141, 2010

[89] R. Kohavi, "A study of cross-validation and bootstrap for accuracy estimation and model selection". *Proceedings of IJCAI*, Vol. 14, No. 2, 1995.

[90] J. Chen, and Y. Saad, "Lanczos Vector versus Singular Vectors for Effective Dimension Reduction". *Technical Report*, Department of Computer Science and Engineering, University of Minnesota, Minneapolis, MN, USA, 2008.

[91] JFreeChart JFree.org, `http://www.jfree.org/`

[92] AMD Core Math Library (ACML), `http://developer.amd.com/cpu/libraries/acml/`

[93] D. Rohde, "SVDLIBC", `http://tedlab.mit.edu/~dr/SVDLIBC`

[94] L. Dagum, R. Menon, "OpenMP: an industry standard API for shared-memory programming". *IEEE Computational Science & Engineering* 5, 46–55 (1998)

[95] R. Gordon, "Essential JNI: Java Native Interface". *Prentice-Hall*, Inc., NJ, Usa (1998)

[96] C. L. Lawson, R. J. Hanson, D. R. Kincaid, F. T. Krogh, "Basic Linear Algebra Subprograms for Fortran Usage". *ACM Transactions on Mathematical Software (TOMS)* 5, 1979.

[97] M. Berry, T. Do, G. O'Brien, V. Krishna, S. Varadhan, "SVDPACKC (Version 1.0) User's Guide". *Citeseer*, 1993.

[98] WikiVs.com, MySQL vs PostgreSQL, http://www.wikivs.com/wiki/MySQL_vs_PostgreSQL

[99] B. Angerson, G. Dongarra, D. C. McKenney, et al., LAPACK: "A portable linear algebra library for high-performance computers". *Proceedings of the 1990 ACM/IEEE Conference on Supercomputing*, pp. 2–11. IEEE Computer Society Press Los Alamitos, CA, Usa (1990)

[100] PostgreSQL, http://www.postgresql.org

[101] J. Ellson, et al. "Graphviz - open source graph drawing tools." *Graph Drawing*. Springer Berlin Heidelberg, 2002.

[102] Iterative and incremental development, Wikipedia in English, http://en.wikipedia.org/wiki/Iterative_and_incremental_development.

[103] C. Spearman, "The proof and measurement of association between two things". *Amer. J. Psychol.* 15: 72101, 1904.

[104] M. Kendall, "A New Measure of Rank Correlation". *Biometrika*, 30, 81-89, 1938.

[105] A. Schlicker, and M. Albrecht "FunSimMat: a comprehensive functional similarity database." *Nucleic acids research* 36 suppl 1: D434-D439, 2008.

[106] P. W. Lord, et al. "Investigating semantic similarity measures across the Gene Ontology: the relationship between sequence and annotation." *Bioinformatics* 19.10: 1275-1283, 2003.

[107] N. Speer, C. Spieth, and A. Zell, "A memetic clustering algorithm for the functional partition of genes based on the gene ontology." *Proceedings of CIBCB 2004.*, the IEEE Symposium on Computational Intelligence in Bioinformatics and Computational Biology, 2004.

[108] E. Ciceri, P. Fraternali, D. Martinenghi, and M. Tagliasacchi, "Humans fighting uncertainty", Dipartimento di Elettronica, Informazione e Bioingegneria, Politecnico di Milano, 2013. *In preparation.*

[109] A. Bairoch, et al., "The universal protein resource (UniProt)." *Nucleic Acids Research* 33 suppl 1: D154-D159, 2005.

[110] R. Sasisekharan, R. Raman, and V. Prabhakar, "Glycomics approach to structure-function relationships of glycosaminoglycans". *Annu. Rev. Biomed. Eng.* 8: 181-231, 2006.

[111] M. Masseroli, D. Chicco, and P. Pinoli, "Probabilistic Latent Semantic Analysis for prediction of Gene Ontology annotations". *Proceedings of IJCNN 2012*, the International Joint Conference on Neural Networks (IJCNN). IEEE, 2012.

[112] D. M. Blei, A. Y. Ng, and M. I. Jordan, "Latent Dirichlet Allocation". *Journal of machine Learning Research* 3: 993-1022, 2003

[113] A. Lazaric, and R. Munos, "Hybrid Stochastic-Adversarial On-line Learning". *Proceedings of COLT*, 2009.

[114] A. Nuzzo, F. Mulas, M. Gabetta, E. Arbustini, B. Zupan, C. Larizza, R. Bellazzi, "Text Mining approaches for automated literature knowledge extraction and representation". *Studies in Health Technology and Informatics* 160(Pt 2), pp. 954–8 (2010).

[115] X. Robin, N. Turck, A. Hainard, N. Tiberti, F. Lisacek, J. C. Sanchez and M. Moeller, "pROC: an open-source package for R and S+ to analyze and compare ROC curves". *BMC Bioinformatics*, vol. 12, pp. 77–84, 2011.

[116] S. Ceri, D. Braga, F. Corcoglioniti, M. Grossniklaus, and S. Vadacca, "Search computing challenges and directions". *Springer*, Berlin Heidelberg, 2010.

[117] D. Chicco, Integration of Bioinformatics Web Services through the Search Computing Technology. *Technical Report*, TR 2012/02. Dipartimento di Elettronica e Informazione, Politecnico di Milano, Milan, Italy.

14 Lists of figures, tables, acronyms, algorithms, gene symbols

List of Figures

List of Tables

List of Algorithms

- Semantically improved SVD with gene clustering and term-term similarity weights (SIM2): Section 5.4;
- SIM best cluster number choice: Section 5.6;
- Singular Value Decomposition (SVD): Section 5.2;
- SVD best truncation level choice: Section 5.6;
- Ten-fold cross validation: Section 5.5;

List of Acronyms

- anDb: analyzed database;
- AC: Annotation Confirmed;
- AP: Annotation Predicted;
- AR: Annotation to be Reviewed;
- AUC: Area Under the Curve;
- BP: Biological Process;
- CC: Cellular Component
- DAG: Directed Acyclic Graph;
- FN: False Negative;
- FP: False Positive;
- GPDW: Genomic Proteomic Data Warehouse;
- GO: Gene Ontology;
- ID: identifier;
- IEA: Inferred from Electronic Annotation;
- LCA: Last common ancestor;
- LDA: Latent Dirichlet Allocation;
- MF: Molecular Function;
- NAC: Non existing Annotation Confirmed;
- ND: No biological Data available;
- OID: Own identifier
- ROC: Receiver Operating Characteristic;
- SIM: Semantically IMproved Singular Value Decomposition method. SIM1: with gene clustering; SIM2: with gene clustering and term-term similarity weights;

- SVD: Singular Value Decomposition. SVD-Khatri: SVD method developed by Khatri and colleagues [54]. SVD-us: SVD method with key parameters optimized by our algorithms;
- TN: True Negative;
- TP: True Positive;
- upDb: updated database.

List of Gene Symbols

- AHR: aryl hydrocarbon receptor [Homo sapiens (human)]
- CCR2: chemokine (C-C motif) receptor 2 [Homo sapiens (human)]
- CHST14: carbohydrate (N-acetylgalactosamine 4-0) sulfotransferase 14 [Homo sapiens (human)]
- CNOT2: CCR4-NOT transcription complex, subunit 2 [Homo sapiens (human)]
- CPA2: carboxypeptidase A2 (pancreatic) [Homo sapiens (human)]
- DPM1: dolichyl-phosphate mannosyltransferase polypeptide 1, catalytic subunit [Homo sapiens (human)]
- FNTA: farnesyltransferase, CAAX box, alpha [Homo sapiens (human)]
- HDAC6: histone deacetylase 6 [Homo sapiens (human)]
- ITGA6: integrin, alpha 6 [Homo sapiens (human)]
- MMP23B: matrix metallopeptidase 23B [Homo sapiens (human)]
- NT5M: 5',3'-nucleotidase, mitochondrial [Homo sapiens (human)]
- P2RY14: purinergic receptor P2Y, G-protein coupled, 14 [Homo sapiens (human)]
- POR: P450 (cytochrome) oxidoreductase [Homo sapiens (human)]
- PPME1: protein phosphatase methylesterase 1 [Homo sapiens (human)]
- RAB26: RAB26, member RAS oncogene family [Homo sapiens (human)]
- RARA: retinoic acid receptor, alpha [Homo sapiens (human)]
- ROPN1B: rhophilin associated tail protein 1B [Homo sapiens (human)]
- SLC13A2: solute carrier family 13 (sodium-dependent dicarboxylate transporter), member 2 [Homo sapiens (human)]
- SLC1A6: solute carrier family 1 (high affinity aspartate/glutamate transporter), member 6 [Homo sapiens (human)]
- VMA9: Vma9p [Saccharomyces cerevisiae S288c]

Reply to the reviewers

Here we report the main corrections and changes made to the thesis first draft, according to the reviewers' main comments and suggestions.

Elena Marchiori's review:

- we corrected all the indicated typos and grammar errors;
- we expanded Section 3 (Motivations and goals), by adding several sentences;
- we inserted a sentence in Section 3 that refers to the thesis main results reported in Section 12 (Conclusions).

Jean-Philippe Vert's review:

- we added more explanations and justifications regarding the gene clustering technique introduced in the Section 5.3;
- regarding the ten-fold cross validation in Section 5.5, we think that the reviewer misunderstood our description, that involves the 10% of the elements of the datasets, and it is not a leave-one-out procedure. We did not modify this part.
- regarding the computational complexity (Section 5.7), the reviewer states we omit the ten-fold cross validation procedure from the final complexity computation. We omitted it because the final complexity esteem does not change: if C(N) is the complexity of the a method applied to a dataset of N elements, by adding ten-fold-cross-validation it would become $10 \cdot C(N \cdot 90\%)$, that is equal to C(N). We did not modify this part.
- regarding the use of ten-fold-cross-validation in the key parameter selection algorithms, we modified Section 5.6 by adding a sentence that confirms the use of it.

We deeply thank the Elena Marchiori and Jean-Philippe Vert for their excellent and detailed reviews.