Bioinformatics and Biology Insights

Libertas Academica
FREEDOM TO RESEARCH

OPEN ACCESS
Full open access to this and thousands of other papers at http://www.la-press.com.

ORIGINAL RESEARCH

Knowledge Discovery in Variant Databases Using Inductive Logic Programming

Hoan Nguyen[1], Tien-Dao Luu[1,2], Olivier Poch[1] and Julie D. Thompson[1]

[1]Laboratoire de Bioinformatique et Génomique Intégratives, Institut de Génétique et de Biologie Moléculaire et Cellulaire Illkirch, France. [2]Cantho University Software Center, Cantho City, Vietnam.
Corresponding author email: nguyen@igbmc.fr

Abstract: Understanding the effects of genetic variation on the phenotype of an individual is a major goal of biomedical research, especially for the development of diagnostics and effective therapeutic solutions. In this work, we describe the use of a recent knowledge discovery from database (KDD) approach using inductive logic programming (ILP) to automatically extract knowledge about human monogenic diseases. We extracted background knowledge from MSV3d, a database of all human missense variants mapped to 3D protein structure. In this study, we identified 8,117 mutations in 805 proteins with known three-dimensional structures that were known to be involved in human monogenic disease. Our results help to improve our understanding of the relationships between structural, functional or evolutionary features and deleterious mutations. Our inferred rules can also be applied to predict the impact of any single amino acid replacement on the function of a protein. The interpretable rules are available at http://decrypthon.igbmc.fr/kd4v/.

Keywords: SNP prediction; inductive logic programming; human monogenic disease; genotype-phenotype relation

Bioinformatics and Biology Insights 2013:7 119–131

doi: 10.4137/BBI.S11184

This article is available from http://www.la-press.com.

© the author(s), publisher and licensee Libertas Academica Ltd.

This is an open access article published under the Creative Commons CC-BY-NC 3.0 license.

Introduction

Single nucleotide polymorphisms (SNPs) refer to a genetic change in which one nucleotide is replaced by another one. SNPs represent one of the most common forms of human genomic variation. SNPs are highly abundant, stable and distributed throughout the genome.[1] Although SNPs are primarily associated with population diversity and individuality, they can also be linked to the emergence of or predisposition to a disease, influencing its severity, its progression or its drug sensitivity.

SNPs that are directly linked to the emergence of a disease are considered to be deleterious. These deleterious SNPs occur in both non protein-coding and protein-coding regions. In the first case, the variation will often affect gene expression by disrupting transcription factor-binding sites, splice sites or other functional sites at the transcriptional level. Protein-coding SNPs can be further divided into synonymous and non-synonymous (nsSNPs). nsSNPs, also called missense mutations, result in the alteration of the amino acid sequence of the encoded protein. nsSNPs have been linked to a wide variety of diseases; for example, by affecting protein function, by reducing protein solubility or by destabilizing protein structure.[2] These protein alterations are considered to be the primary molecular phenotype linked to the missense mutation, with a cascade of consequences that finally leads to the emergence of a genetic disease and the associated phenotype. The elucidation of the complex relationships linking genotypic and phenotypic variations is a major challenge in the post-genomic era.

With the huge amount of information now available in various biological databases, including sequences, structures, functions, pathways, interactions and variations,[3] it is now feasible to develop in silico analysis tools to better understand and/or to predict the correlation between a missense mutation and the associated molecular phenotypes. Several research groups have addressed this topic and have developed tools aimed at predicting the effects of nsSNPs on the function of a protein, with varying degrees of success.[4]

Current methods of prediction can be divided into two main categories. The first category encompasses sequence-based methods, generally based on multiple sequence alignments and incorporating different approaches to quantify the conservation of a residue during evolution. This category includes SIFT,[5] PANTHER,[6] PMUT,[7] PhD-SNP,[8] SNAP[9] and LRT.[10] The second category combines both sequence and protein 3D structure data. Although these methods are limited by the availability of structural data, various studies have shown that the inclusion of structural information can improve the performance of prediction methods based only on sequence data.[11] These studies also provide evidence that a majority of nsSNPs has an impact on the structure.[12] The most widely used methods in this category are PolyPhen,[13] nsSNPAnalyzer,[14] SNPs3D,[15] AutoMute[16] and more recently Polyphen-2.[17]

The effectiveness of a prediction method is mainly based on the choice of predictors and on the underlying computational approaches. The latter are numerous and include the use of empirically derived rules,[13] Support Vector Machines,[8,15] neural networks,[7,9] random forests,[14,16] Hidden Markov Models,[6] statistical models,[5] or Naïve Bayes.[17] All these methods have their strengths and weaknesses (for a review, see[4]). While it is not straightforward to compare these methods using the same quality criteria, most of them seem to perform well for classification purposes. In particular, most of them classify nsSNPs as either deleterious (strong functional effect) or neutral (weak functional effect) with high accuracy. Unfortunately, little explanation concerning the decision computed by these prediction tools is available. PolyPhen provides the rules to predict the effect of nsSNPs on protein function and structure, but these rules are derived empirically. Access to such information is essential in order to understand how genetic alterations affect gene products at the molecular level and subsequently to elucidate the relationships between genotypic and phenotypic variations.

In this context, the extraction of knowledge from large-scale mutation data is an increasingly challenging task and KDD approaches are now being applied in many domains of biomedicine. KDD is commonly defined as the "non trivial process of identifying valid, novel, potentially useful and ultimately understandable patterns in data".[18] From an operational point of view, the KDD process is performed within a KDD system including databases, data mining modules, and interfaces for human interactions, such as editing and visualization. The KDD process involves

three main operations: selection and preparation of the data, data mining, and finally interpretation of the extracted units. Most data mining algorithms used in KDD accept as input a single database table where the data to mine are represented as objects displaying specific values for given properties. This constraint is very prohibitive in the case of biomedical data, where different types of objects are often stored in different tables. This has led to the exploitation of logic reasoning approaches, such as Inductive Logic programming (ILP).[19,20] ILP is a method by which a computer program can learn concepts, or rules, by example. It has been applied successfully to various bioinformatics problems including breast cancer studies,[21] protein structure prediction,[22,23] gene function prediction,[24] protein-protein interaction prediction,[25] protein-ligand interaction prediction[26] and microarray data classification.[27]

Recently, we implemented the ILP method in the KD4v system[28] for the interpretation and prediction of the phenotypic effects of missense variants. The performance of the ILP prediction was shown to be comparable to SIFT and PolyPhen-2, the most commonly used methods in the field. The power of the KD4V approach has also been demonstrated in a number of recent studies devoted to specific human diseases, including multifactorial diseases (complete congenital stationary night blindness).[29,30]

In this paper, we describe a study dedicated to the impact of mutations on protein function in the context of human monogenetic diseases. We extracted background knowledge from the MSV3d[31] annotated mutation database, which integrates a large number of human mutations and phenotypes. We then applied ILP and a clustering analysis to learn a set of rules that can be easily interpreted by the biologist and reused for the prediction of the functional effects of a mutation.

Materials and Methods

MSV3d

The datasets used in this study were taken from the relational database MSV3d ("Database of human missense variants mapped to 3D protein structures", publicly accessible online at decrypthon.igbmc.fr/msv3d). MSV3d is designed to facilitate the investigation of the structural and functional impacts of missense mutations with regard to their phenotypic effects in the context of human genetic diseases. Based on Multiple Alignments of Complete Sequences (MACS)[32] and 3D structures available in the SM2PH database,[33] MSV3d annotates each mutation with various parameters describing the physico-chemical changes induced by the amino acid modification, as well as the conservation of the mutated residue and its position relative to functional features in the available or predicted 3D model.

The human missense variants in MSV3d are mainly retrieved from the dbSNP[34] and UniProtKB databases,[35] but also from several Locus Specific DataBases[36] (LSDBs), such as the ALPL gene mutations database. MSV3d classifies these variants in two categories: mutations linked to known human diseases (deleterious) and those associated with the "polymorphism" term (neutral), in accordance with the nomenclature used in the UniProtKB database.

MVS3d currently contains a total of 404,425 missense variants from 20,219 human proteins, among which 27,401 are considered as deleterious and 377,024 as neutral. Concerning structure modelling, the database contains 58,726 variants mapped to 3D structure, among which 7,113 variants are associated with 1,278 OMIM disease annotations. The database facilities exploration of the relationships between genetic variations and 3D structure via a unified access to databases, including SOAP web services, a Java API, simple queries and full or partial database download services. In addition, the database represents a useful benchmark set for the development and evaluation of machine learning methods for classification or prediction of deleterious/neutral mutations.

Inductive logic programming (ILP)

ILP[37] combines machine learning and logic programming. Given a formal encoding of the background knowledge and a set of examples, an ILP system will derive hypotheses which explain all the positive examples and none, or almost none, of the negative examples. In this approach, logic is used as a language to induce hypotheses from the examples and background knowledge. Briefly, the basic form of the ILP problem is defined as follows.

Given

- A background knowledge B which is the knowledge available before the learning.

- A finite set of examples E, $E = E^+ \cup E^-$ where E^+ is a nonempty set of positive examples, and E^- is a set of negative examples.

Find: hypotheses H (set of rules), such that:

- All or almost all positive examples $e \in E^+$ are covered by H.
- No or few negative examples are covered by H.

In comparison to other machine learning approaches, ILP has several advantages. Firstly, in data mining, ILP is able to discover knowledge from a multi-relational database consisting of multiple tables. Thus, ILP is also called multi-relational data mining.[20] Secondly, using logic programming allows to encode more general forms of background knowledge such as recursions, functions or quantifiers.[38] Finally, the learned rules are comprehensible by humans and computers and can be interpreted without the need for visualization.

Here, our goal is to use ILP to translate a mutation database to a mutation knowledge base. The steps to achieve this goal are illustrated in Figure 1 and the following sections expand on each of the stages.

Problem definition and example construction

The first task involves identifying the problem and translating it into positive and negative examples. Here, we have limited our study to the task of discriminating deleterious/neutral mutations. The list of proteins known to be involved in human monogenic diseases was obtained from the OMIM database.[39] Based on this list, we selected the 8,117 mutations related to monogenic disease with high quality 3D models in MSV3d.

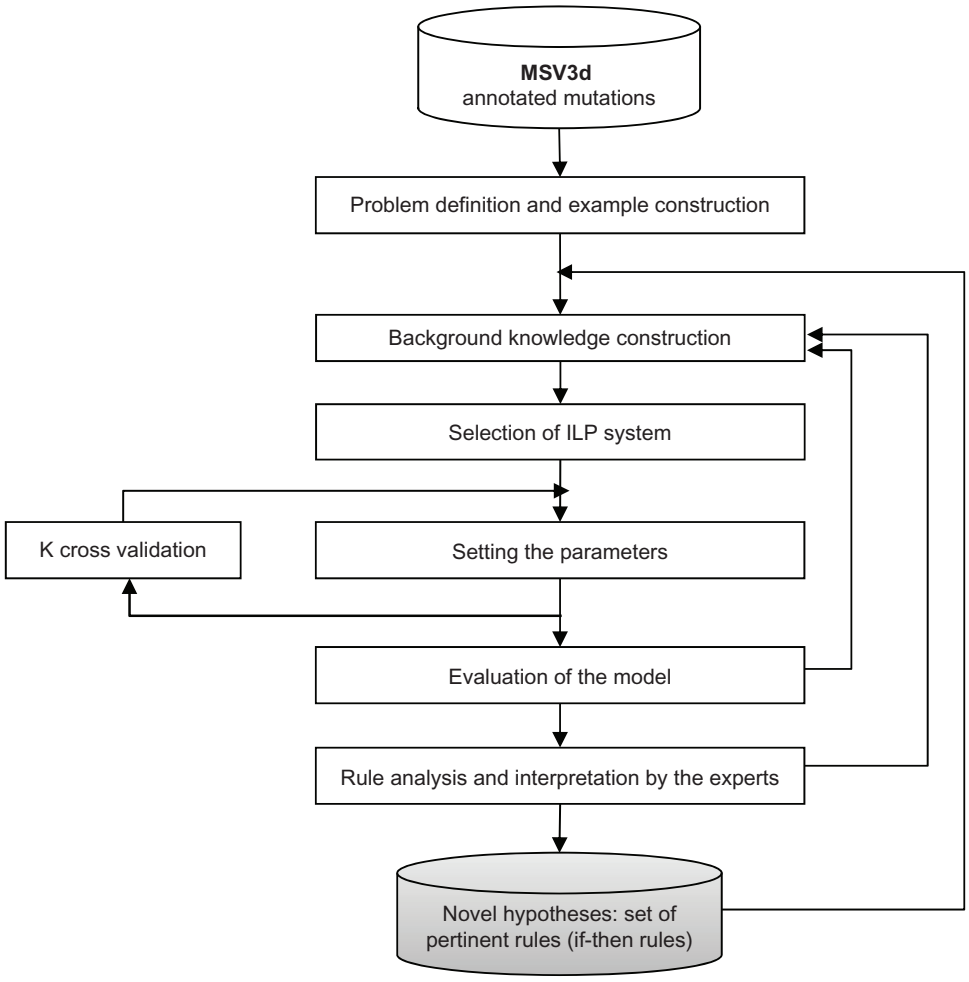


Figure 1. Main steps for an ILP application include: (i) mutation selection from MSV3d, (ii) definition of negative/positive examples in the training set, (iii) background knowledge creation, (iv) selection of the ILP system, (v) selection of the ILP parameters (number of nodes, noisy..) and optimization of the predicates in the background knowledge, (vi) model evaluation using K-fold cross validation, and (vii) the final rules used for interpretation.

Of these, 6,480 deleterious mutations constitute the positive examples. The remaining 1,637 neutral mutations constitute the negative examples.

In this work, the positive and negative examples are formalized as facts in the Prolog language, which represents a logical formula in predicate logic. A fact is a predicate expression that makes a declarative statement about the problem domain. The predicate for the positive examples is "is_deleterious". For example, a positive example in Prolog syntax is:

is_deleterious(m_Q13496_Asn180Lys)

indicating that, in protein Q13496, the replacement of the Asparagine at position 180 by a Lysine is deleterious.

The positive examples are written in a file with a ".f" extension. The negative examples use the same predicate "is_deleterious", but they are written in a different file (with a ".n" extension).

Background knowledge construction

The molecular consequences of missense mutations are related to the functional and structural contexts of the affected position, as well as to the physicochemical characteristics of the substitution. All these types of information are represented in MSV3d for the stored missense mutations and they are used as background knowledge in this study. Detailed descriptions of this information are available in.[31]

Very recently, Kowarsch et al[40] noted that "we believe that researchers should not only look at conservation in their judgment of functional significance of residues in the protein sequence. Correlation patterns between residues clearly provide additional evidence which should not be ignored". To study the effect of neighbouring amino acid residues on a missense mutation, we enhanced the database by including the following additional features:

- Neighbouring residues. Residues are considered to be neighbours of a mutation if they occur within a sphere of radius 10 A°. For example, Figure 2 shows the neighbouring residues of the missense mutation p.Asn180Lys in protein Q13496.
- Classification of amino acids. We used the amino acid classification system of Koolman[41] in which the amino acids are divided into aliphatic, acidic, basic, sulfur-containing, aromatic, neutral and imino, based on the side chain chemical features.

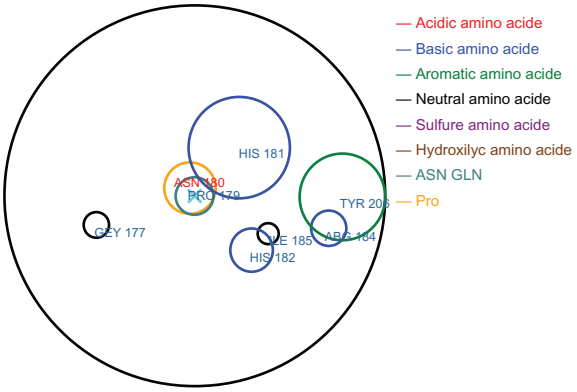


Figure 2. Definition of neighbouring residues.
Notes: For the mutated residue, Asn180 of protein Q13496, a sphere of radius 10 A° is drawn with the residue in the centre. Any residues that lie within the sphere are defined as neighbours.

The use of these features, together with physicochemical, functional, 3D structural and evolutionary features of the missense mutations allows us to discover hidden knowledge from different points of view of the missense mutations. The complete multi-relational data model used in our analysis is shown in Figure 3.

In order to use ILP, we used SQL scripts to translate the information for each mutation stored in the MSV3d database (PostgreSQL) into Prolog facts (Fig. 4). Each feature associated with a mutation is represented in the form: p(ModeType, ModeType, …), where the ModeType is one of: (1) +ModeType specifying the input, (2) –ModeType specifying the output and (3) #ModeType specifying a constant. Table 1 describes all the Prolog facts derived from our mutation data model.

Selection of ILP system and parameters

Many ILP systems have been developed and successfully applied to diverse domains, eg, FOIL,[42] Progol,[43] Tilde[44] and Aleph[45]. We chose Aleph (Version 5) with the SWI-Prolog compiler (Version 5.6.47) to learn rules from our set of examples because of its popularity, frequent update and flexibility. Aleph is also very attractive since it is coded in Prolog and is thus relatively easy to modify. The Aleph algorithm is based on the classic ILP framework involving five main steps:

- Select an uncovered positive example
- Find all the Prolog facts that explain this example

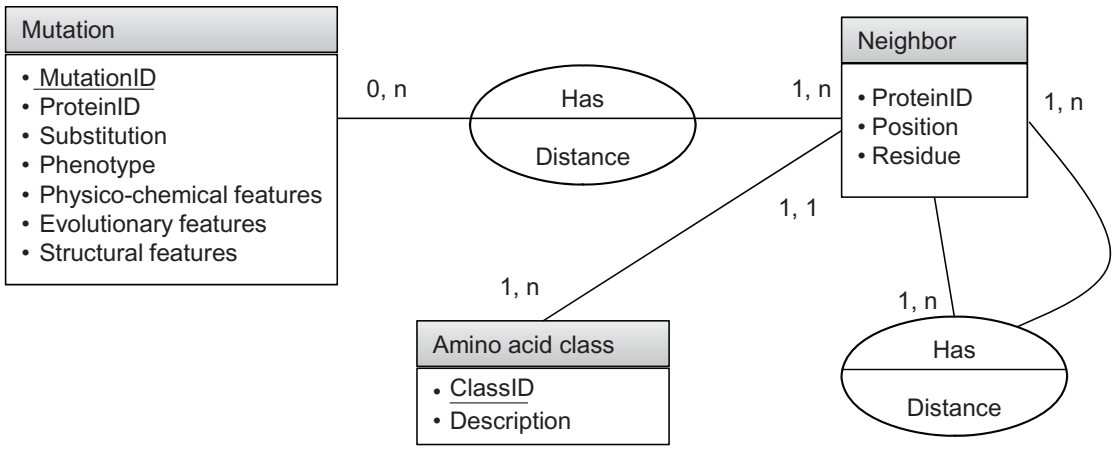
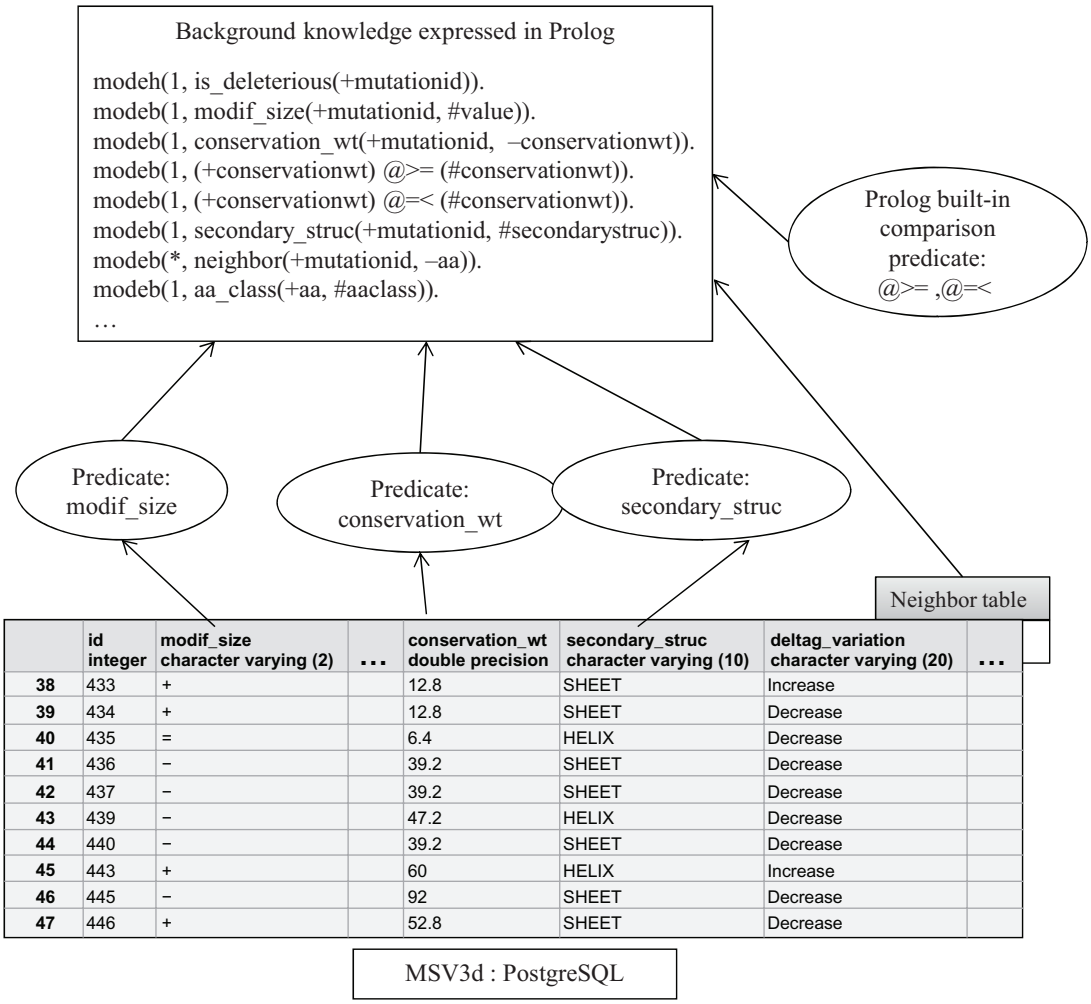


Figure 3. Mutation data model.
Notes: Each missense mutation is characterised by physico-chemical features (size, charge, polarity, hydrophobicity, etc), evolutionary information and 3D structural features. In addition, it may have one or more than one neighbouring residues, each of which can belong to a single class, based on Koolman's classification.



	id integer	modif_size character varying (2)	...	conservation_wt double precision	secondary_struc character varying (10)	deltag_variation character varying (20)	...
38	433	+		12.8	SHEET	Increase	
39	434	+		12.8	SHEET	Decrease	
40	435	=		6.4	HELIX	Decrease	
41	436	–		39.2	SHEET	Decrease	
42	437	–		39.2	SHEET	Decrease	
43	439	–		47.2	HELIX	Decrease	
44	440	–		39.2	SHEET	Decrease	
45	443	+		60	HELIX	Increase	
46	445	–		92	SHEET	Decrease	
47	446	+		52.8	SHEET	Decrease	

Figure 4. Construction of background knowledge from MSV3d.
Notes: Each mutation in the database is identified by a unique identifier 'id' and the values of each. Modeh defines the head of a hypothesised clause, while Modeb declares the predicates that can occur in the body of a hypothesised clause. The asterisk * in the mode declarations indicates that the corresponding predicate can be called many times during the construction of a hypothesised clause.

Table 1. Predicates used as background knowledge.

Type of information	Predicates	Description

Physico-chemical changes induced by the substitution	modif_size(+mutationid, #value)
	modif_charge(+mutationid, #value)
	modif_hydrophobicity(+mutationid, #value)
	modif_polarity(+mutationid, #value)
		Size, charge, polarity and hydrophobicity modifications

	g_p(+mutationid, #gp)	Glycine or proline loss or apparition
Evolutionary features	conservation_wt(+mutationid, –conservationwt)	Percentage of the wild type residue in the alignment column
	conservation_mut(+mutationid, –conservationmut)	Percentage of the mutant residue in the alignment column
	freq_at_pos(+mutationid, –freqatpos)	Number of known mutations at this position
	cluster_5res_size(+mutationid, –cluster5ressize)	Number of mutations at a distance of less than 5 residues in the sequence
Structural features	secondary_struc(+mutationid, #secondary_struc)	Secondary structure element (helix, sheet, no)
	gain_contact(+mutationid, –gaincontact) lost_contact(+mutationid, –lostcontact) identical_contact(+mutationid, –identicalcontact)	Contacts between – the wild type residue and its direct 3D neighbours, based on the wild type 3D model – the mutant residue and its direct 3D neighbours, based on the mutant 3D model are computed and compared
	gain_n1_contact(+mutationid, –gainn1contact) lost_n1_contact(+mutationid, –lostn1contact) identical_n1_contact(+mutationid, –identical_n1_contact)	Contacts between – residues in contact with the wild type residue and their direct 3D neighbours, based on the wild type 3D model – residues in contact with the mutant residue and their direct 3D neighbours, based on the mutant 3D model are computed and compared
	wt_accessibility(+mutationid, –wtacc) mut_accessibility(+mutationid, –mutacc)	Accessibility of the wild type/mutant residue
	cluster3d_10(+mutationid, –cluster3d10) cluster3d_20(+mutationid, –cluster3d20) cluster3d_30(+mutationid, –cluster3d30)	Number of mutations in the 3D cluster at 10, 20 and 30 A°
	stability_decrease(+mutationid) stability_increase(+mutationid) reliability_deltag(+mutationid, –reliabilitydeltag)	The change in protein relative stability upon mutation

- Combine the facts to generate a clause and use an evaluation function to estimate the score of the clause on examples
- Add the clause with the best score to the current hypothesis
- Remove positive examples covered by the best clause

These steps are iterated until all the positive examples are covered.

Aleph allows customization of all the parameters involved in the learning task. In our experiments, we used the default settings for all parameters, except for three important ones. First, the parameter minpos, indicating the minimum number of positive examples to be covered by an acceptable clause, was set to 5. Second, we set the parameter nodes to 50,000 (default 5,000), in order to provide a larger default search space. The nodes parameter defines

the maximum number of nodes in the search space to be explored by the algorithm. Finally, the parameter with the largest effect on the final results was the noise, defined as the maximum number of negative examples to be covered by an acceptable clause. To estimate this parameter, we tuned the model with different values. Six noise values (0.5%, 0.75%, 1%, 2%, 3% and 4% of negative examples) were tested and the optimal value, ie, the value which resulted in the best performance on our data sets, was then used for the final model.

In order to perform a stringent evaluation, we conducted a k-fold cross validation test. In this type of validation, the data set is randomly split into k equally sized subsets. The learning algorithm is then trained and tested k times. Each time, k-1 subsets are combined for training and the remaining one is used as the test set.

In this study, we used sensitivity (Se) and specificity (Sp) to evaluate the performance of our learning system:

$$Se = \frac{TP}{TP + FN}$$

$$Sp = \frac{TN}{TN + FP}$$

where True Positives (TP) and True Negatives (TN) are the number of correct predictions of the positive and negative examples, respectively, False Positives (FP) is the number of negative examples incorrectly predicted as positive, and False Negatives (FN) is the number of positive examples incorrectly predicted as negative.

As described earlier, we have an imbalanced data set (6,480 deleterious mutations and 1,637 neutral mutations). We therefore also considered the additional quality measure that is suitable for class imbalance learning, the Adjusted Geometric-mean[46] defined as:

$$Gmean = \sqrt{Se * Sp}$$

$$AGmean = \frac{(Gmean + Se * Np)}{1 + Np}$$

where Np is the proportion of positive (majority) examples in the dataset.

Results and Discussion

Novel hypotheses related to monogenic diseases

A set of mutations known to be involved in human monogenetic diseases was identified by performing keyword searches in the OMIM database. These mutations were then mapped to the MSV3d database and pertinent structural, functional and evolutionary features were extracted and formalized as Prolog facts, as described in the Materials and Methods section. The examples of 6,480 deleterious mutations and 1,637 neutral mutations constituted the initial knowledge base. We then used the ILP system implemented in the Aleph program to learn rules that define whether a mutation is 'deleterious' or 'neutral', based on its structural, functional and evolutionary characteristics. Table 2 shows the average sensitivity, specificity and AGmean for 3 fold cross-validation for each of the noise parameter values tested. As expected, the specificity decreases with increasing noise value, indicating that more negative examples are covered by the final set of rules. The best performance (with maximum AGmean value) was obtained by setting noise to 0.5%. A total of 173 rules were learned (decrypthon.igbmc.

Table 2. Results of 3 fold cross-validation for comparison between different values of the noise parameter.

Parameter	Sensitivity (%)	Specificity (%)	AGmean (%)
Noise = 0.5%	87.97	50.89	76.25
Noise = 0.75%	87.79	50.89	76.10
Noise = 1%	89.34	46.97	75.65
Noise = 2%	90.52	45.26	75.72
Noise = 3%	92.08	42.08	75.47
Noise = 4%	91.64	43.67	75.83

Note: Gmean = geometric mean of accuracies.

fr/kd4v/cgi-bin/rules), clearly indicating that a large number of factors are involved in the potential pathogenicity of a mutation.

Figure 5 shows some examples of the inferred rules, as presented on the web site. The rules are ranked according to their 'utility' score, defined as P—N, where P, N are the number of positive and negative examples covered by the rule. By clicking on the first column, the user can obtain a list of all mutations covered by each rule. The second column contains the rule identification number. This information is used only to identify the rules in our experiments. The two next columns contain the most important information: the "if" and "then" clauses of the induced rules. The two rightmost columns indicate the number of positive examples (deleterious mutations) and negative examples (neutral mutations) covered by the if-then rule in each row. A filter is available to facilitate the exploration, validation and interpretation of the rules.

To illustrate how to transform ILP rules (expressed in the Prolog language) into English sentences, we can consider the fourth rule in Figure 5 (mutation67_97),

is_deleterious(A):-
conservation_class(A, sub_family_conservation)
and *secondary_struc*(A, no_helix_no_sheet)
and *gain_contact*(A, B) and B>=1
and *stability*(A, decrease)

This rule states that a mutation A is deleterious if:

- The mutated residue belongs to the "sub-family conservation class"
- The residue is found in neither an α-helix, nor a β-sheet
- The number of contacts gained after point mutation is larger than or equal to 1
- The stability of the protein after point mutation is decreased

This rule correctly identified 111 deleterious mutations, while misclassifying 7 neutral mutations as deleterious.

Human exploration of the rule set

In order to facilitate the human interpretation of the rules, the individual rules were grouped into rule subfamilies using the hclust library in R (cran.r-project.org), which performs a hierarchical cluster analysis based on similarity measures between individuals. In our case, the similarity between 2 rules was defined as the number of common deleterious mutations covered by these 2 rules (Jaccard similarity coefficient[47]). The result of the clustering can be represented by a dendrogram (Fig. 4 and http://decrypthon.igbmc.fr/kd4v/download/mono67_dendrogram.pdf) displaying the hierarchical relationships between rules.

In a manual examination of complete dendrogram, two interesting rule subfamilies were identified which together encompass more than 558 deleterious mutations (8.6% of the dataset):

i. the subfamily containing the 4 rules: 67_96, 67_140, 67_58 and 67_210
is_deleterious(A):-
conservation_class(A, sub_family_conservation)
and
modif_charge(A, charge_opposite) and

	Id	If Statement	Then	Coverage Positive	Coverage Negative	Rank
		Enter a key word: sub_family_conservatio Submit				
⊞	mono67_123	conservation_class(A, sub_family_conservation) and freq_at_pos(A, B) and B>=2 and identical_n1_contact(A, C) and C>=20.	deleterious(A)	226 (3.49%)	0 (0.0%)	25
⊞	mono67_140	modif_size(A, size_increase) and modif_charge(A, charge_opposite) and conservation_class(A, sub_family_conservation) and identical_n1_contact(A, B) and B>=37.	deleterious(A)	126 (1.94%)	5 (0.31%)	80
⊞	mono67_85	conservation_class(A, sub_family_conservation) and conservation_wt(A, B) and B>=51.67 and gain_n1_contact(A, C) and C>=5.	deleterious(A)	111 (1.71%)	3 (0.18%)	92
⊞	mono67_97	conservation_class(A, sub_family_conservation) and secondary_struc(A, no_helix_no_sheet) and gain_contact(A, B) and B>=1 and stability(A, decrease).	deleterious(A)	111 (1.71%)	7 (0.43%)	97
⊞	mono67_58	modif_charge(A, charge_opposite) and modif_score(A, B) and B>=27 and conservation_class(A, sub_family_conservation) and secondary_struc(A, helix).	deleterious(A)	109 (1.68%)	8 (0.49%)	98
⊞	mono67_41	modif_polarity(A, polarity_decrease) and conservation_class(A, sub_family_conservation) and stability(A, decrease).	deleterious(A)	87 (1.34%)	6 (0.37%)	112

Figure 5. Part of a screenshot with four induced rules obtained using Aleph with noise = 0.5%, minpos = 5, nodes = 50,000.
Notes: Users can click on the + icon to see the covered examples. The keyword "sub_family_conservation" was used as a filter in this screenshot.

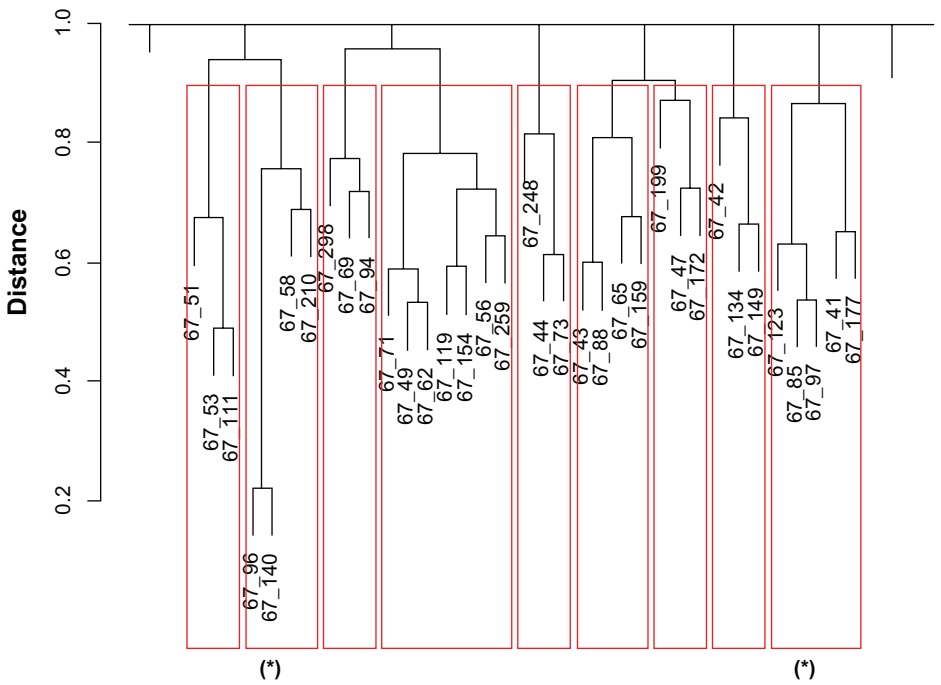


Figure 6. Part of the clustering of the full set of 173 generated rules.
Notes: We performed rule alignment on each subfamily (indicated by red rectangles in the dendrogram). Two interesting rules are indicated by (*).

modif_size(A, size_increase) and (*identical_n1_contact*(A, B) and B>=45)
is_deleterious(A):-
conservation_class(A, sub_family_conservation) and
modif_charge(A, charge_opposite) and *modif_size*(A, size_increase) and
(*identical_n1_contact*(A, B) and B>=37)
is_deleterious(A):-
conservation_class(A, sub_family_conservation) and
modif_charge(A, charge_opposite) and (*modif_score(*A, B) and B>=27) and *secondary_struc*(A, helix)
is_deleterious(A):-
conservation_class(A, sub_family_conservation) and *secondary_struc*(A, helix) and
(*cluster3d_10*(A, B) and B>=2)

ii. the subfamily containing the 5 rules: 67_123, 67_85, 67_97, 67_41 and 67_177
is_deleterious(A):-
conservation_class(A, sub_family_conservation) and
(*identical_n1_contact*(A, C) and C>=20) and
(*freq_at_pos(*A, B) and B>=2)
is_deleterious(A):-
conservation_class(A, sub_family_conservation) and
(*gain_n1_contact*(A, C) and C>=5) and
(*conservation_wt*(A, B) and B>=51.67)
is_deleterious(A):-
conservation_class(A, sub_family_conservation) and
(*gain_contact*(A, B) and B>=1) and
secondary_struc(A, no_helix_no_sheet) and *stability*(A, decrease)
is_deleterious(A):-
conservation_class(A, sub_family_conservation) and
modif_polarity(A, polarity_decrease) and *stability*(A, decrease)
is_deleterious(A):-
conservation_class(A,sub_family_conservation) and (*identical_contact*(A, B) and B>=8) and *modif_charge*(A, charge_decrease) and
(*wt_accessibility*(A, C) and C>=10.31).

The "sub_family_conservation" predicate was found in all the rules in these two subfamilies and was highly predictive for the deleterious state. This supports a well known hypothesis that conserved

positions in a protein are more likely to be functional and thus, that mutations of conserved residues are often deleterious. Nevertheless, the fact that the rules are complex and integrate other factors indicates that residue conservation alone does not determine the effect of the mutation.

As an example, the first subfamily highlights another important factor, represented by the "modif_charge" predicate, which characterizes many deleterious mutations. In this case, the value of "charge_opposite" indicates a change in the residue charge from positive to negative or vice versa. This result confirms recent findings concerning the conservation of the mutated residue and the alteration of the chemical and physical properties of the amino acids in a missense variant having a crucial effect on protein function.[48]

The second subfamily reveals the role played by the 3D context of the mutated residue in pathogenicity. The rules in this subfamily suggest that conserved wild type residues with a large number of contacts with other residues in the 3D structure of the protein are more likely to be deleterious. A similar effect is seen with a gain in the number of contacts by the mutant residue.

In order to further facilitate the validation and the interpretation of the discovered knowledge, we also calculated the frequencies of predicates in the set of rules. We then ordered the predicates from the most predictive to the least predictive. The top five predicates are listed in Table 3. In addition to conservation features, the secondary structure element is identified as an important factor.[28,33]

Prediction service

Based on the rules learnt by the ILP algorithm described above, a function aimed at estimating nsSNP effects related to human monogenic disease has been added in the KD4v server. It can be accessed via the Prediction link on the KD4v web interface in the main menu. KD4v allows users to specify the amino acid position and substitution of a given protein to be predicted. This includes the Uniprot accession number of the protein, the mutation position, the wild type residue, the mutant residue and the knowledge base (proteins related to human monogenic disease) to be used for rule inference. The wild type residue must correspond to the current protein sequence.

Given the input mutation, KD4v sends a request to MSV3d to automatically generate a multi-level characterization. The process starts with the generation of mutant 3D models. Then, physico-chemical changes and structural modifications induced by the substitution, as well as functional and structural features related to the mutated position are calculated. If a 3D model is available, these values are converted into Prolog facts, which then become the input for the prediction engine of KD4v.

The prediction engine was built based on the rules inferred in the previous section. Using Prolog, the deductive reasoning process immediately derives a conclusion (deleterious or neutral mutation).

Table 3. The top five predicates found in the rules defining deleterious or neutral mutations.

Predicates	Frequency in set of rules
secondary_struc	12.7%
conservation_class	11.0%
modif_charge	7.6%
cluster_5res_size	6.3%
conservation_wt	6.1%

Conclusions

The ILP approach is consistent with the database perspective where KDD is organized around queries aimed at either selecting datasets for mining, or transforming these datasets, or querying and exploring large sets of patterns extracted from the data. This study presents a novel application of ILP in the bioinformatics field, namely, the characterization of the effects of a mutation on protein function and the corresponding human monogenic disease. Using MSV3d, a database of annotated mutation and phenotypic data, we identified mutations related to human monogenic disease and constructed background knowledge and a training set of positive and negative examples. The resulting mutation knowledge base contains a set of rules for distinguishing deleterious and neutral mutations. The rules confirmed previous findings concerning the physico-chemical and evolutionary features that characterize a deleterious mutation, such as the importance of the conservation of the mutated residue or the detrimental effects of modification of the amino acid charge, volume and hydrophobicity.

An important feature of our system is the fact that almost all the mutations can be easily accessed via their associated set of rules. Our mutation knowledge base thus provides useful information for understanding the relationships between the genotypic alteration and the phenotypic features in human monogenic diseases. The knowledge discovered by ILP should be helpful for the design of further research experiments. In addition, we have shown that the ILP approach can be effectively used for mutation effect prediction, as illustrated by the performances obtained which are similar to the common and widely used methods.

In the future, we plan to enhance the background knowledge by including more detailed genotypic and phenotypic information, as well as additional data related to the 3D structure, including structural surface topology descriptions.[49] We will also include more functional information, for example log-odds scores based on the Gene Ontology.[50] By integrating richer and more relevant background knowledge, we hope to not only improve the classification of deleterious from neutral mutations, but also to shed light on the complex relationships that exist between phenotype and genotype. In the longer term, these developments should contribute to a more complete elucidation of the chain of events leading from a molecular defect to its pathology.

Author Contributions

HN and TDL conceived and designed the experiments. TDL, JDT and OP analysed the data. HN, TDL, JDT and OP contributed to the writing of the manuscript. All authors reviewed and approved of the final manuscript.

Competing Interests

Author(s) disclose no potential conflicts of interest.

Funding

The work was performed within the framework of the Decrypthon program, co-funded by Association Française contre les Myopathies [AFM, 14390-15392]; IBM and Centre National de la Recherche Scientifique (CNRS); ANR [Puzzle-Fit: 09-PIRI-0018-02, BIPBIP: ANR-10-BINF-03-02: FRISBI: ANR-10-INSB-05-01]; Institute funds from the CNRS, INSERM, the Université de Strasbourg and the Vietnam Ministry of Education and Training (CT 322).

Acknowledgements

The Institut de Génétique et de Biologie Moléculaire et Cellulaire (IGBMC) services are acknowledged for assistance.

Disclosures and Ethics

As a requirement of publication author(s) have provided to the publisher signed confirmation of compliance with legal and ethical obligations including but not limited to the following: authorship and contributorship, conflicts of interest, privacy and confidentiality and (where applicable) protection of human and animal research subjects. The authors have read and confirmed their agreement with the ICMJE authorship and conflict of interest criteria. The authors have also confirmed that this article is unique and not under consideration or published in any other publication, and that they have permission from rights holders to reproduce any copyrighted material. Any disclosures are made in this section. The external blind peer reviewers report no conflicts of interest.

References

1. Stranger BE, Forrest MS, Dunning M, et al. Relative impact of nucleotide and copy number variation on gene expression phenotypes. *Science*. 2007; 315(5813):848–53.
2. Chasman D, Adams RM. Predicting the functional consequences of non-synonymous single nucleotide polymorphisms: structure-based assessment of amino acid variation. *J Mol Biol*. 2001;307(2):683–706.
3. Cochrane GR, Galperin MY. The 2010 nucleic scids research database issue and online database collection: a community of data resources. *Nucleic Acids Res*. 2010;38(Database issue):D1–4.
4. Thusberg J, Olatubosun A, Vihinen M. Performance of mutation pathogenicity prediction methods on missense variants. *Hum Mutat*. 2011;32(4):358–68.
5. Ng PC, Henikoff S. SIFT: Predicting amino acid changes that affect protein function. *Nucleic Acids Res*. 2003;31(13):3812–4.
6. Thomas PD, Campbell MJ, Kejariwal A, et al. PANTHER: a library of protein families and subfamilies indexed by function. *Genome Res*. 2003;13(9): 2129–41.
7. Ferrer-Costa C, Gelpi JL, Zamakola L, Parraga I, de la Cruz X, Orozco M. PMUT: a web-based tool for the annotation of pathological mutations on proteins. *Bioinformatics*. 2005;21(14):3176–8.
8. Capriotti E, Calabrese R, Casadio R. Predicting the insurgence of human genetic diseases associated to single point protein mutations with support vector machines and evolutionary information. *Bioinformatics*. 2006;22(22): 2729–34.
9. Bromberg Y, Rost B. SNAP: predict effect of non-synonymous polymorphisms on function. *Nucleic Acids Res*. 2007;35(11):3823–35.
10. Chun S, Fay JC. Identification of deleterious mutations within three human genomes. *Genome Res*. 2009;19(9):1553–61.
11. Saunders CT, Baker D. Evaluation of structural and evolutionary contributions to deleterious mutation prediction. *J Mol Biol*. 2002;322(4):891–901.
12. Sunyaev S, Ramensky V, Bork P. Towards a structural basis of human non-synonymous single nucleotide polymorphisms. *Trends Genet*. 2000;16(5): 198–200.
13. Ramensky V, Bork P, Sunyaev S. Human non-synonymous SNPs: server and survey. *Nucleic Acids Res*. 2002;30(17):3894–900.

14. Bao L, Zhou M, Cui Y. nsSNPAnalyzer: identifying disease-associated nonsynonymous single nucleotide polymorphisms. *Nucleic Acids Res*. 2005;33(Web Server issue):W480–2.
15. Yue P, Melamud E, Moult J. SNPs3D: candidate gene and SNP selection for association studies. *BMC Bioinformatics*. 2006;7:166.
16. Masso M, Vaisman II. Accurate prediction of stability changes in protein mutants by combining machine learning with structure based computational mutagenesis. *Bioinformatics*. 2008;24(18):2002–9.
17. Adzhubei IA, Schmidt S, Peshkin L, et al. A method and server for predicting damaging missense mutations. *Nat Methods*. 2010;7(4):248–9.
18. Fayyad UM, Piatetsky-Shapiro G, Smyth P. From data mining to knowledge discovery: an overview. In: Fayyad UM, Piatsky-Shapiro G, Smyth P, Uthursamy R, editors. *Advances in Knowledge Discovery and Data Mining*. Palo Alto: American Association for Artificial Intelligence; 1996:1–34.
19. Džeroski S, Cussens J, Manandhar S. An Introduction to inductive logic programming and learning language in logic. *Learning Language in Logic*. 2000:707–51.
20. Džeroski S. Multi-relational data mining: an introduction. *SIGKDD Explor. Newsl*. 2003;5(1):1–16.
21. Woods RW, Oliphant L, Shinki K, Page D, Shavlik J, Burnside E. Validation of results from knowledge discovery: mass density as a predictor of breast cancer. *J Digit Imaging*. 2009.
22. Muggleton S, King RD, Stenberg MJE. Protein secondary structure prediction using logic-based machine learning. *Protein Engineering*. 1992;5(7): 647–57.
23. Cootes AP, Muggleton SH, Sternberg MJ. The automatic discovery of structural principles describing protein fold space. *J Mol Biol*. 2003;330(4): 839–50.
24. King RD. Applying inductive logic programming to predicting gene function. *AI Mag*. 2004;25(1):57–68.
25. Nguyen TP, Ho TB. An integrative domain-based approach to predicting protein-protein interactions. *J Bioinform Comput Biol*. 2008;6(6):1115–32.
26. Kelley LA, Shrimpton PJ, Muggleton SH, Sternberg MJ. Discovering rules for protein-ligand specificity using support vector inductive logic programming. *Protein Eng Des Sel*. 2009;22(9):561–7.
27. Ryeng E, Alsberg BK. Microarray data classification using inductive logic programming and gene ontology background information. *Journal of Chemometrics*. 2010;24(5):231–40.
28. Luu TD, Rusu A, Walter V, et al. KD4v: comprehensible knowledge discovery system for missense variant. *Nucleic Acids Res*. 2012;40:W71–5.
29. Audo I, Bujakowska K, Orhan E, et al. Whole-exome sequencing identifies mutations in GPR179 leading to autosomal-recessive complete congenital stationary night blindness. *Am J Hum Genet*. 2012;90(2):321–30.
30. Zeitz C, Jacobson SG, Hamel CP, et al. Whole exome sequencing identifies mutations in LRIT3 as a cause for autosomal recessive complete congenital stationary night blindness. *Am J Hum Genet*. 2013;92:67–75.
31. Luu TD, Rusu AM, Walter V, et al. MSV3d: database of human MisSense variants mapped to 3D protein structure. *Database (Oxford)*. 2012;2012(0): bas018.
32. Lecompte O, Thompson JD, Plewniak F, Thierry J, Poch O. Multiple alignment of complete sequences (MACS) in the post-genomic era. *Gene*. 2001;270(1–2):17–30.
33. Friedrich A, Garnier N, Gagnière N, et al. SM2PH-db: an interactive system for the integrated analysis of phenotypic consequences of missense mutations in proteins involved in human genetic diseases. *Human Mutation*. 2010;31(2):127–35.
34. Sherry ST, Ward MH, Kholodov M, et al. dbSNP: the NCBI database of genetic variation. *Nucleic Acids Res*. 2001;29(1):308–11.
35. Yip YL, Famiglietti M, Gos A, et al. Annotating single amino acid polymorphisms in the UniProt/Swiss-Prot knowledgebase. *Hum Mutat*. 2008;29(3): 361–6.
36. Fokkema IF, den Dunnen JT, Taschner PE. LOVD: easy creation of a locus-specific sequence variation database using an "LSDB-in-a-box" approach. *Hum Mutat*. 2005;26(2):63–8.
37. Muggleton S. Inductive logic programming. *New Generation Computing*. 1991;8(4):295–318.
38. Srinivasan A, King R, Muggleton S. The role of background knowledge: using a problem from chemistry to examine the performance of an ILP program. *Transactions on Knowledge and Data Engineering*. 1999.
39. Amberger J, Bocchini CA, Scott AF, Hamosh A. McKusick's Online Mendelian Inheritance in Man (OMIM). *Nucleic Acids Res*. 2009;37(Database issue): D793–6.
40. Kowarsch A, Fuchs A, Frishman D, Pagel P. Correlated mutations: a hallmark of phenotypic amino acid substitutions. *Plos Computat Biol*. 2010;6(9):–.
41. Koolman J, Boehm K. *Colour Atlas of Biochemistry*. New York: Thieme; 1996.
42. Quinlan JR, Cameron-Jones RM. FOIL: A Midterm Report. *Proceedings of the European Conference on Machine Learning*. London: Springer-Verlag; 1993.
43. Muggleton S. Inverse entailment and progol. *New Generation Computing*. 1995;13(3):245–86.
44. Blockeel H, Raedt LD. Top-down induction of first-order logical decision trees. *Artif Intell*. 1998;101(1–2):285–97.
45. Srinivasan A. The Aleph Manual. 2004; Available at: http://www.comlab.ox.ac.uk/activities/machinelearning/Aleph/. Accessed Feb 22, 2013.
46. Batuwita R, Palade V. A New Performance Measure for Class Imbalance Learning. Application to Bioinformatics Problems. Paper presented at: International Conference on Machine Learning and Applications; 2009.
47. Tan P-NS, Michael; Kumar, Vipin *Introduction to Data Mining*. Boston: Addison-Wesley; 2005.
48. Thusberg J, Vihinen M. Pathogenic or not? And if so, then how? Studying the effects of missense mutations using bioinformatics methods. *Hum Mutat*. 2009;30(5):703–14.
49. Albou LP, Poch O, Moras D. M-ORBIS: mapping of molecular binding sites and surfaces. *Nucleic Acids Res*. 2011;39(1):30–43.
50. Capriotti E, Altman RB. A new disease-specific machine learning approach for the prediction of cancer-causing missense variants. *Genomics*. 2011; 98(4):310–7.