**PhD Dissertation**

**International Doctorate School in Information and Communication Technologies**

DISI - University of Trento

# Statistical and relational learning for understanding enzyme function

Elisa Cilia

Advisor:
Prof. Andrea Passerini
Università degli Studi di Trento

December 2010

*To my parents, my sister Giulia and my grandmas.*
*Ai miei genitori, mia sorella Giulia e le mie nonne.*

# Abstract

*Unravelling the functioning of the complex processes involved in living systems is a challenging task. Enzymes are involved in almost all of the chemical processes taking place within the cell. They accelerate chemical reactions by forming a complex with the substrate and therefore lowering the reaction activation energy. The characterisation of the enzyme function at the molecular level is a fundamental step, which has several implications and applications in modern biotechnologies. This thesis investigates statistical and relational learning techniques for the characterisation of the enzyme function. The problem is tackled from two sides: the analysis of the enzyme structure and its interactions with other molecules, and the mining of relevant features from the enzyme mutation data. From the first side a pure statistical learning approach is proposed for directly predicting enzyme functional residues. This approach is shown to improve over the current state of the art on several benchmark datasets. The engineered predictors resulting from this investigation are now available to the public of researchers through the CatANalyst web server. Further improvement of the approach is pursued by proposing a supervised clustering technique for collectively predicting all the residues belonging to the same functional site. On the "learning from mutations" side, the focus shifts to the expressivity and interpretability of the learnt models. This thesis proposes novel statistical relational approaches for mining hierarchical features for multiple related tasks. The resistance of viral enzyme mutants to groups of related inhibitors is modelled in a multitask setting. Learnt models are refined on a group or per-task basis at different levels of the hierarchy. The* *proposed hierarchical approach is shown to provide statistically significant improvements over both single and multitask alternatives. Moreover it has the ability to provide explanation of the models which are themselves hierarchical. A task clustering approach is also proposed for inferring the structure of tasks when it is unknown. Finally, a relational approach is proposed for exploiting the learnt relational rules for generating novel mutations with specific characteristics. This allows to drastically reduce the space of possible mutations to be experimentally assessed. Promising preliminary results are obtained, which highlight the potential of the approach in guiding mutant engineering and in predicting the viral enzyme evolution. These findings can pave the way to further research directions in functional interpretation of biological data by means of machine learning techniques.*

**Keywords**

[machine learning, bioinformatics, protein function identification, inductive logic programming, statistical relational learning, enzyme functional sites]

# Acknowledgements

*I really wish to thank*

*first of all, my advisor Andrea Passerini, for guiding me throughout this joint work, advising me in every circumstance, being for me a scientific reference point for both machine learning and bioinformatics aspects, introducing me to the world of relational and statistical relational learning, giving me examples of intellectual honesty and of how scientific work is meant to be done, with his fresh and genuine commitment to research, and much more;*

*Mauro Brunato, for the scientific support in the early stages of the work and his unfailing friendship support along these four years, together with his family;*

*Sergio Ammendola, for introducing me to the world of protein bioinformatics and the protein functional characterisation, and for the fruitful collaborations that conferred further biological relevance to this thesis;*

*Niels Landwehr, for the precious collaboration in the development of the hierarchical extension to kFOIL;*

*Franco Mascia, not only for the scientific collaboration and bizarre discussions in research related issues but especially for everything else;*

*Alessandro Moschitti, for initiating me into the world of machine learning and for the precious collaboration;*

*the members of the thesis commission, who helped me in improving the PhD dissertation with invaluably helpful comments;*

*the unknown journal and conferences reviewers, who contributed in the improvement of my research work;*

*people providing me with great teachings, who I had the pleasure to meet in schools and courses during my PhD: Anna Tramontano, Alex Smola, Berhnard Schölkopf and Nello Cristianini;*

*friends, people gravitating the Learning and Intelligent Optimization group during this PhD course, and bioinformaticians at the Foundation Edmund Mach for the support in the last months of preparation of the PhD dissertation;*

*my family and my sister, far away from me in these years, to whom this thesis is dedicated; my relatives always supporting me.*

# Contents

**1 Introduction** **1**

1.1 Motivations . . . . . . . . . . . . . . . . . . . . . . . . 2

1.2 The Protein Function Identification Problem . . . . . . . . 9

1.2.1 Quick Primer on the Biological Aspects . . . . . . . 9

1.2.2 Defining a Protein Function . . . . . . . . . . . . . 13

1.3 Traditional Approaches to Protein Function Identification . 17

1.4 Review on Machine Learning Techniques . . . . . . . . . . 19

1.4.1 Learning Paradigms and Learning Tasks . . . . . . 19

1.4.2 Primer on Support Vector and Kernel Machines . . 23

1.4.3 Primer on ILP and Statistical Relational Learning . 31

1.4.4 Measuring and Comparing the Performance of Learning Algorithms . . . . . . . . . . . . . . . . . . . 36

1.5 Aspects of Innovation . . . . . . . . . . . . . . . . . . . . 37

1.6 Structure of the Thesis . . . . . . . . . . . . . . . . . . . 41

**I Functional Residue Prediction** **43**

**2 Identifying Functionally Important Residues** **45**

2.1 Introduction and Background . . . . . . . . . . . . . . . . 45

2.2 State of the Art . . . . . . . . . . . . . . . . . . . . . . . 51

**3 A Support Vector Learning Approach 57**
3.1 The Learning Task . . . . . . . . . . . . . . . . . . . . . . 58
3.2 Engineering Residue Features . . . . . . . . . . . . . . . . . 58
3.2.1 Features Extracted from the Sequence . . . . . . . 58
3.2.2 Modelling a Residue Structural Neighbourhood . . 59
3.2.3 Features Extracted from the Structure . . . . . . . 62
3.3 Experimental Results . . . . . . . . . . . . . . . . . . . . . 68
3.3.1 Datasets . . . . . . . . . . . . . . . . . . . . . . . . 68
3.3.2 Experimental Setting . . . . . . . . . . . . . . . . . 69
3.3.3 Results of Different Feature Sets . . . . . . . . . . . 70
3.3.4 Comparison with Other Methods . . . . . . . . . . 77
3.4 Engineering Active Site Features: Kernels on Structures . . 80
3.4.1 3D Kernel on Shapes . . . . . . . . . . . . . . . . . 80

**4 Toward a Collective Learning Approach 93**
4.1 Problem Description and Formalisation . . . . . . . . . . . 94
4.2 Distance-based Supervised Clustering with Maximum-weight Cliques . . . . . . . . . . . . . . . . . . . . . . . . . . . . 98
4.3 The Maximum-weight Clique Clustering Algorithm . . . . 99
4.4 Experimental Results . . . . . . . . . . . . . . . . . . . . . 103
4.4.1 Predicting Active Sites . . . . . . . . . . . . . . . . 103
4.4.2 Predicting Geometry of Metal Binding Sites . . . . 106

**5 CatANalyst - Catalytic Residue Predictor 109**
5.1 CatANalyst Web Server Architecture . . . . . . . . . . . . 109
5.1.1 The Elaboration Engine . . . . . . . . . . . . . . . 111
5.2 Querying CatANalyst . . . . . . . . . . . . . . . . . . . . . 112
5.2.1 Job Submission . . . . . . . . . . . . . . . . . . . . 112
5.2.2 Job Elaboration . . . . . . . . . . . . . . . . . . . . 114
5.2.3 Result Visualisation . . . . . . . . . . . . . . . . . 116

5.3 Performance Evaluation . . . . . . . . . . . . . . . . . . . . . 116

5.4 Statistics . . . . . . . . . . . . . . . . . . . . . . . . . . . 118

5.5 Computational Time Issues . . . . . . . . . . . . . . . . . . . 118

5.6 Work In Progress . . . . . . . . . . . . . . . . . . . . . . . . 119

**II Mining Relevant Features from Mutation Data 121**

**6 Learning from Mutations 123**

6.1 Introduction and Background . . . . . . . . . . . . . . . . . . 123

6.2 State of the Art . . . . . . . . . . . . . . . . . . . . . . . . . 130

**7 Hierarchical Multitask kFOIL 133**

7.1 Introduction . . . . . . . . . . . . . . . . . . . . . . . . . . . 134

7.2 kFOIL: Learning Relational Kernels . . . . . . . . . . . . . . 135

7.2.1 The kFOIL Algorithm . . . . . . . . . . . . . . . . . . 135

7.2.2 Multitask kFOIL . . . . . . . . . . . . . . . . . . . . . 140

7.3 Hierarchical kFOIL . . . . . . . . . . . . . . . . . . . . . . . 141

7.4 Task Clustering . . . . . . . . . . . . . . . . . . . . . . . . . 143

7.5 Experimental Evaluation . . . . . . . . . . . . . . . . . . . . 146

7.5.1 Predicting Drug-Resistance of Mutants . . . . . . . 147

7.5.2 Protein Structure Classification . . . . . . . . . . . 159

**8 A Relational Learning Approach Toward Protein Engineering 167**

8.1 Algorithm Overview . . . . . . . . . . . . . . . . . . . . . . . 169

8.1.1 Background Knowledge . . . . . . . . . . . . . . . . . 169

8.1.2 The Approach . . . . . . . . . . . . . . . . . . . . . . 171

8.2 Experimental Evaluation . . . . . . . . . . . . . . . . . . . . 174

8.2.1 Dataset . . . . . . . . . . . . . . . . . . . . . . . . . 175

8.2.2 Performance Evaluation . . . . . . . . . . . . . . . . . 175
8.3 Discussion . . . . . . . . . . . . . . . . . . . . . . . . . . . . 177
8.4 Future Work . . . . . . . . . . . . . . . . . . . . . . . . . . . 179

**9 Conclusions 183**

**Bibliography 189**

# List of Figures

1.1 Growth of UniprotKB released sequences. . . . . . . . . . 4
1.2 Growth of PDB released protein structures per year. . . . 5
1.3 The Central Dogma of molecular biology. . . . . . . . . . . 10
1.4 Amino acid. . . . . . . . . . . . . . . . . . . . . . . . . . 11
1.5 Peptide bond formation. . . . . . . . . . . . . . . . . . . . 13
1.6 Structure of an $\alpha$-helix and a $\beta$-sheet. . . . . . . . . . . . 13
1.7 The levels of a protein structure . . . . . . . . . . . . . . . 14
1.8 Enzyme Classification number. . . . . . . . . . . . . . . . 16
1.9 Pie chart of the results of the “enzyme” query to the UniprotKB. . . . . . . . . . . . . . . . . . . . . . . . . . . 17
1.10 Supervised learning. . . . . . . . . . . . . . . . . . . . . 21
1.11 Soft margin SVM. . . . . . . . . . . . . . . . . . . . . . . 27
1.12 Mapping in a new feature space. . . . . . . . . . . . . . . 28

2.1 Active site schematisation. . . . . . . . . . . . . . . . . . . 47
2.2 Roles of metal ions in enzymes. . . . . . . . . . . . . . . . 49
2.3 Three different proposed approaches for active site identification. . . . . . . . . . . . . . . . . . . . . . . . . . . 50

3.1 A residue 3D representation. . . . . . . . . . . . . . . . . 60
3.2 The L-arginine:glycine amidinotransferase (1JDW) and its highlighted catalytic pocket. . . . . . . . . . . . . . . . . 61
3.3 A residue structural neighbourhood. . . . . . . . . . . . . 62

3.4 Example of feature vector extracted from the three-dimensional neighbourhood of a catalytic residue. . . . . . . . . . . . . 83
3.5 Histogram of the frequencies of heterogen molecules in the PW dataset. . . . . . . . . . . . . . . . . . . . . . . . . . 84
3.6 Histograms of the distances of the most frequent heterogens from catalytic residues in the PW dataset. . . . . . . . . . 85
3.7 Vector of the feature weights $\vec{w}$ of a classifier trained on the PW dataset. . . . . . . . . . . . . . . . . . . . . . . . . . 86
3.8 ROC and Recall/Precision curves . . . . . . . . . . . . . . 87
3.9 Local ROC curves of the predictions on different benchmarking datasets . . . . . . . . . . . . . . . . . . . . . . . . . 88
3.10 Global ROC curves of the predictions on different benchmarking datasets . . . . . . . . . . . . . . . . . . . . . . . 89
3.11 Recall/Precision curves of the predictions on different benchmarking datasets . . . . . . . . . . . . . . . . . . . . . . . 90
3.12 ROC curves superimposed with those reported in [134] on the PW dataset. . . . . . . . . . . . . . . . . . . . . . . . . 91
3.13 Two examples of triangular shapes extracted from the HYS303 three-dimensional neighbourhood. . . . . . . . . . . . . . 91

4.1 Histogram of the catalytic propensities of the residues in the experimental dataset *HA superfamily* . . . . . . . . . . . 95
4.2 Metal binding sites in the sequence of the equine herpes virus-1. . . . . . . . . . . . . . . . . . . . . . . . . . . . 96
4.3 Functional residues in the sequence of the cloroperoxidase T. 97

5.1 CatANalyst web server architecture. . . . . . . . . . . . . 110
5.2 Input form for the sequence-based CatANalyst prediction. 113
5.3 Input form for the structure-based CatANalyst prediction. 114
5.4 Intermediate elaboration page showing the job status. . . . 115

5.5 Web page showing the results of the CatANalyst predictions. 115
5.6 Results of experiments for the CatANalyst online predictors 117

6.1 Correlation plot of the prediction margins of the residues in the wild type enzyme (x axis) and one of its mutants (y axis). 125
6.2 Portion of the predicted 3D model of SsAH including the active site. . . . . . . . . . . . . . . . . . . . . . . . . 126
6.3 Reverse transcriptase inhibition by Nevirapine (PDB code 1VRT). . . . . . . . . . . . . . . . . . . . . . . . . 128

7.1 Example of a relational kernel function for the HIV resistance mutation database. . . . . . . . . . . . . . . . . . . . 138
7.2 The hierarchical learning approach. . . . . . . . . . . . . . 141
7.3 Example hierarchy of learnt clauses. . . . . . . . . . . . . . 153
7.4 Hierarchical clustering dendrograms. . . . . . . . . . . . . 156
7.5 Relative frequency of the background knowledge predicates in the learnt models. . . . . . . . . . . . . . . . . . . . . 163
7.6 Example of EF-hand like protein structure. . . . . . . . . . 165

8.1 Schema of the mutation engineering algorithm. . . . . . . . 172
8.2 Example of learnt hypothesis for the NCRTI task with highlighted mutations covered by the hypothesis clauses. . . . . 173
8.3 Mean recall by varying the number of generated mutations. 178
8.4 Detail of the mean recall curves of the NNRTI task for a number of generated mutations below 150. . . . . . . . . . 179
8.5 Example of application of the approach for mutant engineering. . . . . . . . . . . . . . . . . . . . . . . . . . . 181

In this thesis some of the molecular graphics images or parts of them were produced using the UCSF Chimera package [112] from the Resource for Biocomputing, Visualization, and Informatics at the University of Cal-

ifornia, San Francisco (supported by NIH P41 RR-01081).

# List of Tables

1.1 Amino acid alphabets. . . . . . . . . . . . . . . . . . . . . . 12

3.1 Sequence-based Features Representation: features extracted from the protein sequence . . . . . . . . . . . . . . . . . . . 58
3.2 Structured-based Features Representation: scalar features extracted from the residue structural neighbourhood. . . . 63
3.3 Classification of the heterogens into three groups. . . . . . 67
3.4 Legend of abbreviations for the different sets of attributes . 70
3.5 Heterogen Analysis . . . . . . . . . . . . . . . . . . . . . 72
3.6 Statistical analysis . . . . . . . . . . . . . . . . . . . . . 73
3.7 Comparison with state-of-the-art sequence-based approach [151] 77
3.8 Comparison with structure-based approaches by *Chea et al.* [37] and *Youn et al.* [149] . . . . . . . . . . . . . . . . . . 78
3.9 Comparison with structure-based approach by Tang et al. . 79
3.10 Comparison with the POOL structure-based approach . . . 80

4.1 Comparison of the results obtained in active site prediction with the distance-based supervised learning approach . . . 105
4.2 Comparison on the metalloproteins dataset . . . . . . . . . 108
4.3 Experimental results on the metalloproteins dataset . . . . 108

5.1 CatANalyst's scheduler policy and job queues. . . . . . . . 111
5.2 results for test proteins in PDBselect. . . . . . . . . . . . . 118

7.1 Hierarchical task structure for the HIV resistance mutation datasets with associated number of instances. . . . . . . . 149
7.2 Summary of the HIV drug resistance background knowledge facts and rules. . . . . . . . . . . . . . . . . . . . . . . . . 149
7.3 Amino acid types encoded in colour classes. . . . . . . . . . 150
7.4 Summary of the hierarchical multitask experiments . . . . 152
7.5 Summary of the hierarchical clustering experiments . . . . 158
7.6 Number of examples (N) in each of the folds in the SCOP dataset. . . . . . . . . . . . . . . . . . . . . . . . . . . . 160
7.7 Summary of the SCOP background knowledge. . . . . . . . 161
7.8 Summary of the hierarchical multitask experiments for the SCOP dataset. . . . . . . . . . . . . . . . . . . . . . . . . 162

8.1 Summary of the background knowledge facts and rules. . . 170
8.2 Statistical comparisons of the performance of the algorithm with an algorithm generating mutations at random . . . . 176
8.3 Example of learnt model. . . . . . . . . . . . . . . . . . . . 180

# List of Algorithms

4.1 WMC main procedure. . . . . . . . . . . . . . . . . . . . . . . 101
4.2 The ChooseNode procedure. . . . . . . . . . . . . . . . . . . . 103
7.1 kFOIL algorithm. . . . . . . . . . . . . . . . . . . . . . . . . 137
7.2 Hierarchical kFOIL algorithm. . . . . . . . . . . . . . . . . . . 142
7.3 kFOIL refinement algorithm. . . . . . . . . . . . . . . . . . . . 143
7.4 Hierarchical clustering kFOIL algorithm. . . . . . . . . . . . . 146
7.5 kFOIL algorithm for cluster node. . . . . . . . . . . . . . . . . 146
8.1 Algorithm for novel relevant mutations discovery. . . . . . . 174

# Chapter 1

# Introduction

The beginning of the genomic era made it possible for scientists to compare the genome sequences of different organisms and species. These genome sequences can have different size and complexities depending on the organism they belong to, from viruses and bacteria, to plants and mammals.

The availability of the human genome sequence has emphasised how far we still are from understanding the relationship between the structure of the molecules and their function.

By analysing the human genome and genomes of simpler organisms researchers observed that the number of genes is not proportional to the size or the complexity of an organism. This observation led to the paradox that, for instance, the human genome contains some tens of thousands of genes [110], encoding approximately the same number of well characterised proteins, but apparently, the number of protein functions seems to be higher. This sort of paradox can be explained by hypothesising that one gene codes for more than one protein (an example is given by the alternative splicing mechanism) and that the same protein is able to perform more than one function. Hence, identifying the function or the functions that a protein performs is one of the big challenges characterising the post-genomic era.

Small variations of the same protein can be present within different organs of the same organism and between different species. This makes the protein function identification still more challenging and the analysis and mining of such a large quantity of biological data requires the use of automated approaches.

Computational methods can play a key role in "omics" projects aiming at giving a functional interpretation to the information contained in the genomes. The birth of a discipline like bioinformatics is the evidence of the importance of computational approaches in genomics and proteomics projects.

## 1.1 Motivations

*The goal of protein bioinformatics is to assist experimental biology in assigning a function or suggesting functional hypothesis for all known proteins. The task is formidable. A simple calculation shows that we cannot possibly study each and every biological molecule of the universe. Therefore, we need fast and reliable computational methods to extrapolate the knowledge accumulated on a subset of cases to the rest of the protein universe.*

*Anna Tramontano in "The Ten Most Wanted Solutions in Protein Bioinformatics"*

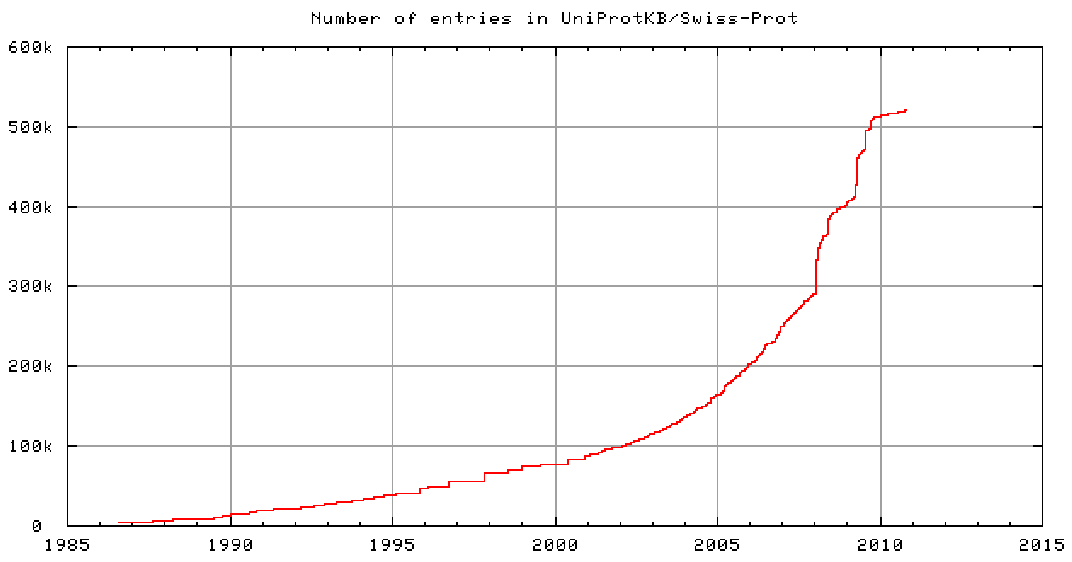
Bioinformatics is a recently born discipline. Its aim is to analyse the information contained in the biological molecules — nucleic acids and proteins — by means of computational methods. Past research in bioinformatics has been devoted to the production and understanding of genomic data, but recently it is increasingly permeating other fields of biology like functional and structural genomics and proteomics.

Protein bioinformatics has the main objective to discover the molecular mechanisms that rule the correct operation of a protein or, in case of pathologies, which lead to an altered or null protein function. Through the knowledge of these mechanisms it is possible to understand the complex processes involved in living systems and possibly correct dysfunctional behaviours. Such research is quite complex to carry out as a protein function involves a combination of several factors, many of which are still unknown.

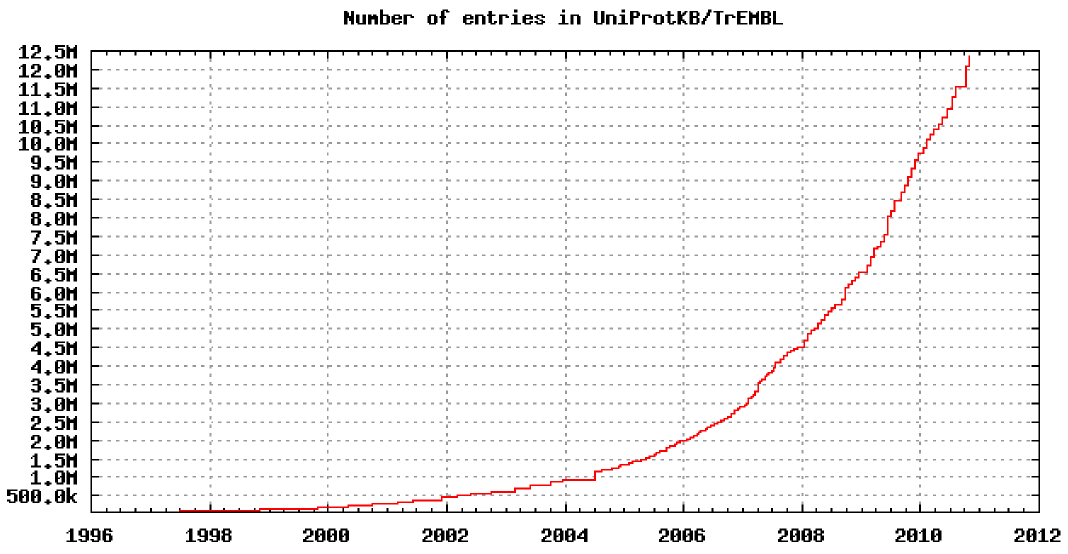
Knowing the protein three-dimensional structure is of primary importance for understanding its role and its function. The three-dimensional shape allows the protein to correctly interact with other molecules and macromolecules. The function that the protein carries out varies on the basis of the spatial conformation it assumes. The three-dimensional structure of a protein can be experimentally determined. Two quite expensive techniques are used with this aim: the X-ray crystallography [53] and the Nuclear Magnetic Resonance (NMR) [144].

Large scale genomics projects [81] are providing a huge amount of protein sequential and, at a lower but increasing rate, structural information [24, 27]. At the beginning of november 2010 the Universal Protein Resource database (UniprotKB) [131] counted for more than twelve millions of synthesised proteins. Figure 1.1 shows the growth of the database in the last years for the manually annotated and reviewed sequences (Swiss-Prot) and the automatically annotated and not reviewed sequences (TrEMBL).

At the same date, the database of protein structures, Protein Data Bank (PDB) [14], maintained by the Research Collaboratory for Structural Bioinformatics (RCSB), reported more than 64000 deposited protein structures. A large portion of such proteins have their function still undetermined, as it is often not straightforward to understand the details of a protein function even when its three-dimensional structure is known. The task requires a time-consuming trial-and-error process of hypothesis formulation and verification by targeted experiments such as site-directed mutagenesis [73].



(a) Growth of UniprotKB/Swiss-Prot released sequences.



(b) Growth of UniprotKB/TrEMBL released sequences.

Figure 1.1: Growth of UniprotKB released sequences.

Considering the rate at which protein sequences are synthesised and protein structures are solved (see Figure 1.2), the gap with respect to functionally characterised proteins is destined to increase over time. Automatic approaches for the detection of a protein function can be very useful in narrowing this gap.

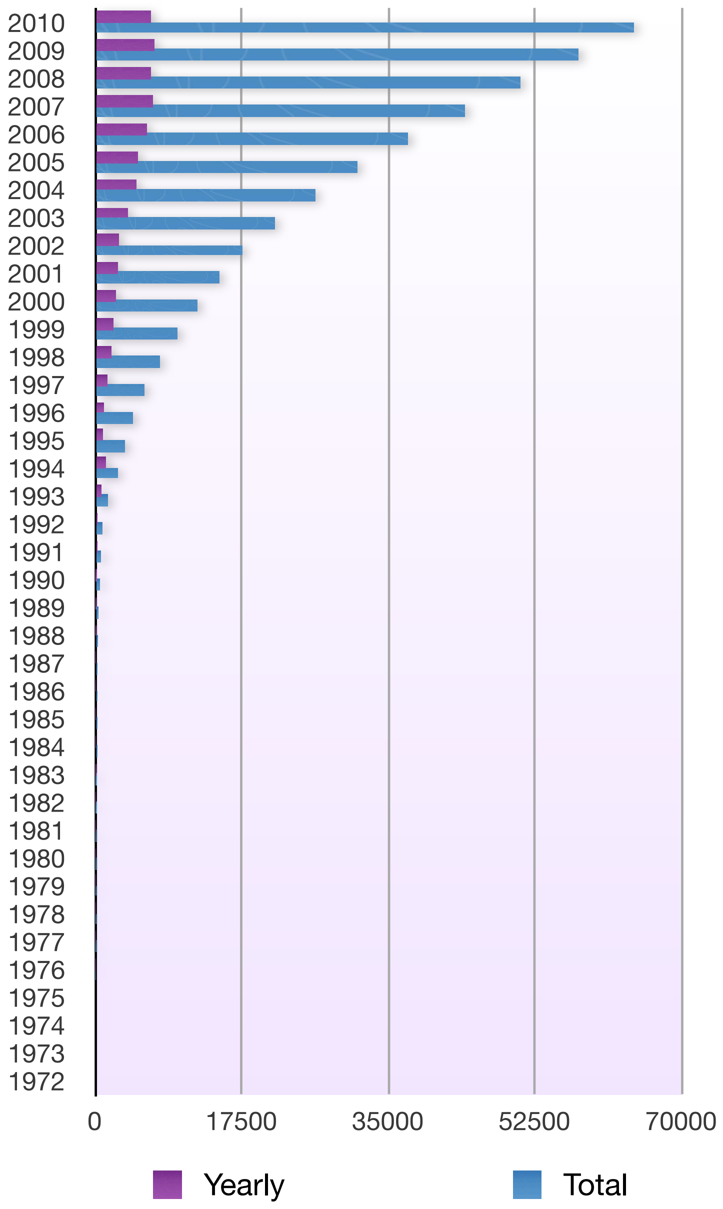


Figure 1.2: Growth of PDB released protein structures per year.

The most used approaches for the functional characterisation of genes and their proteins implement homology-based strategies. Novel protein function is inferred by aligning the sequences or by superimposing the structures with already annotated proteins. By using automatic approaches exploiting the sequence and structural comparison of new proteins with manually annotated ones, it is possible to assign a function to a large number of proteins in a fraction of the time and cost needed for a laboratory study. Unfortunately those simple approaches have shown their limitations in various context (see Section 1.3). Furthermore, an annotated homologue of the target protein needs to be available preventing the homology transfer applicability to novel folds and the increasing lack of functional annotations makes this strategy even less effective.

Machine learning techniques provided a valid alternative to homology-based methods. They allow for the development of predictors that are able to abstract from the single case and learn an accurate statistical or logical model based on known examples (*supervised learning*).

In this thesis the focus is on the characterisation of the molecular function of enzymes. Enzymes are those proteins whose role is to accelerate chemical processes inside a cell. The identification of the enzyme function is a required step for the innovation of biotechnologies used in the agriculture and food fields, as well as in the pharmacological and biochemical fields. New enzymes can be produced for their application in agriculture and food biotechnological processes as, for instance, enzymatic sensors for food control, or in water depuration and soil treatment and disinfection. From the pharmacological side, understanding the mechanisms of enzymes operation is fundamental for making an antibiotic an effective drug. Enzymes present in pathogenic bacteria must be able to form a complex and therefore to inhibit proteins that are essential for the bacterium life.

An enzyme function is defined by a topological region — called functional or active site — composed of amino acid residues located in specific positions of the three-dimensional disposition taken by the protein chain. The understanding of the enzyme molecular function has two prerequisites. The first is to discover its functional sites and characterise their properties. The second is to characterise the properties of the single amino acids involved and the way in which they interact among each other and with other molecules, e.g. drugs.

Two "wet laboratory" approaches are typically used for gaining insights about an enzyme molecular function: inhibition studies and random mutagenesis. By inhibiting one amino acid at a time and then by observing variations in the enzyme activity it is possible to infer a putative functional role of the inhibited residues. In principle, the inhibition of a functional residue should correspond to a decrease and sometimes to a complete loss of the enzyme biological activity. Random mutagenesis aims at generating a number of mutants of the same enzyme (the *wild type*). While directed mutagenesis requires the knowledge of the sequence or the structure of the protein, random mutagenesis can be applied without knowing any information and allows for the creation of a library of mutants to be screened. The induced mutations can lead to inactive mutants, more convenient mutants showing an improved biological activity or, they can be neutral mutations not affecting the enzyme activity. By analysing the common patterns of mutation and their correlation with the observed mutant activity it is possible to formulate hypotheses on the functional sites and the amino acids involved in them.

The analysis and mining of information from mutation data is also extremely important in molecular genetics studies aiming at developing effective cures to a number of diseases that stem from specific protein defects [143]. Diseases like the cystic fibrosis, cancer and Human Immunodeficiency Virus (HIV) infection all depend in some way on specific mutations taking place into specific proteins, especially enzymes — for the cystic fibrosis even a single mutation over a total of 1500 amino acids. The HIV case is an emblematic example of infection that encompasses many aspects of protein structure and function. Stopping an HIV infection requires to understand these many aspects. The virus has a high mutation rate and it is prone to the development of the resistance to specific drugs. Hence, the analysis of the virus mutations and the discovery of those residues that are necessary for the virus proteins proper functioning, is extremely important. Moreover, drug resistance development is often the result of multiple mutations occurring along the primary sequence. The correlation among mutations and their relationship with the resistance to a drug should be also taken into consideration.

Geometrical and statistical approaches are crucial for understanding the relationship between molecular structures and their function. Geometrical approaches are needed for managing and analysing structural properties of the molecules and their interactions. Statistical approaches are needed for handling and analysing large quantity of data, with the aim of extrapolating rules that can be generalised to novel cases. In particular, machine learning techniques can be used for identifying spatial regions hosting protein binding and functional sites. They can also be very useful in determining each one of the functional residues or reducing the number of candidates to be experimentally verified.

In the last years, the most promising machine learning techniques have evolved in the direction of learning from structured objects — such as sequences and graphs — and performing structured prediction. Among the supervised learning techniques, kernel methods [123] and especially Support Vector Machines (SVMs) [23, 43] have been successfully applied to several bioinformatics applications [76].

Statistical Relational Learning (SRL) [64] techniques also shown to be particularly suitable for learning and mining data in bioinformatics applications. They combine the advantages of Inductive Logic Programming (ILP) and statistical learning, namely the ability to learn a model of the concepts that can be readily interpreted by a human domain-expert, and the robustness principles of the statistical learning theory.

The aim of this thesis is to investigate and develop effective and advanced machine learning techniques for helping in characterising enzyme molecular functions and understanding the mechanisms allowing the proteins to operate. For instance, machine learning approaches for mining features that are relevant for causing the enzyme functioning or the enzyme malfunction, can be very useful. They can suggest the reason why a mutant shows an improved or reduced activity on a certain drug or, why an enzyme is resistant to a certain drug and/or not resistant to another one. For instance, by modelling the presence of a mutation close to the functional site that changes a structural amino acid, like a proline, into a large and basic amino acid, like arginine or lysine.

## 1.2 The Protein Function Identification Problem

In the previous section the problem of the protein function understanding and its complexity has been introduced, but what is a protein and how can we define its function?

In this section a short introduction to the underlying biological aspects needed to understand the rest of the work will be given.

### 1.2.1 Quick Primer on the Biological Aspects

The complex processes involved in living systems are the result of the harmonic action of molecules and macromolecules, mainly proteins. The cell is a complex biochemical machinery and proteins are its "functional devices".

Proteins are synthesised according to rules written in the genome. Oversimplifying the complex mechanisms involved, the process can be roughly schematised in the so called *Central Dogma* of molecular biology: "from

DNA to RNA to protein" (Figure 1.3).

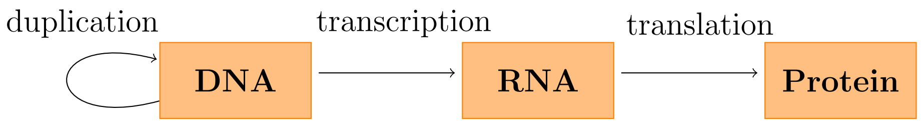


Figure 1.3: The Central Dogma of molecular biology.

Each gene in a cell genome is a deoxyribonucleic acid (DNA) fragment, which codes for a protein. Hence, genes "building blocks" are nucleotides, made of the union of a sugar (deoxyribose) and one of the four bases: adenine (A), guanine (G), cytosine (C), and thymine (T). The sequence in which the nucleotides are placed along the DNA strand determines the properties of the different genes.

Each one of the triplets of a gene sequence in the alphabet $\Sigma_N = \{A, C, T, G\}$ codes for one of the twenty amino acid in nature, in the alphabet

$$\Sigma_P = \{A, C, D, E, F, G, H, I, K, L, M, N, P, Q, R, S, T, V, W, Y\},$$

according to the universally accepted Genetic Code [106].

Amino acids are the "building blocks" of proteins. They are organic molecules characterised by two functional groups: an *aminic* group ($NH_2$) and an *acidic* group, the carboxylic group ($COOH$), which together give the name of "amino acid".

The two functional groups are held together by a carbon atom, known as *alpha carbon* ($C_\alpha$). A further chain of atoms, bound to the $C_\alpha$, makes the amino acid side-chain (R), which differentiates from the main chain (see Figure 1.4) and is the part whose composition confers specificity to the amino acid: all of the twenty amino acids have the same main chain but their side-chains are made of different atoms, ranging from a single hydrogen atom, as in the case of a glycine, to more complex structures of benzene rings, as for a tyrosine.

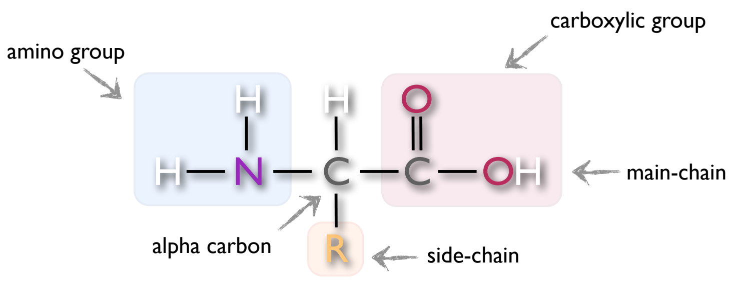


Figure 1.4: Amino acid.

Table 1.2.1 also shows the classical 20 symbol alphabet with the three-letter and the corresponding one-letter abbreviations for the amino acids. Since the twenty amino acids differ on the basis of the chemical properties of their side chain (which can be polar, apolar, acid or basic), different alphabets can be used that group them according to functional and chemical properties (see Table 1.2.1).

A protein is composed of one or more chains of amino acids held together by a peptide bond, a covalent bond between the carboxylic and the amino group of two consecutive amino acids. The formation of a peptide bond between to amino acids (see Figure 1.5) produces a free water molecule.

The result is a polypeptide chain of *residues* of amino acids. The sequence of the residues along the chain is the protein *primary structure*. The amino acids along the chain, due to their three-dimensional structure, can arrange into often repetitive and bound structures, called *secondary structure elements* (SSE), for example $\beta$-sheets and $\alpha$-helices (see Figure 1.6). The *tertiary structure*, is the result of the chain *folding process*, due to: (a) the space the amino acid side-chains take up and, (b) the weak forces of attraction and repulsion — e.g. hydrophobicity and hydrophilicity — among either the side-chain groups and the fluid in which the polypeptide

| Classical Alphabet ($\|\Sigma\| = 20$) | | Chemical Alphabet ($\|\Sigma\| = 8$) | | Functional Alphabet ($\|\Sigma\| = 3$) | | Hydrophobicity Alphabet ($\|\Sigma\| = 2$) | |
|---|---|---|---|---|---|---|---|
| **AA** | **Symbol** | **Symbol** | **Meaning** | **Symbol** | **Meaning** | **Symbol** | **Meaning** |
| Ala | A | L | aliphatic | H | hydrophobic | O | hydrophobic |
| Arg | R | B | basic | C | charged (+) | I | hydrophilic |
| Asn | N | M | amidic | P | polar | I | hydrophilic |
| Asp | D | A | acidic | C | charged (-) | I | hydrophilic |
| Cys | C | S | sulphuric | P | polar | I | hydrophilic |
| Gln | Q | M | amidic | P | polar | I | hydrophilic |
| Glu | E | A | acidic | C | charged (-) | I | hydrophilic |
| Gly | G | L | aliphatic | P | polar | I | hydrophilic |
| His | H | B | basic | C | charged (+) | I | hydrophilic |
| Ile | I | L | aliphatic | H | hydrophobic | O | hydrophobic |
| Leu | L | L | aliphatic | H | hydrophobic | O | hydrophobic |
| Lys | K | B | basic | C | charged (+) | I | hydrophilic |
| Met | M | S | sulphuric | H | hydrophobic | O | hydrophobic |
| Phe | F | R | aromatic | H | hydrophobic | O | hydrophobic |
| Pro | P | I | iminic | H | hydrophobic | O | hydrophobic |
| Ser | S | H | hydroxylic | P | polar | I | hydrophilic |
| Thr | T | H | hydroxylic | P | polar | I | hydrophilic |
| Trp | W | R | aromatic | H | hydrophobic | O | hydrophobic |
| Tyr | Y | R | aromatic | P | polar | I | hydrophilic |
| Val | V | L | aliphatic | H | hydrophobic | O | hydrophobic |

Table 1.1: Amino acid alphabets.

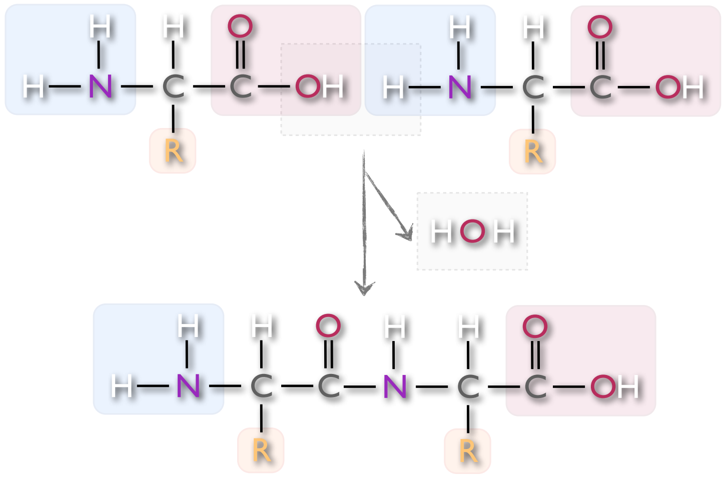


Figure 1.5: Peptide bond formation.

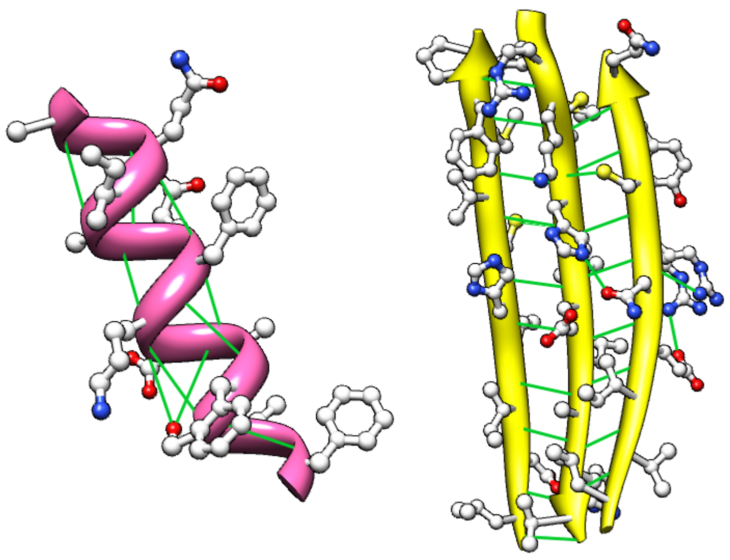


Figure 1.6: Structure of an $\alpha$-helix and a $\beta$-sheet.

chain is dip into. Finally the *quaternary structure* is the composition of a number of polypeptide chains belonging to the same protein, in other words, it is the whole set of subunits that compose an oligomer. The four levels of a protein structure are visualised in Figure 1.7.

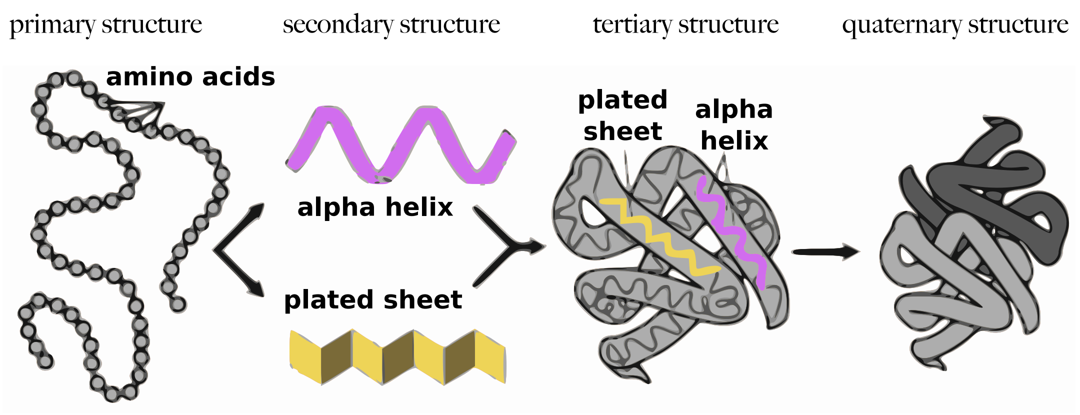


Figure 1.7: The levels of a protein structure (adapted from a *courtesy of the National Human Genome Research Institute*)

### 1.2.2 Defining a Protein Function

There is no single or fully standardised way of defining a protein function. A protein function can be defined at the molecular, biological or cellular scale and, at each level, further levels of detail can be considered. Furthermore, a protein function has many other distinctive aspects that depend on the environment in which the protein operates and which conditions affects its behaviour [61]. For example, the same enzyme can perform different functions depending on whether it is located in the liver or in the eye.

Proteins inside the cell perform several different functions. Three fundamental examples are storage proteins, structural proteins and enzymes. At the biological level we can distinguish among enzymes, transport proteins (haemoglobin), hormones and proteins involved in defence against germs (antibodies), in structural support or body movement (contractile proteins) [143].

A quite successful attempt to standardise the protein function definition at the different scales is represented by the Gene Ontology (GO) [29]. Concepts in the GO, called GO terms, are organised in a directed acyclic graph (DAG) structure. GO terms are connected on the base of a general-to-specific relation.

This thesis focuses on one of the aspects of a protein function definition: the molecular function. At the molecular level a protein function is defined by a topological region of the protein called functional site, which is a functional domain in the protein three-dimensional structure. Functional domains can have a multiplicity of roles within a protein. For example in enzymes, functional domains are also called active or catalytic sites, while in other kind of proteins they correspond to binding sites with another macromolecule.

In this thesis the attention is concentrated on enzymes. Enzymes are proteins able to accelerate chemical processes inside a cell. The enzyme works by forming complexes with the reactants and in doing so it lowers the activation energy of the reactions thus increasing their rate. This process is the catalysis. An enzyme has usually the structure of a globular protein and the 3D disposition of the residue chain is somewhat specific. The residues take up well-defined positions which are essential for the recognition and binding of specific substrates, in other words, for the biological activity of the enzyme. The residues that are directly involved in the catalytic process (e.g. nucleophiles, proton-donors) constitute the active site, while residues in the surrounding space play the role of attracting and orienting the molecule to bind and constitute the binding domain.

The Enzyme Commission (EC) Nomenclature, proposed by the International Union of Biochemists (IUB), is the first example of an enzyme molecular function categorisation. The classification has the aim to standardise the definition of the enzyme function by assigning reactions to a hierarchy of four categories from general to specific, namely the class, the superfamily, the family and the subfamily (Figure 1.8).

Enzymes are grouped in six functional classes in the EC Nomenclature, based on the type of the catalysed reaction:

1. oxidoreductases, for oxidation-reduction reactions;

2. transferases, for the transfer of functional groups;

3. hydrolases, for hydrolysis reactions;

4. lyases, for breaking chemical bonds;

5. isomerases, for isomeration reactions;

6. ligases, for chemical bonds formation.

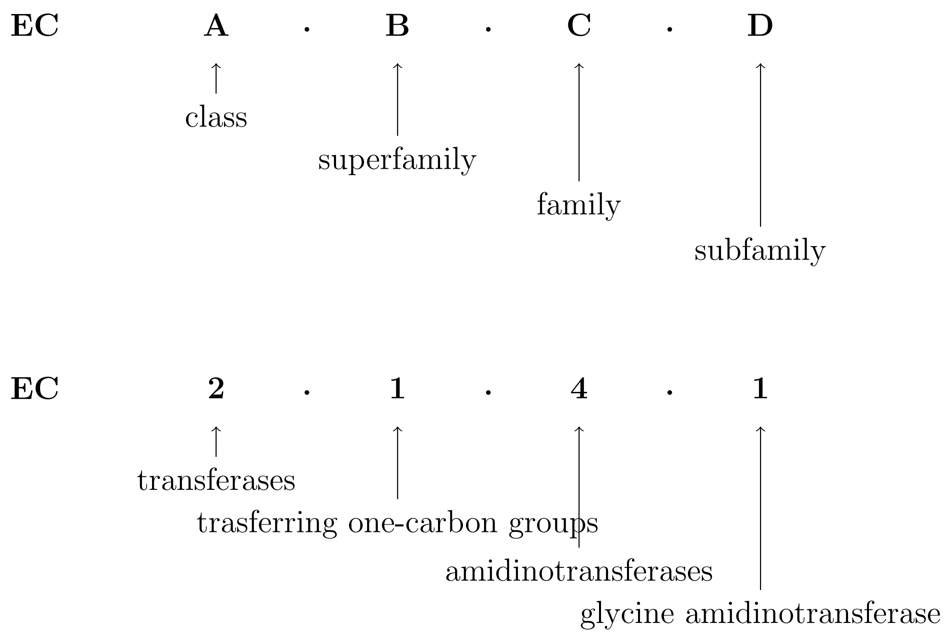


Figure 1.8: Enzyme Classification number.

The pie chart in figure 1.9 shows the results obtained by querying the UniprotKB with the keyword “enzyme”. The chart highlights the number of synthesised enzymes currently in the Uniprot divided by EC class.

In order to perform their function, many enzymes need to be bound to an additional non-protein component called cofactor. Cofactors can be grouped in:

(a) coenzymes, i.e. dissociable cofactors that are usually organic;

(b) prosthetic groups, i.e. non dissociable cofactors.

Common examples of cofactors are metal ions. The enzyme lacking the cofactor is inactive and it is called apoenzyme, while the enzyme with the cofactor is active and it is called holoenzyme.

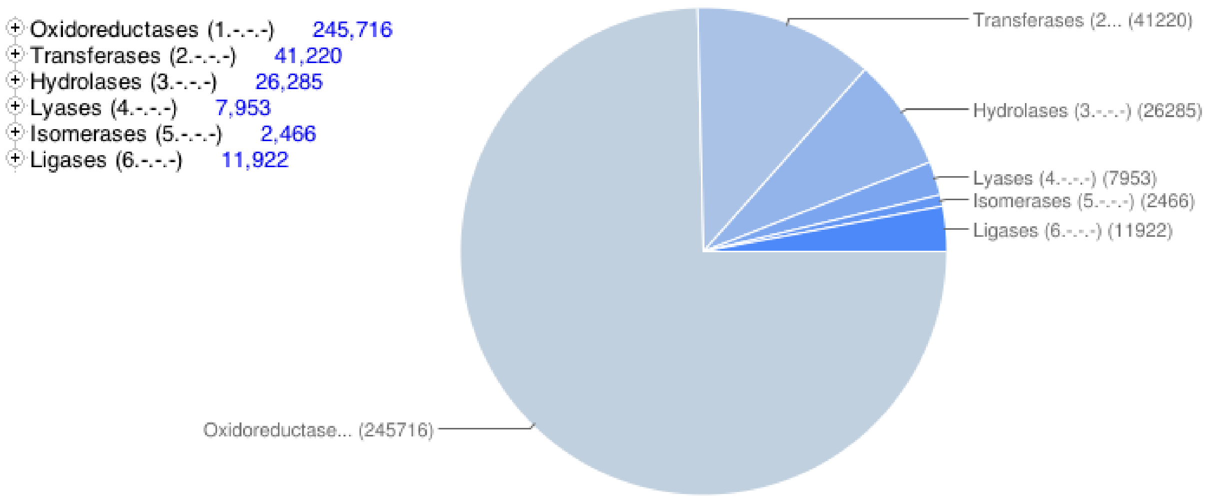


Figure 1.9: Pie chart of the results of the "enzyme" query to the UniprotKB.

## 1.3 Traditional Approaches to Protein Function Identification

Traditional approaches for the functional characterisation of genes and their proteins implement homology-based strategies. Novel protein function is inferred by aligning the sequences or by superimposing the structures with already annotated proteins and then by transferring the annotation from the most similar to the novel one.

Typically, sequence databases are searched for finding a significant sequence similarity to another protein whose function has been experimentally characterised. Basic Local Alignment Search Tool (BLAST) [28] is a popular tool for searching a query sequence against a database of known protein sequences and discovering sequence similarities. For this reason the word "BLASTing" is often used nowadays referring to this search process.

The biological rationale for the homology-based transfer is as simple as powerful: very similar sequences most probably share a common ancestor (they are homologous) and therefore, given this evolutionary relationship, have identical or very similar function.

The protein structure is even more informative than the amino acid sequence alone. Knowing the protein structure is fundamental for understanding the biochemical mechanisms by which the protein performs its function. Whenever the tertiary structure of a novel protein with unknown function is available, the structural similarity to other proteins can allow the functional annotation transfer [61]. For instance, two protein structures with the same fold can have an identical or similar function even if the two protein sequences have little similarity [25]. That is because the protein tertiary structure tends to be more preserved than the primary structure along evolution.

By using automatic approaches exploiting the sequence and structural similarity of new proteins with manually annotated ones, it is possible to assign a function to a large number of proteins in a fraction of the time and cost needed by a laboratory study [58]. Unfortunately those simple approaches have shown their unreliability for functional annotation even in presence of a high sequence or structural identity percentage and other limitations in various context. First, an annotated homologue of the target protein needs to be available preventing their applicability to novel folds. Furthermore, if a protein has more than one domain a detected evolutionary relationship, even based on a high sequence similarity with another protein, can be limited to only one of the domains, leading to an erroneous annotation [135]. Even considering the 3D structure similarity, there are cases in which proteins with similar overall tertiary structure can have different active sites, i.e. different functions [101, 133], and proteins with different overall tertiary structure can show the same function and similar active sites. Finally, the increasing lack of functional annotations makes the homology-based strategies even less effective as it becomes even more difficult to find an annotated reference sequence for novel proteins.

Machine learning techniques provide a valid alternative to homology-based methods in cases in which the homology-based transfer cannot be applied. These methods are able to abstract from the single case and to learn accurate statistical or logical models based on observations. In recent years they have been successfully applied for predicting enzyme functional sites and functional residues.

## 1.4 Review on Machine Learning Techniques

Machine learning primary aim is to provide effective algorithms for training a machine in automatically acquiring useful and accurate models from the experience.

A plethora of machine learning algorithms have been proposed in recent years, with different peculiarities concerning the problem formalisation they approach, the underlying method used, and how efficiently they are able to solve a learning task.

The following sections present a rapid excursus on the learning paradigms, tasks, and algorithms that are of interest for this thesis.

### 1.4.1 Learning Paradigms and Learning Tasks

Machine learning is a wide field that includes many different approaches for solving different learning problems: from medical diagnosis and financial analysis, to speech recognition and text categorisation or image analysis and computer vision. In recent years these algorithms have been also successfully applied to solve several bioinformatics applications [88].

Different learning paradigms exist. Among them there are the supervised learning and the unsupervised learning. In this thesis particular attention is devoted to these two learning paradigms.

**Supervised Learning**

In supervised learning, we are given a set of examples $(x^{(i)}, y^{(i)}) \in \mathcal{X} \times \mathcal{Y}$ with $i = 1, ..., m$, where $\mathcal{X}$ is the input space and $\mathcal{Y}$ is the output space. The learning problem is to derive a model $h : \mathcal{X} \rightarrow \mathcal{Y}$ for the unknown input-output relationship. This is achieved combining fitting of the training data with some constraints on the hypothesis space, in order to avoid overfitting and generalise to unseen instances. A typical approach consists of penalising overly complex hypotheses, a procedure called “regularisation”.

The error in fitting training data is formalised as the empirical risk:

$$Emp[h(\cdot) \neq y] = \frac{1}{m} \sum_{i=1}^{m} l(y^{(i)}, h(x^{(i)})) \tag{1.1}$$

where $l : \mathcal{Y} \times \mathcal{Y} \rightarrow \mathbb{R}$ is a *loss* function, which returns values according to the “goodness” of the prediction $h(x)$ with respect to the real output $y$. If we consider the 0-1 loss, which is defined as:

$$l(y, \hat{y}) = \begin{cases} 0 & \text{if } y \neq \hat{y} \\ 1 & \text{otherwise} \end{cases} \tag{1.2}$$

where $\hat{y} = h(x)$, the empirical risk corresponds to the number of misclassified instances of the training set.

Figure 1.10, adapted from [129], schematises the learning supervised by examples. A new example $x$ is assigned to $y$ according to the learnt model $h$.

Depending on the nature of the output space $\mathcal{Y}$, we can distinguish among different learning tasks. The most common are:

- **classification** $\mathcal{Y} = \{y_1, ..., y_l\}$ is a discrete set of labels. Among the classification tasks a further distinction can be done:

  - *binary classification* whenever $l = 2$, typically $\mathcal{Y} = \{+1, -1\}$ a new example is classified in one of two classes (as positive or negative);
  - *multiclass classification* whenever $l > 2$.

- **regression** $\mathcal{Y} = \mathbb{R}$ is a continuous set.

The rest of the thesis focuses on classification problems. Classification and regression are standard supervised learning problems in which the outputs are simple scalars. The output space $\mathcal{Y}$ can be also characterised by more complex and structured objects like, sequences, trees or graphs (see section 1.4.2).

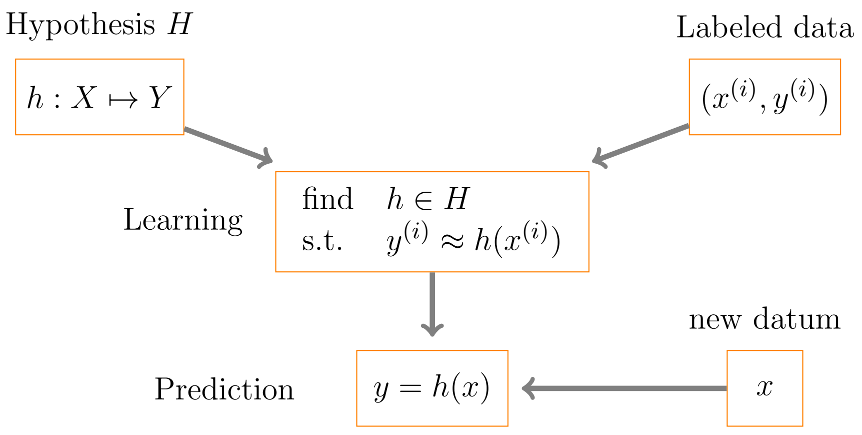


Figure 1.10: Supervised learning (taken from [129]).

*Multitask Learning* [34] is a particular example of supervised learning paradigm consisting in learning a model for a task and other related tasks at the same time. It deals with the problem of exploiting information on related tasks in order to improve the predictive performance and it is considered a kind of inductive transfer.

In this setting given a set $T$ of learning tasks and, for each $t \in T$, a set of examples $\mathcal{D}_t = \{(x_t^{(i)}, y_t^{(i)})\}$ with $i = 1, ..., m_t$, the learning problem is to derive simultaneously a model $h_t$ for each $t \in T$ such that $h_t(x_t^{(i)}) \approx y_t^{(i)}$ while retaining the capability of the models to generalise to unseen data. Indeed, the primary aim of multitask learning is to improve the generalisation performance of a learnt model leveraging additional information from related tasks.

### Unsupervised Learning

In unsupervised learning input data $x^{(i)}$ with $i = 1, ..., m$, have no supervised target outputs. The objective is to learn how data are organised, reveal patterns from them or build a representation that can be used for further analyses [65]. Clustering falls in this category.

Given a set $\mathcal{D}$ of $m$ inputs, the problem is to find a set $\mathcal{C} \subseteq \mathcal{P}(\mathcal{D})$ such that

$$\mathcal{C} = \{\mathcal{C}_j \in \mathcal{P}(\mathcal{D}) | \bigcup_j \mathcal{C}_j = \mathcal{D} \wedge \bigcap_j \mathcal{C}_j = \emptyset\}$$

Consequently a partial clustering can be defined as:

$$\mathcal{C}_P = \{\mathcal{C}_j \in \mathcal{P}(\mathcal{D}) | \bigcup_j \mathcal{C}_j \subseteq \mathcal{D} \wedge \bigcap_j \mathcal{C}_j = \emptyset\}$$

The hierarchical clustering is a popular approach to clustering, which is of interest in the present thesis. This clustering technique, differently from partitioning approaches, does not require to specify a priori the number of desired clusters. The algorithm can proceed with a bottom-up strategy, by starting with singletons and recursively aggregating pair of clusters. Alternatively, a top-down strategy can be followed by splitting an initial cluster with all the examples recursively into smaller clusters. Merging or splitting two clusters require to define inter-cluster similarity or distance measures (the linkage function).

The results is a hierarchy of clusters that is usually depicted using a tree structure, the dendrogram. By cutting at different levels of the dendrogram, i.e. at different similarity thresholds, a different clustering — with different number of clusters — can be obtained.

The hierarchical clustering can provide noticeable additional information on the structure of the data.

### 1.4.2 Primer on Support Vector and Kernel Machines

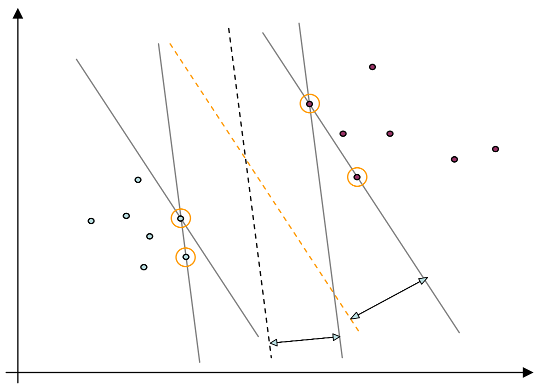
Among the supervised learning techniques kernel methods [123] and especially SVMs [23, 43] have been successfully applied to many problems in computational biology [124].

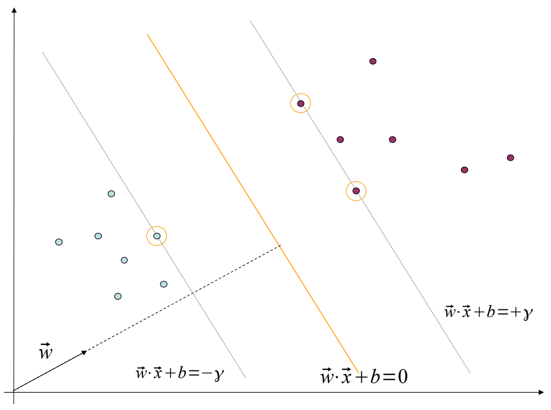
Some examples are translation initiation site recognition in DNA genes [152], promoter region-based classification of genes [109], protein-protein interaction [20], functional classification from microarray expression data [108], protein function classification [21].

#### Support Vector Machines

SVMs are learning algorithms based on principles of the statistical learning theory [139]. They aim at linearly separating examples with a large margin, possibly accounting for margin errors [42].

Suppose that $\mathcal{X} = \mathbb{R}^n$, i.e. examples are represented as vectors of features, and that we can define an inner product $\langle\cdot,\cdot\rangle$ on $\mathcal{X}$. An SVM learns the parameters of an hyperplane separating positive and negative examples. In Figure 1.11(a) examples of separating hyperplanes are shown. The functional margin $\gamma_i$ of an example with respect to an hyperplane $\langle\vec{w},\vec{x}\rangle + b = 0$ is the product $y_i(\langle\vec{w},\vec{x_i}\rangle + b)$. Its geometrical definition $\langle\frac{\vec{w}}{||\vec{w}||},\vec{x_i}\rangle + \frac{b}{||\vec{w}||}$ corresponds to the distance of $\vec{x_i}$ from the hyperplane. The margin of a separating hyperplane is the minimum margin among the margins of the training set. Separating hyperplanes with different margins are shown in Figure 1.11(a). Figure 1.11(b) shows the separating hyperplane that realises the maximum margin $\gamma$ over the training set. Results from the statistical learning theory state that the $h$ that grants the lowest bound on the generalisation error is the one that maximises the separation among positive and negative examples.


(a) Separating hyperplanes.


$\vec{w}$
$\vec{w}\cdot\vec{x}+b=-\gamma$
$\vec{w}\cdot\vec{x}+b=0$
$\vec{w}\cdot\vec{x}+b=+\gamma$
(b) Maximal separating hyperplane.

Finding the maximal margin hyperplane translates into solving the following optimisation problem (*hard margin*):

$$\begin{aligned} \min_{\vec{w},b} \quad & \tfrac{1}{2}||\vec{w}||^2 \\ \text{subject to} \quad & y_i(\langle \vec{w}, \vec{x} \rangle + b) \geq 1 \quad \forall i = 1, ..., m \end{aligned} \tag{1.3}$$

where $b \in \mathbb{R}$ and $\vec{w} \in \mathbb{R}^n$. Note that, this primal formulation is obtained after a rescaling of $\vec{w}$ in a way that the points lying on $\langle \vec{w}, \vec{x} \rangle + b = \pm\gamma$ now lie on $\langle \vec{w}, \vec{x} \rangle + b = \pm 1$.

This quadratic convex optimisation problem has a unique global optimum. Given the learnt hyperplane, the decision function is simply:

$$h(\vec{x}) = \text{sgn}(\langle \vec{w}, \vec{x} \rangle + b) \tag{1.4}$$

Solving 1.3 in its dual form can be more convenient and leads to considerations that allow for the extension of the algorithm to non linearly separable training sets and more complex input data. By deriving the Langrangian of the primal optimisation problem in 1.3, we first obtain:

$$L(\vec{w}, b, \vec{\alpha}) = \frac{1}{2}||\vec{w}||^2 - \sum_{i=1}^{m} \alpha_i (y_i(\langle \vec{w}, \vec{x} \rangle + b) - 1) \tag{1.5}$$

where $\alpha_i \geq 0$ are the langrangian multipliers for incorporating the linear constraints. By differentiating the Langrangian with respect to the primal variables and setting the derivatives to be equal to 0, we obtain:

$$\frac{\partial L(\vec{w}, b, \vec{\alpha})}{\partial \vec{w}} = 0 \Rightarrow \vec{w} = \sum_{i=1}^{m} \alpha_i y_i \vec{x}_i \tag{1.6}$$

$$\frac{\partial L(\vec{w}, b, \vec{\alpha})}{\partial b} = \sum_{i=1}^{m} \alpha_i y_i = 0 \tag{1.7}$$

Substituting 1.6 and 1.7 in the primal Langrangian 1.5 we obtain the dual form of the optimisation problem 1.3:

$$\begin{aligned} &\max_{\alpha \in \mathbb{R}^m} \quad \textstyle\sum_{i=1}^{m} \alpha_i - \frac{1}{2} \sum_{i,j=1}^{m} y_i y_j \alpha_i \alpha_j \langle \vec{x}_i, \vec{x}_j \rangle \\ &\text{subject to} \quad \alpha_i \geq 0 \quad \forall i = 1, ..., m \\ &\qquad \textstyle\sum_{i=1}^{m} y_i \alpha_i = 0 \end{aligned} \tag{1.8}$$

In this dual formulation training data appear only in the so called *Gram matrix*, of their inner products. By substituting 1.6 in 1.4, the decision function becomes:

$$h(\vec{x}) = \text{sgn}(\sum_{i=1}^{m} \alpha_i y_i \langle \vec{x}_i, \vec{x} \rangle + b)$$

The Karush-Khun-Tucker conditions state that the optimal solution $\alpha_i^*(\vec{w}^*, b^*)$ satisfies the following relation:

$$\alpha_i^*(y_i(\langle \vec{w}^*, \vec{x}_i \rangle + b^*) - 1) = 0 \qquad i = 1, ..., m \tag{1.9}$$

This implies that if $\alpha_i^* = 0$, the training input $\vec{x}_i$ is not affecting $\vec{w}^*$ otherwise $y_i(\langle \vec{w}^*, \vec{x}_i \rangle + b^*) - 1$ must be equal to 0, hence $\vec{x}_i$ lies on the frontier distant 1 from the maximal hyperplane. The latter points $\vec{x}_i$ are the *support vectors* and are the only ones of interest for characterising the separating function. In Figure 1.11(b) the support vectors and the maximum margin hyperplane are highlighted in orange.

*Soft Margin SVM*

With the aim of improving the generalisation and separability in case of noisy data, the primal optimisation problem 1.3 can be rewritten relaxing the constraints for taking into account classification errors. Regularisation can be introduced in the objective function. The resulting optimisation problem is:

$$\begin{array}{ll} \min_{w,b,\xi} & \frac{1}{2}||w||^2 + C\sum_{i=1}^{m} \xi_i \\ \\ \text{subject to} & y_i(\langle w, x \rangle + b) \geq 1 - \xi_i \quad \forall i = 1, ..., m \end{array} \tag{1.10}$$

where $\xi_i \geq 0$ are slack variables for constraints relaxation and $C$ is the regularisation parameter that measures the tradeoff between the misclassification error and the margin maximisation. As $C$ grows, margin errors are more penalised. Therefore, for $C \rightarrow \infty$ the problem approximates the solution with the hard margin. Figure 1.11 show how solving a soft margin problem could result in larger margins and possibly better generalisation.

In some cases this regularisation can allow to solve the inseparability of noisy data. In the next section, it is discussed how the inseparability can be solved mapping the input data in a feature space where the inner product provides a measure of similarity between input examples and a separating function based on these properties.

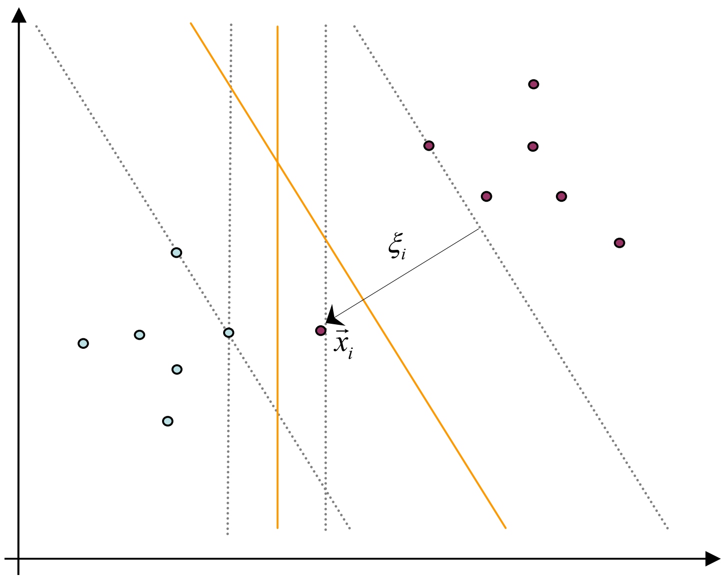


Figure 1.11: Soft margin SVM.

### Kernel Methods

In many real-world problems dependencies aimed at making predictions are better captured by nonlinear models. Kernels are largely used in learning algorithms because they allow for an implicit mapping of objects of the input space in a larger feature space where a linear separation can be used (see Figure 1.12).

In the input space a *kernel* function $k : \mathcal{X} \times \mathcal{X} \rightarrow \mathbb{R}$ can be used as a similarity measure. $k$ corresponds to an inner product in a high dimensional feature space which is in general different from the input space $\mathcal{X}$. Thus $k$ implicitly builds a mapping $\phi : \mathcal{X} \rightarrow H$ where

$$k(x, x') = \langle \phi(x), \phi(x') \rangle \tag{1.11}$$

generalising the notion of inner product to arbitrary domains [123, 125]. This is called the *kernel trick*.

In the previous section we have seen as in the dual formulation of the support vector learning optimisation problem, input data appear only in

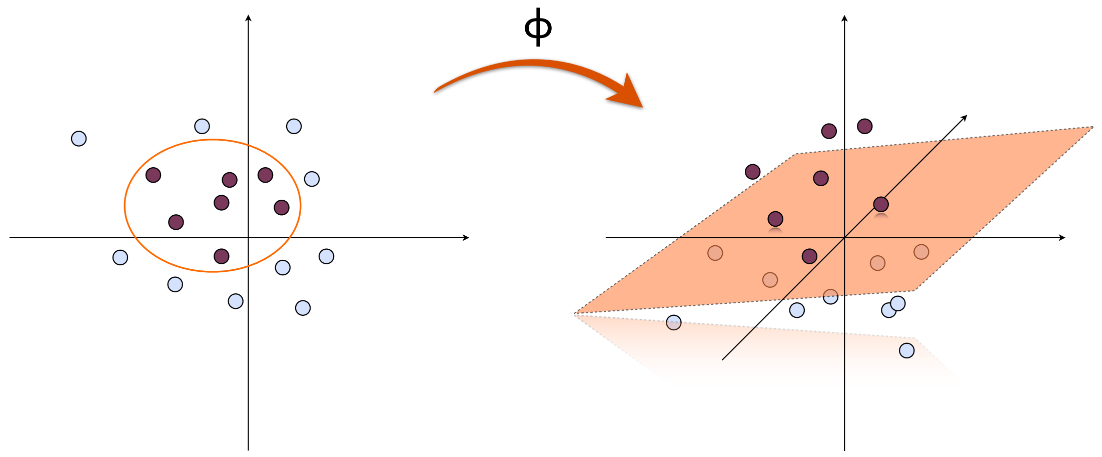


Figure 1.12: Mapping in a new feature space.

the inner product, the *Gram matrix*. Now it easy to see that the inner product can be substituted with a kernel function applied to an arbitrary input domain. This leads to the following decision function:

$$h(x) = \text{sgn}(\sum_{i=1}^{m} \alpha_i y_i k(x_i, x) + b)$$

A kernel function is symmetric and positive semi-definite (Mercer's conditions [139]). This positive definiteness grants some properties: closure under sum, direct sum, multiplication by a scalar, product, tensor product, zero extension, point-wise limits and exponentiation [43, 69].

This properties are worth because they allow for the definition of more complex kernel functions as a combination of simpler kernel function.

*Polynomial kernels* map the input space vectors into feature vectors containing the same input features plus those representing their conjunctions.

It can be defined as:

$$k(\vec{x}, \vec{z}) = (\langle \vec{x}, \vec{z} \rangle + c)^d \tag{1.12}$$

where $c$ and $d$ are parameters. $d$ is the degree of the polynomial kernel. If we assume a dimension $n$ of the input space $X$, the feature space dimension becomes $\binom{n+d}{d}$, corresponding to all the possible monomial up to the degree $d$ (a demonstration by induction can be found in [125]).

**Kernels for Structured Data**

Thanks to the *kernel trick*, algorithms like SVMs can be applied not only to data that can be easily embedded in a Euclidean space, but also on structured objects — like trees, graphs and strings — provided that an appropriate kernel function can be defined on them. A survey of kernels for structured data can be found in [62].

Special attention has been devoted to convolution kernels [69]. Convolution Kernels are based on the idea that given a relation $\mathcal{R}$ between a composite object and its parts, the similarity between two objects can be evaluated composing similarity values between their parts, calculated by defining appropriate kernels on them.

More formally let $x \in \mathcal{X}$ be a structured object composed by $D$ "parts" $(x_1, ..., x_D) \in \mathcal{X}_1 \times \mathcal{X}_2 \times ... \times \mathcal{X}_D$. We define $\mathcal{R}$ as the relation between the object and its "parts" and the corresponding decomposition relation $\mathcal{R}^{-1}(x) = \{(x_1, ..., x_D) : \mathcal{R}((x_1, ..., x_D), x)\}$. A convolution kernel on two structured objects is given by:

$$k(x, x') = \sum_{(x_1,...,x_D) \in \mathcal{R}^{-1}(x)} \sum_{(x_1,...,x_D)' \in \mathcal{R}^{-1}(x')} \prod_{i=1}^{D} k_i(x_i, x_i') \tag{1.13}$$

Convolution kernels have been defined for structures like, sequences, trees and graphs [62]. A typical approach consists into evaluate the similarity between these structured objects in terms of number of common substructures [41, 63, 97]. Some examples are $k$-mers in strings [92], random walks in graphs [63], subtrees [41] or tree fragments [97] in tree structures.

**Predicting Structured Data**

Predicting structured data means learning functional dependencies between arbitrary input and output domains. Given a function $F : \mathcal{X} \times \mathcal{Y} \rightarrow \mathbb{R}$, $h$ is chosen such that $h(x) = \arg\max_{y\in\mathcal{Y}} F(x, y)$.

In this context the output $y$ has a structure, like a string, a tree or a graph [4]. Typical application for structured output prediction include RNA secondary structure prediction [49], named entity recognition [119], sequence alignment learning [150] and part of speech tagging [36]. In all these cases the output is a string. Another example is the parsing problem in natural language processing [136] in which the input $x$ is a sentence while output $y$ is a tree.

Given a novel input $x$, the prediction involves to search the output $y$ that maximises $F$ when paired with $x$. The structured output prediction problem is usually separated into a learning and a search problem. The main issue in this learning setting is given by the large number of output values $y$, which is usually exponential in the size of the input $x$. Therefore, in some cases one has to resort to heuristic optimisation strategies. By exploiting characteristics of the specific learning problem, it is possible to devise algorithms for efficiently approximating the optimal solution, e.g. based on dynamic programming.

Kernel machines have been adapted to structure-output prediction [136] by addressing the following optimisation problem:

$$\begin{aligned} &\min_{w,\xi} \;\; \tfrac{1}{2}||w||^2 + \tfrac{C}{m}\textstyle\sum_{i=1}^{m} \xi_i \\ &s.t. \qquad \langle w, \phi(x_i, y_i)\rangle - \langle w, \phi(x_i, \hat{y})\rangle \geq \Delta(y_i, \hat{y}) - \xi_i \;\; \forall \hat{y} \in \mathcal{Y} \setminus y_i \;\; i = 1, ..., m \end{aligned} \tag{1.14}$$

$\Delta(y, y')$ is the loss function and represents the cost of misclassifying $y$ for $y'$,$\phi(x, y)$ is a joint feature map on input-output pairs. A measure of similarity $\Delta$ must be defined also on the output space, which is usually problem specific.

For many prediction problems this computation is intractable, due to the exponential number of constraints appearing in the Quadratic Programming (QP) problem in (1.14). Thus, learning $w$ and computing $h(x)$ in an efficient way are the main issues.

In [136] by exploiting the structure of maximum-margin problem, a cutting-plane algorithm is proposed. The algorithm is an iterative procedure that at each step adds to the optimisation problem (that initially has an empty set of constraints) one of the most violated constraints until convergence to an approximate solution (examples are used in [136] and [150] for protein alignment models). Thus at each iteration a smaller size QP is solved.

### 1.4.3 Primer on ILP and Statistical Relational Learning

While statistical learning mainly focuses on propositional or attribute-value representation of the data and the features induced during learning, relational learning approaches allow to retain the relational structure of the data. This section introduces concepts and terminology from relational learning and statistical relational learning.

#### Relational Learning and Hypothesis Search

In Inductive Logic Programming (ILP) [100], training examples and background knowledge, as well as the features that are induced during learning, are represented in first-order logic. More specifically, definite clauses, which form the basis for the programming language Prolog, are used as the representation language.

A definite clause is an expression of the form $h \leftarrow b_1, ..., b_n$, where $h$ and the $b_i$ are atoms. Atoms are expressions of the form $p(t_1, ..., t_n)$ where $p/n$

is a predicate symbol of arity $n$ and the $t_i$ are terms. Terms are constants (denoted by lower case), variables (denoted by upper case), or structured terms. Structured terms are expressions of the form $f(t_1, ..., t_k)$, where $f/k$ is a functor symbol of arity $k$ and $t_1, ..., t_k$ are terms. The atom $h$ is also called the head of the clause, and $b_1, ..., b_n$ its body. Intuitively, a clause represents that the head $h$ will hold whenever the body $b_1, ..., b_n$ holds.

As an example, consider the atom $mut(A, h, C, y)$ indicating a mutation that results in the replacement of the amino acid “Histidine” by the amino acid “Tyrosine”. The constants “h” and “y” represent “Histidine” and “Tyrosine”, respectively. $A$ and $C$ are variables that are matched against a particular example; $A$ indicates an example identifier and $C$ the position at which the mutation occurs. Furthermore, consider the clause

$$resistant(A, nrti) \leftarrow mut(A, h, C, y), position(C, 208)$$

encoding that a mutation resulting in a change from “Histidine” to “Tyrosine” and occurring at position 208 entails resistance to the drug *nrti*. Such a clause can be matched against an example by grounding, and if the matching operation is successful the clause is said to *cover* the example.

Let $\mathcal{D}$ be a set of positive and possibly negative examples in the form of true and false facts and a background knowledge $\mathcal{B}$ as a set of definite clauses. The learning problem consists of searching for a set of definite clauses, the hypothesis $H^* = \{c_i, ..., c_m\}$, covering all or most positive examples, and none or few negative ones if available. First-order clauses can therefore be interpreted as relational features or rules that characterise the target concept. An example is classified as positive by the hypothesis if it is covered by one of its clauses. More formally, $\mathcal{B} \cup H \models x$, which means that the example is logically entailed by the hypothesis and the background knowledge.

Finding $H^*$ consists of solving the maximisation problem:

$$H^* = \max_{H \in \mathcal{H}} S(H, \mathcal{D}, \mathcal{B}) \tag{1.15}$$

where $S$ is a scoring function for evaluating the candidate hypothesis, e.g. the accuracy.

There are different approaches to searching for an (approximately) optimal set of clauses within a pre-defined hypothesis space $\mathcal{H}$ called the *language bias*. A central idea in most ILP systems is to structure the search space $H$ according to generality. A hypothesis $G \in H$ is called more general than another hypothesis $S \in H$, denoted $G \preceq S$, if all examples covered by $S$ are also covered by $G$. The generality relation induces a lattice on the hypothesis space $H$, and thus provides a way to systematically search $\mathcal{H}$. A popular approach is to search the lattice top-down, that is, from general to specific hypotheses, using a *refinement operator*. A refinement operator $\rho$ takes a clause $c$ and returns all specialisation $c' \in \rho(c)$ of the clause that fall within the language bias. In the simplest case, these specialisation (or *refinements*) are obtained by simply appending a literal to the clause $c$. For example, the clause $c' = \leftarrow mut(A, h, C, y), position(C, 208)$ is a refinement of the clause $c = \leftarrow mut(A, h, C, y)$. Note that $c$ will match any example matched by $c'$; thus, a hypothesis in which the clause $c$ is replaced by the clause $c'$ becomes more specific.

Top-down search based on refinement operators is the main principle underlying many ILP algorithms. For instance, incremental rule learners such as the well-known FOIL algorithm greedily search for a set of clauses that covers all positive examples by performing a hill-climbing search in the hypothesis space using a refinement operator [115].

The main advantages of logic-based learning with respect to other machine learning techniques is the expressivity and interpretability of the learnt models and the possibility to make use of specific background knowledge. Models can be readily interpreted by human experts and provide direct explanations for the predictions. Furthermore, they provide the ability to deal with complex structured data and to learn relations among substructures.

### Statistical Relational Learning

Statistical Relational Learning (SRL) techniques (see [64] for a broad introduction) combine the advantages of Inductive Logic Programming (ILP) and Statistical Learning, namely the ability to learn a model of the concepts that can be readily interpreted by a human domain-expert and, the robustness principles of the statistical learning theory. These characteristics make these techniques very appealing for bioinformatic tasks.

In [87] a framework for the integration of the two approaches is defined. It consists in solving an integrated optimisation problem:

$$\max_{H \in \mathcal{H}} \max_{f \in \mathcal{F}_H} S(f, \mathcal{D}, \mathcal{B}). \tag{1.16}$$

Here, $\mathcal{H}$ denotes the logical hypothesis space under consideration, i.e. the set of all possible sets of clauses. $f$ is a function chosen in the set of all possible functions $\mathcal{F}_H$ that can be represented based on the hypothesis $H$ and it maps input examples into the outputs $f(x; H, \mathcal{B}) : \mathcal{X} \to \mathcal{Y}$. $S$ denotes a scoring function that measures the predictive performance of $f$ on the training data $\mathcal{D}$, and $\mathcal{B}$ the available logical background knowledge.

The learning problem consists of jointly optimising the logical hypothesis $H$ and the function $f(x; H, \mathcal{B})$ [87]. Indeed, an outer and an inner optimisation problems can be identified in this formulation. In the following, we will refer to the outer optimisation problem as *hypothesis learning* and the inner optimisation problem as *function learning.*

Hypothesis learning implies searching in a discrete space of candidate solutions, which is a complex task. Thus, heuristic strategies are employed. In contrast, function learning takes place in a continuous space, for which principled search techniques are available. It is thus unclear whether scoring functions employed for function learning are also suitable for hypothesis learning. Statistical relational learning systems often employ different scoring functions for learning the logical model structure and the statistical part of the model. Problem (1.16) should therefore be generalised to the following formulation:

$$\max_{H \in \mathcal{H}} S_O \left( \arg\max_{f \in \mathcal{F}_H} S_I(f, \mathcal{D}, \mathcal{B}), \mathcal{D}, \mathcal{B} \right) \tag{1.17}$$

where $S_O$ and $S_I$ are the scoring functions used for hypothesis and function learning respectively (see [87] for a more detailed discussion).

An simple example of defining of function $f$ and $S_I$ is to replace the logical coverage notion with:

$$f(x; H, \mathcal{B}) = \begin{cases} 1 & \text{if } \mathcal{B} \cup H \models p(x) \\ -1 & \text{otherwise} \end{cases} \tag{1.18}$$

and to define a scoring function that corresponds to the opposite of the empirical risk:

$$S_I(f, \mathcal{D}, \mathcal{B}) = -\frac{1}{m} \sum_{i=1}^{m} \mathbb{I}[f(x_i; H, \mathcal{B}) \neq y_i] \tag{1.19}$$

where the indicator function compute the 0-1 loss. The same considerations made in the previous sections can be applied for solving this inner optimisation problem.

The defined SRL framework fits a number of settings proposed in the literature as discussed in [87]. For example Stochastic Logic Programs [99], Bayesian Logic Programs [79], Markov Logic Networks [83] and kFOIL [86].

### 1.4.4 Measuring and Comparing the Performance of Learning Algorithms

Evaluating the performance of a learning algorithm and comparing it with those of other learning algorithms requires to define performance measures. Here is a list of the most used in machine learning that are also used for evaluating the approaches proposed in this thesis:

- *Accuracy* $= \frac{t^+ + t^-}{t^+ + f^+ + t^- + f^-}$
- *Precision* $= \frac{t^+}{t^+ + f^+}$ (P)
- *Recall* or *Sensitivity* or *TP rate* $= \frac{t^+}{t^+ + f^-}$ (R)
- *FP rate (1-specificity)* $= \frac{f^+}{t^- + f^+}$ (FPR)
- $F_1$ *measure*, $F_1 = \frac{2 \times Precision \times Recall}{Precision + Recall}$ ($F_1$)
- *Matthews Correlation Coefficient* $= \frac{t^+ t^- - f^+ f^-}{\sqrt{(t^+ + f^+)(t^+ + f^-)(t^- + f^-)(t^- + f^+)}}$ (MCC)
- *Area Under the Receiver Operator Characteristic (ROC) Curve* (AUCROC)
- *Area Under the Recall/Precision Curve* (AUCRP)

where $t^+, t^-, f^+, f^-$ are the true positives, true negatives, false positives and false negatives respectively. $F_1$ is the harmonic mean between *Recall* and *Precision*, giving equal weight to the two complementary measures. ROC and RP curves and their areas provide a broader picture of a classifier performance, as they do not require to choose a fixed decision threshold to discriminate positive and negative examples, but evaluate all possible thresholds. For highly skewed datasets, the area under the RP curve is more informative than the area under the ROC [48].

## 1.5 Aspects of Innovation

By drawing on solving the problem of determining the molecular function of an enzyme, this thesis aims at investigating and developing effective and advanced machine learning approaches that could help in the identification of novel protein functions and understanding the mechanisms allowing an enzyme to operate.

The proposed approaches face satellite tasks all aiming at solving the complex problem of protein function discovery and characterisation, approaching it from two main different viewpoints: the target protein structure and its interactions with other molecules, and the analysis of the target protein mutation data.

The first main contribution of the thesis is the realisation of a predictor of functional residues and the active site they belong to, starting from either sequence or structural information. The identification of enzyme functional residues is fundamental in applications like molecular docking and de novo drug design, the engineering of new drugs fitting a certain function. The prediction of functional residues can be very useful for guiding site-directed mutagenesis experiments or inhibition studies. Both techniques are used to validate a formulated hypothesis about the protein molecular function and eventually produce enzyme mutants with improved activities.

The predictors take a discriminative learning approach and are based on support vector classifiers. The proposed structure-based predictor is able to significantly improve state-of-the-art predictive performance on the functional residue learning task [158]. This result is obtained thanks to the development of an effective representation of the structural information that models spherical regions around candidate residues and extracts statistics on the properties of their content. The statistically significant improvement is obtained on six different benchmark datasets that, in the whole, cover a set of more than seven-hundred enzymes in the PDB.

With the aim of collectively predicting all the residues belonging to the same functional site, a structured-output learning approach is proposed [162]. The problem of detecting functional sites in the protein is formulated as the problem of identifying groups of functional residues composing them.

Two slightly different distance-based supervised clustering approaches are devised for sequence and structure-based prediction. In the case of sequence-based prediction training proteins are employed to learn a proper distance function between residues. The learning stage simply consists of training a pairwise support vector machine to obtain a classification function predicting for each pair of residues whether they belong to the same site. For structure-based prediction the distance function is the Euclidian distance between residues. Residue weights are also considered, that correspond to their catalytic propensity as predicted by the previously built state-of-the-art support vector predictor.

In both prediction settings, after the learning stage, a partial clustering is returned by searching for maximum-weight cliques in the resulting weighted graph representation of proteins. A stochastic local search algorithm based on tabu search is employed to efficiently generate approximate solutions.

The rationale for the approach is that given a reasonable pairwise similarity measure, the algorithm should isolate few densely connected components which correspond to the desired solution while discarding most of the nodes in the graph. The proposed method is shown to achieve significant improvements over the support vector classifier performance alone. Therefore, in the case of structure-based prediction, it further improves over the state-of-the-art performance on the largest available benchmark dataset.

This thesis also presents CatANalyst a web server, freely accessible at

`http://catanalyst.disi.unitn.it`

which provides the described functional residue prediction service to the public of researchers.

The CatANalyst predictor is shown to be useful in real case studies. First, for discovering residues that could affect the enzyme active site. This is done by analysing mutants of an amidase enzyme [45], generated by random mutagenesis experiments, and correlating their functional site predictions with those of the wild type enzyme. Second, for directly predicting functional residues of the same enzyme starting from its predicted 3D model [154].

The second main contribution of the thesis is in the investigation of statistical relational learning approaches for mining relevant feature from the protein mutation data — for example those of the reverse transcriptase, an HIV enzyme which is essential for the success of the viral propagation. Mutagenesis data produced need to be analysed to gain insights about the properties of a functional site and of the residues involved. The mined information can be used to engineer mutants of the enzyme with an improved activity on a certain substrate or, as in the case of HIV, to predict the virus evolution and its response to specific drugs and devise the appropriate countermeasures.

Two statistical relational learning approaches are proposed. The first one is a hierarchical multitask statistical relational approach [155]. kFOIL is chosen as statistical relational learner, which combines techniques from inductive logic programming —– specifically, the FOIL algorithm [115] —– with kernel methods.

A hierarchical extension of the multitask kFOIL algorithm is proposed for mining relevant relational features explaining the resistance of enzyme mutants to certain inhibitors, as in the case of HIV reverse transcriptase. The activity or resistance of an enzyme and its mutants is usually experimentally evaluated on a number of different substrates (drugs). Exploiting the information on related classification tasks can be useful for improving predictive performance. Multitask classification is here useful for learning models that are general and can benefit from tasks relatedness. Drugs are usually organised in groups depending on their phisico-chemical characteristics or their target sites. In order to exploit these relationships, the kFOIL extension proposes a hierarchical approach that first learns a core logic representation common to all tasks, and then refines it by specialisation on a per-task basis. The approach can be generalised to a deeper hierarchy of tasks, for instance, when learning the resistance of a mutant to an inhibitor or a specific class of inhibitors. Experimental results show the advantage of the hierarchical version over both single and multi task alternatives and its potential usefulness in providing explanatory features for the domain.

A clustering approach is also proposed for learning the latent structure among related tasks when this is not known [156]. Task clustering allows to further improve the predictive performance. The major advantage of this hierarchical strategy is the ability to provide explanations for the learnt models which are themselves hierarchical: a subset of relational features relevant to all tasks can be identified together with more specific task-dependent ones.

A relational learning approach is also proposed [160], which aims at employing the mined features for engineering novel enzymes with useful functions. First, mutation data are used to learn a set of relational rules characterising the resistance to a certain inhibitor. Then, these rules are used for generating a set of candidate mutations that are most probable to confer drug resistance. At the author's knowledge this is the first attempt to learn relational features of mutations affecting a protein behaviour and use them for generating novel relevant mutations. The approach is tested on a dataset of HIV drug resistance mutations, comparing it to a baseline random generator. Statistically significant improvements are obtained on both categories of nucleoside and non-nucleoside HIV reverse transcriptase inhibitors. The promising preliminary results suggest that the proposed approach for learning mutations has a potential in guiding mutant engineering, as well as in predicting virus evolution in order to try and devise appropriate countermeasures. The approach can be generalised to learn mutants characterised by more complex rules correlating multiple mutations as those learnt with the kFOIL.

It is worth to underline that all the techniques proposed in this thesis can be potentially applied to many other learning problems and they are not confined to the enzyme characterisation. The same discriminative learning approaches and representation implemented in CatANalyst can be, for instance, used for predicting binding residues at the interfaces of protein-protein interactions. The structured output learning approach is actually shown to be effective also for predicting metal binding sites. The hierarchical relational learning approach can be easily applied to other learning tasks. In particular it is shown to be useful for classifying protein structures into folds. Finally, the generative approach, aims in general at modifying complex structures to improve certain properties, by exploiting statistical relational learning techniques to efficiently constraint the search space.

## 1.6 Structure of the Thesis

The rest of thesis is composed of two main parts. The first part focuses on the prediction of enzyme functional residues and functional sites they belong to. Chapter 2 introduces the problem of functional site identification in enzymes and Chapter 2.2 presents the most popular approaches used for the detection of functional residues and functional sites. The proposed approaches are presented in the subsequent chapters. First, a support vector learning approach is described (Chapter 3) and the web server CatANalyst implementing it is presented (Chapter 5). Chapter 4 reports details on the distance-based supervised clustering approach for the collective prediction of the functional residues and the active site they belong to.

The second part focuses on mining relevant relational features from enzyme mutation data to gain insights about the characteristics of residues that can affect an enzyme function. First, an introduction to the topic of learning from mutations is given (Chapter 6), followed by an overview of the state-of-the-art methods for learning from mutation data (Chapter 6.2). Subsequent chapters presents two related approaches. The first, in Chapter 7, proposes a hierarchical multitask statistical relational learning approach for learning from mutation data exploiting the tasks relatedness and structure. The second, in Chapter 8, starting from a relational model of a mutation or a mutant aims at generating novel instances satisfying the learnt model.

# Part I

# Functional Residue Prediction

# Chapter 2

# Identifying Functionally Important Residues

In this part of the manuscript the focus is on the prediction of residues composing an active site. The aim is characterising the function of enzymes, but the proposed approach can be easily generalised to the other kind of proteins.

## 2.1 Introduction and Background

Enzymes are those proteins whose role is to catalyse chemical reactions inside a cell. The enzyme works by forming complexes with the reactants and in doing so it lowers the activation energy of the reactions thus increasing their rate. This process is the catalysis. The simplest reaction one can think of works in the following way: the enzyme (E) forms a complex with the reactant, namely a substrate (S), which is usually a small molecule. The resulting complex (ES) is then transformed into (EP) the product of the reaction (P) plus the free enzyme (E), which is again ready for reacting with another molecule (S).

$$E + S \Longleftrightarrow ES \Longleftrightarrow EP \Longleftrightarrow E + P$$

By analysing enzyme crystal structures, two constant elements of the molecular conformation of the active sites emerge. The first is the presence of a “pocket”, which is devoted to receive the substrate for modifying it. The second is the presence of chemical groups with catalytic function located in specific positions of the molecule.

These residues that are directly involved in the catalytic process constitute the active site, while residues in the surrounding space play the role of attracting and orienting the molecule to bind, and constitute the binding domain. The first kind of residues is of interest in the present study. In the following they are referred to as functional or catalytic residues. For instance, two enzymes that are able to bind the same substrate can modify it in different ways. This implies that they have different active sites and perform different functions.

A graph representation of a catalytic triad in an enzyme is shown in Figure 2.1. The vertices of the graph represent amino acids connected in a chain by peptide bonds (grey edges). Functional residues are highlighted in red. Even if they are far apart along the primary sequence:

$$D_r \dots D_2 D_1 A_1 B_1 B_2 \dots B_n A_2 C_1 C_2 \dots C_m A_3 E_1 E_2 \dots E_s$$

they become close one to each other when the polypeptide chain folds in the tertiary structure. The proximity in the 3D space allows the residues to reciprocally interact and/or to interact with the substrate and other molecules, for example water.

Identifying the characteristics that allow to discriminate a functional residue from a non-functional one is quite difficult. Local structural information can be useful for characterising the catalytic core in which functional residues are typically buried. Residue conservation can be another important discriminative feature. Usually functional residues are more conserved than other across evolution because the protein function should

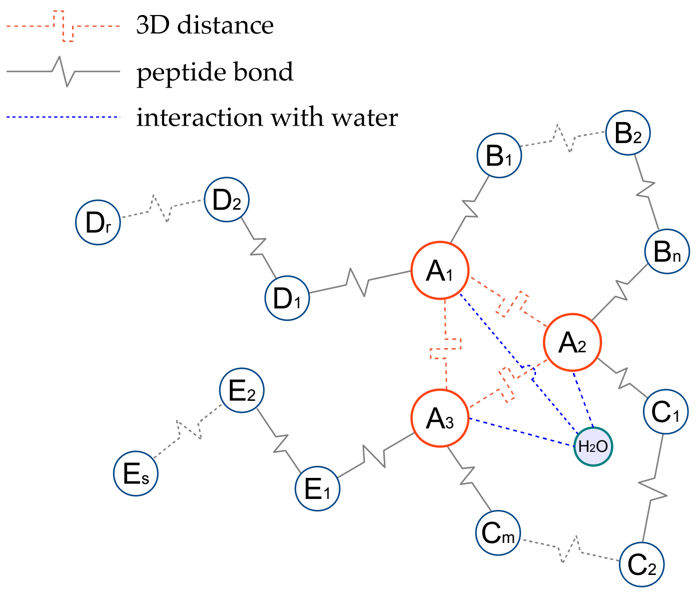


Figure 2.1: Active site schematisation.

be preserved.

In order to accomplish their biological function, enzymes often interact also with different types of external molecules, called cofactors (see section 1.2.2). Some examples of cofactors are: metal ions, prosthetic groups and various organic compounds.

The present study adheres to the definition of enzyme catalytic residue proposed in [7], and here reported. A residue is functional if one of the following cases is verified:

1. the residue is directly involved in the catalytic mechanism (e.g. nucleophiles, proton-donors).

2. the residue exerts an effect on another residue or water molecule which is directly involved in the catalytic mechanism which aids catalysis (e.g. by electrostatic or acid–base action).

3. the residue stabilises a proposed transition-state intermediate.

4. the residue exerts an effect on a substrate or cofactor which aids catalysis, e.g. by polarising a bond which is to be broken. Includes steric and electrostatic effects.

Therefore, according to this definition solely binding substrates, cofactors or metals, which are often involved in enzymatic reactions, does not characterise a residue as catalytic. The same definition is applied in the Catalytic Site Atlas (CSA) [113], the reference database for functional residue annotations.

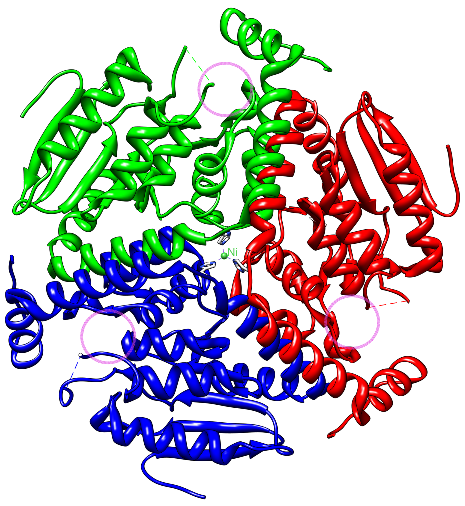
Metal ions are a category of cofactors that require special attention in this context. Many of them are crucial for the catalytic activity of an enzyme. Other are important for stabilising its three-dimensional structure and inducing conformational changes as in other metalloproteins [16].

In methylmalonyl coa decarboxylase [13] the nichel ion has the structural role of holding the chains together as shown in Figure 2.2(a). The catalytic pockets are located in different places of the polypeptide chains highlighted with purple circles.

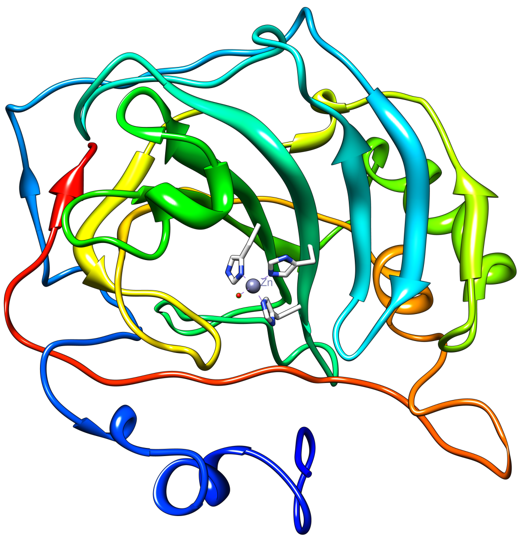
In a metalloenzyme like the carbonic anhydrase (E.C. 4.2.1.1) — which catalyses the conversion of carbon dioxide ($CO_2$) and water in bicarbonate ($H_2CO_3$) — a zinc ion is essential for the catalytic mechanisms (see Figure 2.2(b)). The zinc ion in the active site is coordinated by three hystidine residues. The zinc ion polarises the water molecule and in this way allows for the acceleration of the nucleophilic attack of the carbon dioxide.

Predicting metal binding sites and the residues that compose them is a a task strictly related to the identification of enzyme catalytic sites. In the example given above, the hystidines just coordinate the zinc ion, which has a catalytic role. The three hystidine residues are part of a metal binding site but their are not functional residues.

Discriminating among functional and metal binding residues is quite difficult and the two tasks have been always tackled separately.



(a) Methylmalonyl coa decarboxylase (PDB code 1EF8)



(b) Human carbonic anhydrase (PDB code 1CA2)

Figure 2.2: Roles of metal ions in enzymes. (a) The metal binding site and the catalytic sites are located in different parts of the polypeptide chain. The nichel ion has a structural role. (b) The zinc ion binding site is located in the catalytic pocket, which also contain a water molecule.

In this part of the thesis two approaches are proposed for the identification of functional residues and the active sites they belong to respectively. The approaches are depicted in Figure 2.3.

All of them are based on support vector learning. The first one realises a standard binary classification tasks for predicting whether a residue has a functional role within a protein. A linear classifier is built based on features extracted from the protein sequence and structural information. The structure based approach exploits the protein structural information by representing the spherical neighbourhood of a residue. The last one is an approach for the collective prediction of functional residues in the same active site. It exploits engineered features and classifiers developed for the previous approaches and, building upon them, it mines densely connected

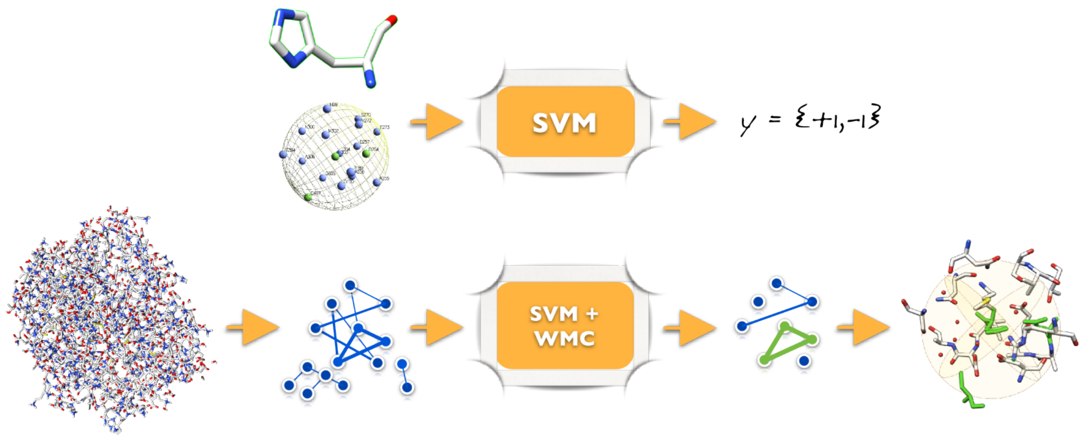


Figure 2.3: Three different proposed approaches for active site identification.

clusters of functional residues in the protein. The approach is also applied to the related task of predicting metal binding sites.

## 2.2 State of the Art

Traditional approaches to functional site identification use homology-based strategies. Novel protein function is inferred by aligning the sequences or by superimposing the structures with already annotated proteins.

The research work in [154] is an example of approach tipically used by biologists for discovering active sites in a protein based on in vitro experiments guided by homology with other annotated proteins. The hypothesis being investigated is the presence in an amidase enzyme (EC 3.5.1.4) of a catalytic site similar to that of nitrilases (EC 3.5.5.1). The hypothesis is suggested by laboratory experiments and by the predicted model of three-dimensional structure of the enzyme, and then confirmed by site-directed mutagenesis experiments and by inhibition studies. Characterising the enzyme functional site requires months of study and the expertise of researchers and biologists on a single target protein at a time. Automatic approaches can significantly speed up the whole process.

In [96], active sites of non annotated proteins in the Pfam database [58], which contains about 8,200 protein families, are predicted by using a rule-based technique which exploits the homology and sequence similarity with other annotated proteins. The methodology is based on the transfer of experimentally determined active site data to other sequences within the same Pfam family. The authors show that it is possible to gain functional annotation of a large number of sequences in the Pfam database (enzymatic families) for which the residues responsible for catalysis have not been determined.

By using family-based resources like Pfam, which cluster sequences into evolutionary families, it is possible to annotate a large number of proteins. However, even within the same family proteins can perform many different functions, and an accurate annotation is not possible without manual validation and automatic approaches for the detection of the specific function-discriminating residues [90]. A further issue that could rise, even with a high percentage of sequence identity, is the erroneous annotation of multi-domain sequences [135].

By knowing the protein three-dimensional structure it is possible to overcome many of the problems rose by sequence similarity, as it is better preserved with respect to the sequence during evolution. However, an annotated homologue of the target protein needs to be available, preventing the applicability of homology-based strategies to novel folds.

A number of researchers have recently tackled the problem of functional residues prediction. In [95] we can see the first general approach to structure searches based on local characteristics of the active site rather than exclusively on the overall fold similarity. The authors showed that a three-dimensional signature consisting of only a few functionally important residues can be diagnostic of membership in a superfamily of enzymes which can represent a first step in the inference of some functional properties. This membership results not only from fold similarity but also from the disposition of residues involved in a conserved function.

In [82], the authors propose a methodology for predicting active sites from the protein structure. The proposed methodology combines statistical analysis with the extraction of information from the shapes of the theoretical titration curves calculated for the ionizable residues in a protein. The authors observe that residues with significantly perturbed predicted titration behaviour tend to occur in the active site of an enzyme and in binding sites of proteins. This phenomenon is observed to such a great extent that the mentioned residues serve as revealing markers of reactivity and recognition. The algorithm is able to identify these perturbed residues (positive residues) and searches for clusters of two or more such residues in physical proximity. These clusters are considered accurate predictors of interaction sites. The method, as the authors underline, is independent from structural comparisons thus can be applied to proteins with few or no known close homologues.

In [103] the authors generated three-dimensional templates of protein active sites with rigid prosthetic groups. Their approach is based on the simultaneous alignment of several protein structures, and relies on local atomic-level similarities based on multiple comparisons. The generated patterns include 3D atomic coordinates, position of chemical groups, and cavity locations. However the approach remains limited to the subset of proteins having rigid prosthetic groups.

Machine learning approaches to the automatic identification of protein functional residues have been investigated in recent years. The problem has been mostly formalised as a binary classification task at the residue level. These algorithms provide a valid alternative when the homology-based transfer cannot be applied, thanks to their capacity of providing the contribution of specific attributes to the classification. They can be very useful for suggesting candidate residues to be experimentally verified and potentially allowing automatic annotation of functional residues.

Petrova and Wu [111] and Youn et al. [149] have addressed the functional residue prediction with a Support Vector Machine (SVM) fed with both protein sequence and structural properties. Capra and Singh [31] relied on an information-theoretic approach for estimating sequence conservation. The authors show that conservation of sequentially close residues improves predictive performance, especially when catalytic residues are involved. In [59], carefully crafted conservation scores were shown to play a major role in predictive performance. Closeness centrality measures [37] have been used in [128] to improve catalytic residue prediction by using neural networks trained with a genetic algorithm. A review on approaches and applications for structure-based protein function prediction can be found in [66]. A recent study [151] showed that sequence information alone could provide results similar to those obtained by previous structure-based methods. This seems to indicate that much work has still to be done in order to fully exploit the information contained in protein three-dimensional structures. Recent approaches investigated the use of topological [145], electrostatic [142] and graph theoretic [2] structured-based features for predicting ligand binding sites or protein functional sites. THEMATICS [105] electrostatic features and geometric features are combined with sequence conservation features in a maximum likelihood approach called Partial Order Optimum Likelihood (POOL) [134]. The authors underline the key role of THEMATICS features which are extracted from the residue theoretical titration curves.

Few servers for functional residue prediction exist. They mostly rely on primary sequence only (CPred [151], FRpred [59]) or evolutionary analysis (INTREPID [122]). However, correctly identifying functional residues is an extremely challenging task even when the protein tertiary structure is known. Available structure-based online predictors focus on the identification of the pocket in which active site residues could be located [89], or allow for searching the protein structure for a limited set of three-dimensional motifs [67]. THEMATICS approach is an example of structure-based predictor that characterises each single residue with electrostatic features extracted from the titration curves [142].

Prediction of metal binding sites is a task closely related to the functional residue prediction task. Both problems are fundamental for understanding the role of residues in the active site. Residues are sometimes involved in the catalytic process and also in binding a metal ion, in other cases binding a catalytic metal ion does not implies that the residue has a functional role.

The task of metal binding site prediction has been mostly addressed as a binary classification task at the residue level: given a protein sequence, predict for each residue whether it is involved in a metal binding site [107], [126]. Most existing approaches for modelling the full metal binding geometry assume knowledge of the 3D structure of the protein [3, 54] and focus on detecting apo-proteins, i.e. proteins solved without the ion. A recent attempt [60] to predict metal binding geometry from sequence formulates the problem as a structured-output task. The proposed solution is a search algorithm greedily assigning residues to ions (or a default *nil* ion if predicted as free) guided by a scoring function trained to rank correct moves higher than incorrect ones. The algorithm is guaranteed to find the solution maximising the overall score, given the matroid structure of the problem. However, the scoring function is learnt from examples and there is no guarantee that it correctly approximates the true underlying function.

# Chapter 3

# A Support Vector Learning Approach

This chapter describes a support vector learning approach for functional residue prediction [158]. It shows how to effectively employ the protein 3D structure information by modelling the structural neighbourhood of candidate residues, represented as a sphere centred on the residue side chain. Such neighbourhood information is encoded with statistics on the properties of its content, such as physico-chemical properties, atomic density, flexibility, presence of water molecules.

A support vector machine is trained combining the structural neighbourhood features with evolutionary enriched sequence information as well as previously developed 3D features [111]. The structure-based method achieves improvements over both sequence-based and structure-based state-of-the-art predictors, as measured on a set of benchmark datasets with varying characteristics, and structural neighbourhood information is shown to be responsible for such improvements.

The additional investigation of the role of ligand information in presence of heterogen molecules, playing possible catalytic or structural roles, shows that exploiting such information in both sequence-based and structure-based active site predictions is an interesting direction for further research.

## 3.1 The Learning Task

Functional residue prediction can be cast into a binary classification task at the residue level, namely predicting for each residue of a given protein, whether it is directly involved in the catalysis or not. The learning task is addressed with an SVM and devising a residue vectorial representation that exploits information related to the properties of the local structure surrounding a residue.

## 3.2 Engineering Residue Features

Different sets of features are extracted from both primary and tertiary protein structure in order to represent candidate residues. Tables 3.1 and 3.2 summarise extracted sequence and structural features respectively.

### 3.2.1 Features Extracted from the Sequence

The features extracted from the primary sequence encode characteristics of the target residues and evolutionary information (see Table 3.1):

| **Features** | **Description** |
|---|---|
| $1D_1$ | Target amino acid name |
| $1D_2$ | Target amino acid type |
| $1D_3$ | Conservation profiles |

Table 3.1: Sequence-based Features Representation: features extracted from the protein sequence

$1D_1$ encodes the amino acid name of the residue.

$1D_2$ encodes the amino acid type of the residue based on its physico-chemical properties: H, R, K, E, D as charged; Q, T, S, N, C, Y, W as polar and G, F, L, M, A, I, P, V as hydrophobic [7].

$1D_3$ encodes evolutionary information in the form of multiple alignment profiles.

$1D_1$ and $1D_2$ are categorical (or nominal) attributes, and are encoded one-hot: each attribute is encoded with a vector of bits of size equal to the number of possible attribute values; value $k$ is encoded with a vector having one at position $k$, and zero at all other positions. $1D_3$ is a real vector of conservation profiles computed from multiple alignments. We performed a two iteration Position-Specific Iterative Blast Search (PSI-Blast) [1] on a database of non-redundant protein sequences (nr) (downloadable from ftp://ftp.ncbi.nlm.nih.gov/blast/db/). A threshold of $5 \cdot 10^{-3}$ on the expectation value was employed for both initial iteration and extending hits. We enriched the profile extracted from the multiple alignment with two values indicating its informativeness and reliability, namely profile entropy and weight of the conservation profile with respect to pseudocounts.

### 3.2.2 Modelling a Residue Structural Neighbourhood

We represent a residue in the 3D space as a single representative point, the centroid of its side-chain atoms (point SC in Figure 3.1), since such atoms are more likely to be involved in the catalysis. The single representative point of a glycine residue is the carbon-alpha ($C_\alpha$) atom.

Given such a 3D representation of residues, we define the structural neighbourhood of a residue $x$ as the set of residues and molecules contained in the volume of a sphere centred on $x$ ($x$ will be a target residue in our setting).

One can consider spherical regions of different radius. The radius of the sphere is fixed to a maximum of 8 Å which is the maximum interaction distance between a residue and a water molecule. The rationale behind this choice is that the interaction with a water molecule is very important

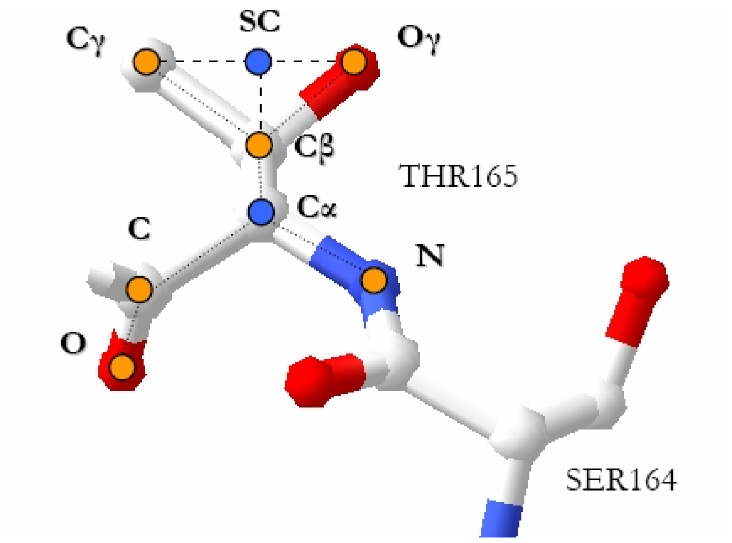


Figure 3.1: A residue 3D representation: point SC is the side-chain centroid, which we used as the residue representative point.

for the catalysis in enzymes like the hydrolases.

As an example, Figure 3.2 shows the crystal structure of L-arginine glycine amidinotransferase (PDB code 1JDW), a mitochondrial enzyme involved in the creatine biosynthesis. The catalytic pocket is highlighted and the catalytic triad of residues is shown: ASP254, CYS407, HIS303 [73]. The cysteine is the nucleophile and binds the carbon on the substrate (arginine) side chain. The histidine activates the substrate to deprotonate CYS407 and deprotonates glycine, while the aspartic acid primes the histidine by activating water, a cofactor or a residue. In Figure 3.3(a) we show the 8 Å sphere centred on the HYS303 residue: the sphere contains all active site residues (shown in green). In Figure 3.3(b) we show the same sphere with residues represented with their side-chain centroids.

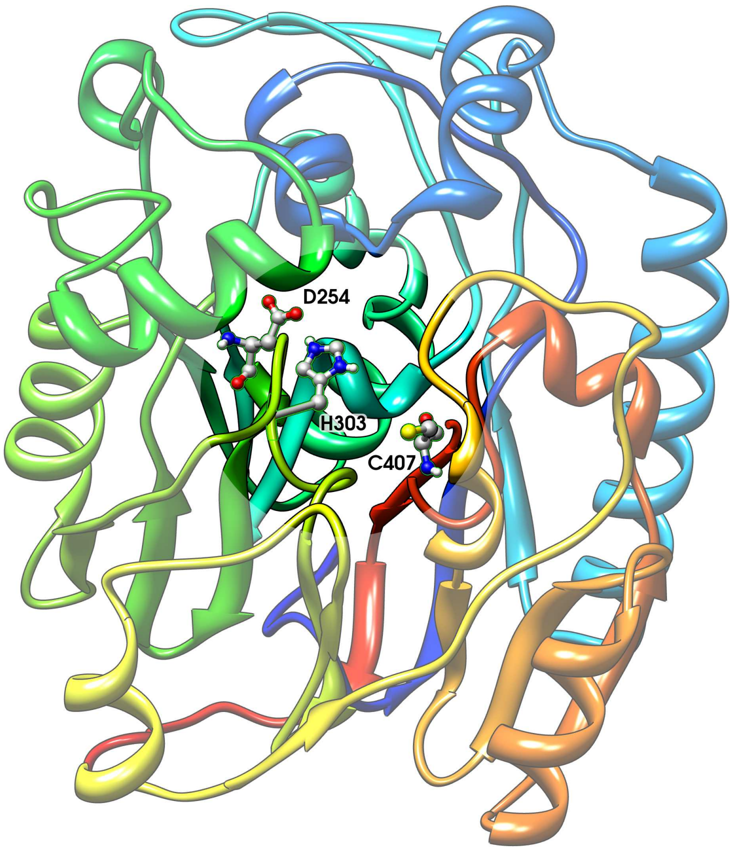
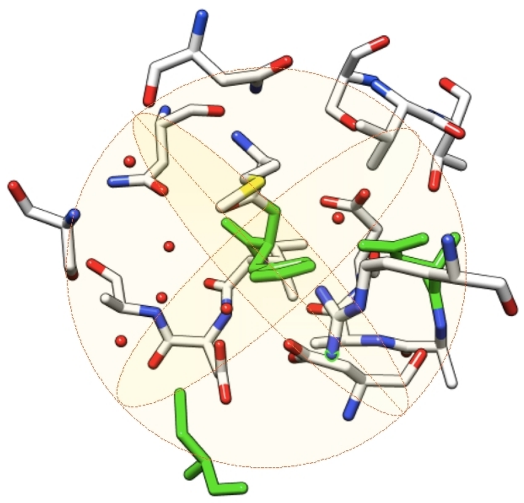
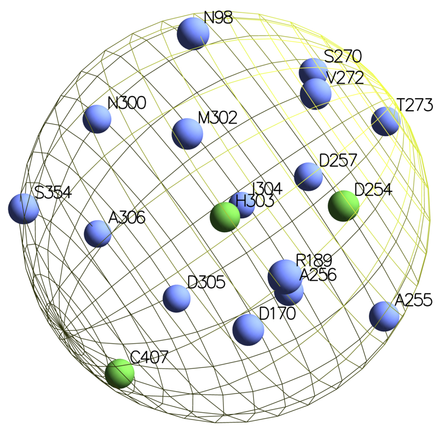


Figure 3.2: The L-arginine:glycine amidinotransferase (1JDW) and its highlighted catalytic pocket.



(a) Active site 8 Å sphere of L-arginine:glycine amidinotransferase (1JDW) centered on catalytic residue HYS303. The three catalytic residues are shown in green.



(b) Side-chain centroids within the 8 Å sphere of L-arginine:glycine amidinotransferase (1JDW) centered on HYS303. The three catalytic residues are shown in green.

Figure 3.3: A residue structural neighbourhood.

### 3.2.3 Features Extracted from the Structure

Features characterising a residue can be extracted from the protein structure if available. We showed (see section 3.3) that extracting features from a residue neighbourhood, thus exploiting the locality of the protein structure, can be useful to discriminate between functional and non functional residues. Table 3.2 summarises the scalar features we extracted from the residue 3D neighbourhood. The first group contains statistics on the properties of the neighbourhood content, while the second encodes information on possible ligands contained in the neighbourhood. Each row in the table corresponds to an attribute or a set of attributes encoding the properties specified in the description. In the following we provide a detailed description of such features.

In Figure 3.4 we provide an example of feature vector extracted from the 3D-structural neighbourhood of the target residue (GLU 988 of the PDB protein structure 1A26).

| **Features** | **Description** |
|---|---|
| $3D_1$ | Physical and chemical properties (amino acid attributes) |
| $3D_2$ | Amino acidic composition |
| $3D_3$ | Charge/Neutrality |
| $3D_4$ | Water molecule quantity |
| $3D_5$ | Atomic density |
| $3D_6$ | Flexibility B-factor |
| $3D_7$ | Disulphide bond |
| $3D_8$ | Heterogens |
| $3D_9$ | Cofactor binding |

Table 3.2: Structured-based Features Representation: scalar features extracted from the residue structural neighbourhood.

*Statistics of the Residue Structural Neighbourhood Properties*

The first set of features encodes aggregate values representing properties of the atoms included in the sphere.

$3D_1$ encodes chemical and physical properties of the residue neighbourhood. This set of attributes represents properties such as hydrophobicity, polarity, polarizability and Van der Waals volume of the neighbouring residues. They are encoded in a three bin distribution (normalised number of residues with low, medium, high hydrophobicity, polarity, polarizability and Van der Waals volume) according to the indices reported in the Amino Acid Index Database [77]. The same encoding was used in [22] for protein function classification.

$3D_2$ encodes the amino acid composition of the 3D sphere, represented as the frequency of occurrence of each one of the twenty amino acids.

$3D_3$ represents charge or neutrality of the 3D sphere, encoded into three values: the number of positively charged residues, the number of negatively charged residues and their sum.

$3D_4$ encodes the quantity of water in the sphere, measured as the number of water molecules within the sphere radius. This group of attributes is motivated by the fact that an active site is usually located in a hydrophobic core of the protein, while on the surface the quantity of water is higher and the residues exposed to the solvent are not hydrophobic.

$3D_5$ measures the atomic density of the sphere, calculated as the total number of atoms it contains.

$3D_6$ represents the residue temperature factor (B-factor), as a measure of the residue flexibility. It is calculated as the average of the atomic B-factors of atoms composing the residue, normalised over the whole protein. As the temperature factor could depend on the crystal structure, normalising over the whole protein helps to exclude the variations that can be present among different protein crystal structures. Note that in [111] an unnormalized version of the residue B-factor was employed instead.

*Ligand Features*

In oxidising environments, cysteines tend to form covalent bonds called disulphide bridges, which help stabilising the 3D structure of the protein. Disulphide bonded cysteines are usually not involved in the catalytic process: in the PW dataset of 79 enzymes the only exception is given by a protein disulphide isomerase (PDB code 1MEK). It has two catalytic cysteine residues in a thioredoxin domain similar to one of the well-known thioredoxin proteins. We encoded information on bridges by a flag ($3D_7$) indicating whether the target residue is a disulphide bonded cysteine.

Enzymes often employ cofactors in order to help interacting with the substrate. Therefore, the presence of a cofactor in the structural neighbourhood of a certain residue is an indication that the area could be an active site. On the other hand, many heterogens bind residues for structural rather than catalytic purposes, like NI in the methylmalonyl coa decarboxylase (PDB code 1EF8) [13] which is involved in trimerization.

The Het-PDB Navi database [148] provides information on a large set of small molecules found in the protein structures of the PDB. For example information about the reaction in which the cofactors, substrates and products are involved, and the cofactor interface propensity. A description of the mechanisms of the catalysis is included in the CSA functional annotations whenever such information is available. It describes the role of the cofactors and which are the substrates and products of the reaction.

In the dataset that we used for the feature engineering, 51 out of 79 enzyme structures contain heterogen molecules. For the remaining structures we can not say whether they are apoenzymes or they just do not require any help from cofactors during the catalysis. In the former case, methods for predicting metal-binding sites in apo protein structures [3] may be used to identify the presence of possible cofactors.

In Figure 3.5 we show a histogram of the most frequent heterogens we found in the PW dataset. Each one of those heterogens appears at least in two protein structures. All the details about the heterogens and their 3 letter code in PDB can be found in the Het-PDB Navi database [148].

According to our analysis on this set of proteins, most of those heterogens have a demonstrated or putative role in the catalytic process (ZN, NAG, NAD, BME, MG, MN, U5P, ADP, HEM, FAD, MPD), while for others this role can be clearly ruled out (CL, NA, K, MAN), or it is just uncertain (PO4, SO4, POP).

In order to correctly encode discriminant features related to the presence of cofactors, we divided the heterogen molecules into groups (at least the 71 we found in the PDB dataset, excluding DNA molecules) based on their physico-chemical, functional, spatial or shape characteristics.

As an example, in this dataset of 79 enzymes ZN usually has a verified role within the active site, thus we considered it as a primarily catalytic cofactor. Actually among the whole set of known enzymes there are cases, such as the DNA glycosylase, having a zinc-finger in which ZN has a structural role. We believe other features of the residue 3D neighbourhood (e.g. four cysteine residues in the same sphere around a ZN atom) should help discriminating functional from non functional residues in these cases.

We analysed the distances of the heterogens from the catalytic residues, representing each heterogen by the centroid of the atoms composing it. We observed that the role mentioned in the literature is correctly reflected by the distribution of the distances from the catalytic residues. Figure 3.6 reports histograms of the distance of each one of the most frequent heterogen from catalytic residues. The first three rows contain heterogens having a role in the catalytic site: the peak of the frequencies is around values between 3 Å and about 15 Å depending on the space occupancy of the molecule. One exception is given by the N-acetyl-D-glucosamine (NAG) which is a monosaccharide that takes part in enzymatic processes like glycosilation: its average distance from the protein will make its presence in the residues neighbourhood quite a rare event. The fourth row of histograms relates to non-catalytic heterogens: the frequency peak is shifted around values greater than 15 Å, even for single ions such as CL, NA and K. Finally, the last row contains heterogens for which the distribution of distances does not allow to indicate a clear proximity or remoteness with respect to the catalytic site. In fact they appear as part of protein sites which are not annotated as catalytic.

By merging the above-mentioned literature-based information with our analysis of the distances from catalytic residues, we derived the final classification into three groups reported in Table 3.3.

| **Class** | **Heterogens** |
|---|---|
| **catalytic** | FE2, MN, CU1, MG, ZN3, ZN, HEM, HEG, HEC, SRM, MPD, MRD, FOK, PLP, P5P, PHS, OWQ, NO3, FS4, SF4, PVL, PYR, SEG, DHZ, FMT, HAD, CIT, ACN, PAC, ACT, 2PE, CNA, U5P, IKT, PGC, PGH, IMU, F6P, IMP, EEB, GLP, FBP, UD1, FCN, AZA, CRB, DHS, BME, ATP, ADP, GSH, FAD, FMN, SAM, AMP, NAD, GDP, GTP, GMP, MHF, NDP, NAG, NRI |
| **non-catalytic** | K, NA, NI, FE, CA, CL, SAC, FCY, PCA, MES, MAN |
| **uncertain** | PO4, PI, IPS, POP, SO4, SUL, GOL |

Table 3.3: Classification of the heterogens into three groups.

We encoded this information as a set of attributes ($3D_8$) describing the presence of heterogen molecules in the 3D neighbourhood of a residue. Following Table 3.3, this set includes three features counting the number of potentially catalytic, non-catalytic and uncertain heterogen molecules respectively.

According to the catalytic residue definition given in [7], which guides the annotation of the residues as functional in the CSA database, residues which bind a substrate or a cofactor are not annotated as catalytic unless they are in some way directly involved in the catalytic process. This consideration can be particularly useful to discriminate among residues with a high catalytic propensity (e.g. CYS, HIS) that bind cofactors for structural reasons. We represented this information as an additional feature ($3D_9$) encoding the presence of a bond between the target residue and a cofactor. We used a distance threshold of 3 Å for detecting bonds.

## 3.3 Experimental Results

Functional residue prediction can be cast into a binary classification task at the residue level, namely predicting for each residue of a given protein, whether it is directly involved in the catalysis or not.

### 3.3.1 Datasets

We performed a detailed analysis and feature engineering on a dataset (PW) of 79 enzymes selected by Petrova and Wu [111] for their structural and functional heterogeneity with respect to their SCOP fold classification, EC numbers and BLAST sequence similarity. The dataset contains enzymes from all the six classes in the EC Nomenclature. We collected sequential and three-dimensional data for a total of 23,635 residues from the enzymes PDB files. Few residues were removed with respect to the 23,664 extracted in [111] due to uncertain correspondence in the mapping between the two datasets or due to conflicts between the residues reported in the PDB structure file and in the FASTA sequence from Uniprot [131]. Only 254 out of 23,635 residues are labeled as functional in the CSA. Hence the dataset is strongly unbalanced with a ratio between positive and negative examples of about 1:92.

We also conducted a broad experimental evaluation of the obtained features on a set of larger benchmark datasets which were proposed by previous sequence and structured-based approaches. Three benchmark datasets with varying homology level were proposed in [149]: a SCOP fold dataset (EF fold), a SCOP family dataset (EF family) and a SCOP superfamily dataset (EF superfamily). Two additional datasets were included to study the performance of our approach in the presence of low homology: the HA SCOP superfamily dataset from [37] and the independent test set T-124 proposed in [151]. The characteristics of these five datasets are summarised in [151]. Finally we included the dataset of 160 proteins (POOL-160) used in [134] in order to compare with their approach.

#### Dataset Normalisation

Attribute values were normalised in the [-1,+1] range applying the following linear transformation: $value' = 2 \cdot \frac{value-min}{max-min} - 1$. While this implies a lower data sparsity with respect to a [0,1] normalisation, preliminary experiments showed that it achieved better overall results. Missing values were managed by replacing categorical attributes with their modes and numerical attributes with their means, both computed from the distributions of observed values in the dataset.

### 3.3.2 Experimental Setting

All experiments were carried out using the $SVM^{light}$ [74] software (downloadable from http://svmlight.joachims.org/). Our experimental evaluation is based on a 10-fold cross-validation procedure stratified at the protein level, that is, assuring that all residues of a certain protein always appear together in the same fold.

We fixed the regularisation parameter (parameter $c$ in the $SVM^{light}$ implementation) to 1, and tuned the cost factor (parameter $j$ in the $SVM^{light}$ implementation), which outweighs the error on positive examples with respect to that on negative ones, on each fold of the 10-fold cross-validation by an inner cross-validation procedure inside its training set. Tuning the cost factor is particularly important for this application due to the strong imbalance between the number of positive and negative examples. Previous works [128, 151] addressed such a problem by subsampling negative examples according to a certain ratio and training the classifier on the reduced set.

Our aim was to exploit information related to the properties of the local structure surrounding a residue. We added these features to those already used in [111], which aim at modelling properties of the target residue plus its relationship with the whole region containing it. Such combined representation allowed us to obtain significant improvements, as detailed in the following.

Table 3.4 reports a legend of the abbreviations we employed for the different sets of attributes that we tried. These sets of features include also the set of 24 attributes proposed in [111].

| **Abbreviation** | **Description** |
|---|---|
| $SVM\_P5_{1D}$ | the attributes extracted from the protein sequence among the 24 in [111] |
| $SVM\_P24$ | the whole set of 24 attributes proposed in [111] |
| $SVM\_P7$ | the optimal set of 7 attributes selected among the 24 in [111] |
| $SVM\_1D_{i-j}, 3D_{k-r}$ | the attributes from $1D_i$ to $1D_j$ and/or from $3D_k$ to $3D_r$ as described in section *Methods*, with $i, j = 1, 2, 3$ and $k, r = 1, ..., 9$ |

Table 3.4: Legend of abbreviations for the different sets of attributes tried in the experiments.

We evaluated the statistical significance of the performance differences between the various settings by paired Wilcoxon tests on the $F_1$ measure reported for each fold. We employed a confidence level $\alpha$ of 0.05.

### 3.3.3 Results of Different Feature Sets

We conducted a set of experiments aimed at elucidating the role of the different feature sets on the PW dataset. Preliminary experiments showed that polynomial (second and third degree) or Gaussian kernels did not significantly improve performance with respect to simpler linear kernels. All reported results thus refer to the latter type of kernel. Table 3.5 reports a summary of experimental comparisons for different sets of sequence- and structure-based features we used.

The first set of experiments refers to a sequence-based functional residue predictor, where each residue is characterised by features extracted from the protein sequence only (see Table 3.1). In Table 3.5, row 1 reports experimental results obtained by using our sequence-based attributes only, including the multiple alignment conservation profiles. We also experimented windows of conservation profiles of size varying between 1 and 10, where size $w$ implies a window of $w$ residues on each side of the target residue along the primary sequence, in addition to the profile of the target residue itself. Including such windows only provides a slight improvement (with $w = 7$) while drastically reducing the classifier efficiency. Furthermore, the features proved harmful when combined with structural information, possibly because the large number of features they introduced covered the signal coming from other more informative ones [161].

Rows 2 and 3 report additional results on sets of attributes extracted from sequence information only. The set $SVM\_P5_{1D}$ is a group of five attributes from [111] which includes the $1D_1$ and $1D_2$ attributes (see Table 3.1) and a conservation score from the Scorecons server [138], plus its entropy and relative entropy values, in place of our conservation profile. The results are comparable with those obtained with conservation profiles. Results combining all the available features extracted from the protein sequence are reported in row 3 ($SVM\_P5_{1D}\_1D_{1-3}$).

Results in the rows from the fourth on include additional information provided by structural features. In rows 4 and 5 we employed the two sets of attributes proposed in [111], i.e. the subset of the 7 optimal ones ($SVM\_P7$) and the entire set of 24 attributes ($SVM\_P24$) respectively. Note that we obtained performance improvements over the original results in [111] (achieving $F_1$=13% and MCC=23% for the $P24$ feature set) by

|  | **Performance % $\pm s.d.$** |  |  |  |  |  |  |  |
|---|---|---|---|---|---|---|---|---|
|  | **CV Exp** | P | R | FPR | $F_1$ | $MCC$ | $AUCROC$ | $AUCRP$ |
| 1. | $SVM\_1D_{1-3}$ | 22 ± 11 | 30 ± 11 | 1.3 ± 0.7 | 24 ± 7 | 24 ± 8 | 0.9172 | 0.2777 |
| 2. | $SVM\_P5_{1D}$ | 26 ± 8 | 29 ± 12 | 0.9 ± 0.3 | 27 ± 9 | 26 ± 9 | 0.9311 | 0.3129 |
| 3. | $SVM\_P5_{1D}\_1D_{1-3}$ | 27 ± 10 | 30 ± 10 | 1.0 ± 0.4 | 27 ± 8 | 27 ± 8 | 0.9370 | 0.3204 |
| 4. | $SVM\_P7$ | 22 ± 11 | 37 ± 11 | 1.8 ± 1.3 | 26 ± 10 | 27 ± 10 | 0.9490 | 0.3532 |
| 5. | $SVM\_P24$ | 26 ± 10 | 37 ± 14 | 1.2 ± 0.5 | 30 ± 9 | 30 ± 10 | 0.9529 | 0.3605 |
| 6. | $SVM\_P24\_1D_{1-3}$ | 26 ± 6 | 44 ± 10 | 1.4 ± 0.3 | 32 ± 7 | 33 ± 7 | 0.9556 | 0.3659 |
| 7. | $SVM\_P24\_1D_{1-3}, 3D_{1-6}$ | 28 ± 9 | 46 ± 10 | 1.4 ± 0.5 | 34 ± 8 | 34 ± 8 | 0.9635 | 0.3723 |
| 8. | $SVM\_P24\_1D_{1-3}, 3D_{1-9}$ | 33 ± 14 | 48 ± 8 | 1.4 ± 0.7 | 37 ± 7 | 38 ± 6 | 0.9633 | 0.4125 |

Table 3.5: Summary of the results of the cross-validation on different selected attributes (linear kernel, regularisation parameter $c = 1$).

tuning the cost factor for false positives versus false negatives, as compared to random sub-sampling negative examples in order to obtain a balanced set.

Table 3.6 reports F$_1$ measures of our best combination of sequence-based features, the sequence and structure based features from [111], plus our additional set of structural neighbourhood features, excluding those coming from ligand information.

| | HA superfamily | EF fold | EF superfamily | EF family | PW | POOL-160 |
|---|---|---|---|---|---|---|
| $SVM\_P5\_1D_{1-3}$ | 23 | 24 | 25 | 24 | 27 | 24 |
| $SVM\_P24$ | 21 | 24 | 24 | 23 | 30 | 23 |
| $SVM\_P24\_1D_{1-3}, 3D_{1-6}$ | 26 ○ ● | 28 ○ ● | 27 ○ ● | 28 ○ ● | 34 ○ | 27 ○ ● |

Table 3.6: Statistical comparisons of our best set of sequence-based features ($SVM\_P5\_1D_{1-3}$), the set of sequence- and structure-based features employed in Petrova and Wu [111] ($SVM\_P24$), and their combination with our additional set of structural neighbourhood features ($SVM\_P24\_1D_{1-3}, 3D_{1-6}$), excluding those coming from ligand information. Cross-validated $F_1$ measures (%) and results of a paired Wilcoxon test ($\alpha = 0.05$) on the statistical significance of the performance differences are reported for all benchmark datasets employed in this study. A white circle indicates a statistically significant improvement of the classifier in the row over the sequence-based classifier ($SVM\_P5\_1D_{1-3}$), while a black bullet indicates a statistical significant improvement over the Petrova and Wu features ($SVM\_P24$).

Results are reported for all test datasets described in section 3.3.1. The first relevant finding is that appropriate sequence-based features taking into account evolutionary information ($SVM\_P5_{1D}\_1D_{1-3}$) achieve performance which are comparable to carefully crafted structure-based ones [111] ($SVM\_P24$). The difference is never statistically significant in all tested datasets. This confirms the finding of [151] that state-of-the-art sequence-based predictors have performance comparable with recent structured-based approaches. Selecting the appropriate and discriminant structural attributes for functional residue prediction is thus not a trivial task.

On the one hand, using features extracted from primary sequence alone allows us to apply the predictor to the much larger set of sequentially but not necessarily structurally determined proteins. On the other hand, as we already discussed in the introduction and also stated in the review of [66], the availability of structural information should be able to significantly contribute in solving the task. Indeed, adding three-dimensional information in the form of properties of the residue structural neighbourhood allowed us to achieve significant improvements, as detailed below.

Row 6 in Table 3.5 reports results of the combination of our conservation profiles ($1D_3$) with all the sequence and structural attributes in [111]. Row 7 reports the result obtained by adding structural attributes encoding statistics of the residue three-dimensional neighbourhood properties ($3D_{1-6}$) without including the attributes related to the ligands ($3D_{7-9}$). Such results are always significantly better than those of sequence-based classifiers according to the statistical tests (see Table 3.6). Furthermore, performance improvements with respect to previous structure-based results ($SVM\_P24$) are significant in all but the smallest test set.

Finally, row 8 reports the performance obtained by including all the available ligand-based features, which allow to achieve further improvements and correctly predict some especially tough cases (detailed below), paving the way to an interesting research direction. Additional files 3 and 4 report detailed results and predictions for this classifier.

In order to better understand which are the features contributing the most to the classifier performance, we analysed the weight vector $\vec{w}$ describing the separating hyperplane learnt by the SVM. The $\vec{w}$ components with higher absolute value are associated to the most discriminant features of the classifier. In Figure 3.7 we represent the weight vector of a classifier trained on the PW dataset. Among the most relevant properties there are features related to the relative position on the protein surface (*cleft_rank*, *cleft_Vol_SA*, *cleft_Area_SA*, *nearest_cleft_distance*), features related to the conservation along the primary sequence (*conservation_score*, *entropy* and *relative_entropy* plus some other features from the conservation profiles $1D_3$) and also features describing the structural residue neighbourhood. For instance, the fractions of amino acids with low and medium hydrophobicity are quite discriminative in opposite directions. The same holds for low and medium Van der Waals volume. Other discriminative features include the atomic density of the residue sphere and the features related to its amino acidic composition: in particular the number of ASN, CYS and GLN residues, but also that of MET, PHE and TRP ones.

Further analyses on the effect of different sets of features on prediction errors provide some interesting insights on their usefulness and reliability. The quality of multiple alignments strongly influences the performance of sequence-based classifiers. On the proteins for which PSI Blast did not provide good alignments we observed poor performance. In those cases structural features help in compensating such deficiencies. The inclusion of ligand features allows the correct prediction of many catalytic residues which have low catalytic propensity, like the glycine in the methylglyoxal synthase (PDB code 1B93) and the glycine in the human glutathione synthetase (PDB code 2HGS). The latter is one of the emblematic cases of the importance of ligand features, as in the absence of those features only one of its four catalytic residues is correctly predicted. In the phosphofructokinase (PDB code 1PFK) the encoding of ligand features helps to correctly predict the two arginine residues of the active site. By looking at the three-dimensional structure of the protein, the active site seems to be exposed rather than located in a hydrophobic core. This implies that active site residues have associated structural features which may differ from those typical of the other catalytic residues in the dataset. On the other hand, we also observed few cases in which the addition of ligand features worsens predictions. This happens mainly when no heterogen appears in the crystal structure, possibly because the enzyme was solved in its apo form. We are planning to verify such conjecture by applying techniques for detecting binding sites in 3D structures [3].

Note that while the presence of a heterogen provides a clear hint that the area could contain an active site, it is not by itself sufficient to determine the set of catalytic residues. Out of the 365 heterogen-binding residues in the dataset, only 62 were actually labelled as catalytic. If we restrict to the subset of heterogens which tend to occur near catalytic sites in enzymes, the fraction becomes 51 out of 285. As detailed in [7], the sole fact of binding a substrate or cofactor does not classify a residue as catalytic. It also has to perform some specific activity such as proton abstraction from substrate, cofactor or water activation. For instance, the above-mentioned phosphofructokinase (PDB code 1PFK) contains three heterogens: ADP, beta-fructose diphosphate (FBP), and a magnesium ion; of the 15 residues which bind one of them, only four are actually catalytic. In this case the predictor manages to selectively exploit ligand information in identifying two active arginine residues, one of which does not directly bind any heterogen, with a single additional FBP-bound arginine incorrectly predicted as catalytic. Given that information on binding residues helps detecting active ones, it would be interesting to predict it when missing, either because sequence information alone is available, or because the 3D-structure does not contain the bound cofactor and/or substrate. Indeed, both binding and active residues should be identified in order to fully characterise the functional domain. We believe that combining active and binding site prediction in a single collective model, as already done with profile-HMM for specific functional domains [15], is a promising research direction, which can rely on a number of works for predicting binding sites from both sequence [93, 126] and structural information [3, 54].

## 3.3.4 Comparison with Other Methods

We conducted a broad range of experiments on multiple benchmark datasets (see section 3.3.1), and compared the results with the most recent methods for both sequence-based and structure-based prediction. Considered that none of the other methods directly encodes information on heterogens, we excluded such features from our set in all these comparisons.

| Method | | Datasets of competing methods | | | | | |
|---|---|---|---|---|---|---|---|
| | | HA superfamily [37] | EF fold [149] | EF superfamily [149] | EF family [149] | PW [111] | T-124 [151] |
| CRpred | R | 54.0 | 48.2 | 52.1 | 58.3 | 53.7 | 50.1 |
| | P | 14.9 | 17.0 | 17.0 | 18.6 | 17.5 | 14.7 |
| | | $SVM\_P24\_1D_{1-3}, 3D_{1-6}$ | | | | | |
| Equal P | R | 67.4 | 64.6 | 66.2 | 61.3 | 69.7 | 54.8 |
| Equal R | P | 21.0 | 24.1 | 23.9 | 20.5 | 22.5 | 15.5 |

Table 3.7: Comparison with the CRpred [151] sequence-based approach on six benchmark datasets. For each dataset we report recall obtained by our predictor at a precision equal to that of the competing method and precision at equal recall. Results are obtained without including ligand information.

Table 3.7 shows experimental comparisons with the state-of-the-art sequence based predictor CRpred [151] on a number of datasets. Adhering to the setting in [151], we employed a 10-fold cross-validation procedure for all datasets but the T-124 one, for which we trained a single predictor on the entire EF fold dataset and tested it on the T-124 one. Our structural neighbourhood features allow to consistently improve performance on all datasets, as measured by recall at equal precision, and precision at equal recall. The ROC and RP curves for the two low homology datasets HA

| Method | | Datasets of competing methods | | | |
|---|---|---|---|---|---|
| | | HA superfamily [37] | EF fold [149] | EF superfamily [149] | EF family [149] |
| Competing | R | 29.3 | 51.1 | 53.9 | 57.0 |
| methods | P | 16.5 | 17.1 | 16.9 | 18.5 |
| | | $SVM\_P24\_1D_{1-3}, 3D_{1-6}$ | | | |
| Equal P | R | 63.4 | 64.2 | 67.3 | 61.7 |
| Equal R | P | 30.9 | 22.1 | 22.5 | 20.9 |

Table 3.8: Comparison with the structure-based approaches by *Chea et al.* [37] and *Youn et al.* [149] on their benchmark datasets. For each dataset we report the recall obtained by our predictor at a precision equal to that of the competing method and the precision at equal recall. Results are obtained without including ligand information.

superfamily and EF fold are shown in Figure 3.8, while those for the other datasets are reported in Figures 3.9, 3.10 and 3.11.

Previous results [151] suggested that appropriate sequence-based features managed to match performance of different structure-based predictors on the same datasets, a result we also observed in our early experiments on the PW dataset. Conversely, the improvements we achieve here show that structural information can indeed be effectively employed in predictions. Nonetheless, further research is needed in order to fully exploit it, as our results using heterogen information seem to indicate.

Table 3.8 reports comparisons with the structure-based predictors from Chea et al. [37] and Youn et al. [149] for each of the benchmark datasets. Results, again measuring recall at equal precision and precision at equal recall, clearly indicate that our structural features consistently improve over the different methods on all datasets.

Table 3.9 reports experimental comparisons with an additional structure-based predictor recently developed by Tang et al. [128] and tested on the PW dataset: the GANN method employs a neural network trained using a genetic algorithm. It includes a highly discriminant feature measuring network centrality, which accounts for the tendency of catalytic residues to have multiple interactions with other residues. The $F_1$ and MCC measures of the two methods do not allow to draw clear conclusions, with our method achieving better $F_1$ and worse MCC with respect to GANN. However, the availability of the detailed predictions of the cross-validation allows us to evaluate the overall threshold independent performance by areas under the ROC and RP curves. Both of them clearly show the advantages of our structural features.

**Performance %**

| Method | P | R | FPR | $F_1$ | *MCC* | *AUCROC* | *AUCRP* |
|---|---|---|---|---|---|---|---|
| *Tang et al. (GANN)*[1] | $19^2$ | 73 | 3.8 | $31^2$ | 36 | 0.9313 | 0.3556 |
| $SVM\_P24\_1D_{1-3}, 3D_{1-6}$ | 28 | 46 | 1.4 | 34 | 34 | 0.9635 | 0.3723 |

[a]subsampling of negative examples with a ratio of 1:6 w.r.t. positives

[b]directly computed

Table 3.9: Comparison with the structure-based approach by *Tang et al.* [128] on the PW dataset. Results include both performance measures at fixed decision threshold and average areas under ROC and RP curves. Results are obtained without including ligand information.

Finally, we compared with the recent structure-based predictor POOL [134] (Partial Order Optimum Likelihood), which combines effective electrostatic features from THEMATICS [105] with geometric and sequence conservation features in a maximum likelihood approach. Averaged ROC curves are reported in Figure 3.12. We compared our method and the POOL predictor with different sets of features as taken from [134]. The point representing Petrova and Wu results was also included in the graph. Our method achieves superior recall for all possible values of the false positive rate. We also conducted experiments on the dataset of 160 proteins proposed by the POOL authors [134]. In Table 3.10 we compare our results with the results of the best classifier $(POOL(T) \times POOL(G) \times POOL(C))$

reported in [134] at equal recall and at equal precision. Averaged Recall (AvgR) and Precision (AvgP) are computed as in [134] as averages at the protein level. The area under our averaged ROC curve is 0.9523 as compared to 0.925 achieved by the best set of features for POOL.

| | | **Performance %** | | |
|---|---|---|---|---|
| | **Method/Dataset** | AvgP | AvgR | *AUCROC* |
| 1. | POOL(T)POOL(G)POOL(C)/allprotein [134] | 19.07 | 64.68 | 0.925 |
| 2. | $SVM\_P24\_1D_{1-3}, 3D_{1-6}$ at Equal Precision | 19.07 | 78.10 | 0.948 |
| 3. | $SVM\_P24\_1D_{1-3}, 3D_{1-6}$ at Equal Recall | 26.61 | 64.68 | 0.948 |

Table 3.10: Comparison with the best results reported for the POOL structured-based method (*Tong et al.* [134]) on their benchmark dataset of 160 proteins. Performance measures include: the average per-protein precision at equal recall, the average recall at equal precision, and the average area under the ROC curve (AUCROC). Results are obtained without including ligand information.

## 3.4 Engineering Active Site Features: Kernels on Structures

In order to further investigate the discriminative potential of the features extracted from the 3D residue neighbourhood we also experimented a structured kernel.

### 3.4.1 3D Kernel on Shapes

Geometric shapes extracted according to spatial considerations can be viewed as features characterising a residue structural neighbourhood. Planar shapes, for instance, can be viewed as substructures of the 3D space surrounding a residue and characterising its interactions with the other residues.

We employed a 3D decomposition kernel on planar shapes in the 3D space. This kernel was proposed in [35] for the classification of small molecules.The 3D kernel is defined as a convolution kernel on three-dimensional shapes of order $n \geq 2$, for example, a line is a shape of order 2, a triangle is a shape of order 3. Hence, a shape is defined by a sequence of $\frac{n\cdot(n-1)}{2}$ edges. The kernel on a pair of shapes is then defined as the product over all edges in the shapes of the product of a Dirac kernel on the edge labels of each shape and a Gaussian kernel over the atom distances. We adapted it to the functional residue prediction task by extracting specific shapes from the residue structural neighbourhood.

Among the different design choices we tried, the best performing one was the set of planar shapes of two (segment) and three (triangle) vertices in the 3D neighbourhood of a residue. One of the vertices was the target residue itself, and the others were residues evolutionary conserved over one of the hydrophobic, charged or polar classes. Labelling these residues with their class types — charged (Ch), hydrophobic (Hy) or polar (Po) —, allowed us to reduce the sparseness of the whole set of shapes, thus increasing the likeliness of shape matches during the kernel evaluations. We represented the three-dimensional neighbourhood of a residue as: (a) a cloud of points corresponding to the side-chain centroids of the residues, labeled with *Ch*, *Hy* or *Po* according to their class; (b) a graph where each pair of vertices in the cloud is connected by an edge if their distance is less than 5 Å.

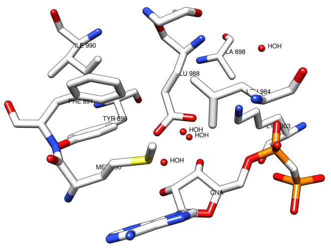
From these two representations we extracted different sets of shapes to be used along with the 3D decomposition kernel: (1) shapes only composed of residues with conserved class, (2) shapes containing the target residue, and (3) shapes containing connected residues only (i.e. pairwise distances less than 5 Å). We consider the class of a residue conserved when the sum of the profile entries corresponding to amino acids belonging to it is greater than 0.5. In Figure 3.13 two triangular shapes centred on the target residue HYS 303 are shown.

The 3D kernel measures the similarity between two residues in terms of the shapes which are shared between their respective 3D neighbourhoods.

While providing reasonable performance when used alone, with an average $F_1$ of 22% and an average MCC of 25%, such shapes failed to improve performance in combination with the remaining sequence- and structure-based features.

In a preliminary work tree kernels were also applied on structures extracted from the 3D neighbourhood to predict whether a spherical regions has the characteristics to potentially host an active site [157]. Good results were obtained by learning in conjunction with polynomial kernels on a reduced set of hydrolases but, when applied to several different enzyme families, as for the other approaches discussed in this chapter, this approach was not able to give results comparable with the state of the art.

These results confirm that effectively exploiting three-dimensional information for modelling catalytic residues is a hard task, and further research is needed.



| | Feature | Value |
|---|---|---|
| $3D_1$ | aa_with_LOW_hydrophobicity | 2/9 |
| | aa_with_MEDIUM_hydrophobicity | 3/9 |
| | aa_with_HIGH_hydrophobicity | 4/9 |
| | aa_with_LOW_Van_der_Waals_vol | 2/9 |
| | aa_with_MEDIUM_Van_der_Waals_vol | 3/9 |
| | aa_with_HIGH_Van_der_Waals_vol | 4/9 |
| | aa_with_LOW_polarity | 5/9 |
| | aa_with_MEDIUM_polarity | 2/9 |
| | aa_with_HIGH_polarity | 2/9 |
| | aa_with_LOW_polarizability | 2/9 |
| | aa_with_MEDIUM_polarizability | 3/9 |
| | aa_with_HIGH_polarizability | 4/9 |
| $3D_2$ | ALA | 1 |
| | ARG | 0 |
| | ASN | 0 |
| | ASP | 0 |
| | CYS | 0 |
| | GLN | 0 |
| | GLU | 1 |
| | GLY | 1 |
| | HIS | 0 |
| | ILE | 1 |
| | LEU | 1 |
| | LYS | 1 |
| | MET | 1 |
| | PHE | 1 |
| | PRO | 0 |
| | SER | 0 |
| | THR | 0 |
| | TRP | 0 |
| | TYR | 1 |
| | VAL | 0 |
| $3D_3$ | aa_with_positive_charge | 1 |
| | aa_with_negative_charge | 1 |
| | aa_with_positive/negative_charge | 2 |
| $3D_4$ | number_of_H2O_molecules | 4 |
| $3D_5$ | number_of_atoms | 58 |
| $3D_6$ | normalised_B_factor | 0.304 |
| $3D_7$ | disulf_bond | 0 |
| $3D_8$ | ccofact_dist | 1 |
| | scofact_dist | 0 |
| | dcofact_dist | 0 |
| $3D_9$ | cofactor_bond | 1 |

Figure 3.4: An example of feature vector extracted from the three-dimensional neighbourhood of the (catalytic) residue GLU 988 in the poly(adp-ribose) polymerase (1A26).

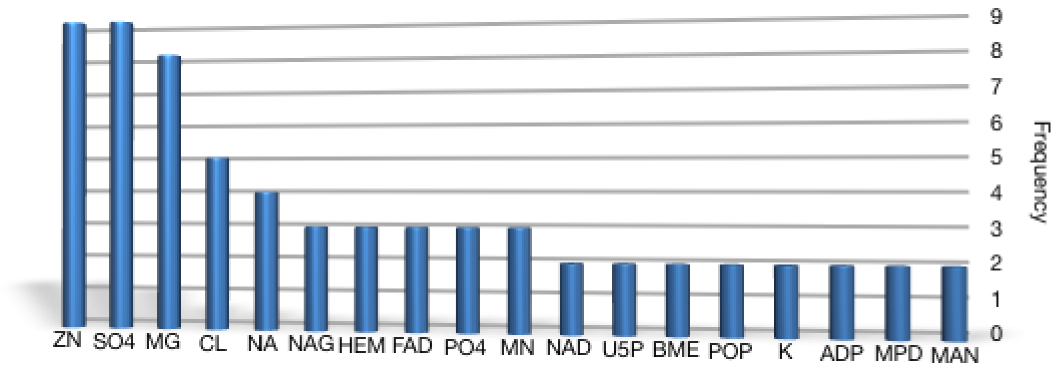


Figure 3.5: Histogram of the frequencies of heterogen molecules in the PW dataset (79 enzymes). Only the heterogens appearing in more than one protein structure are reported.

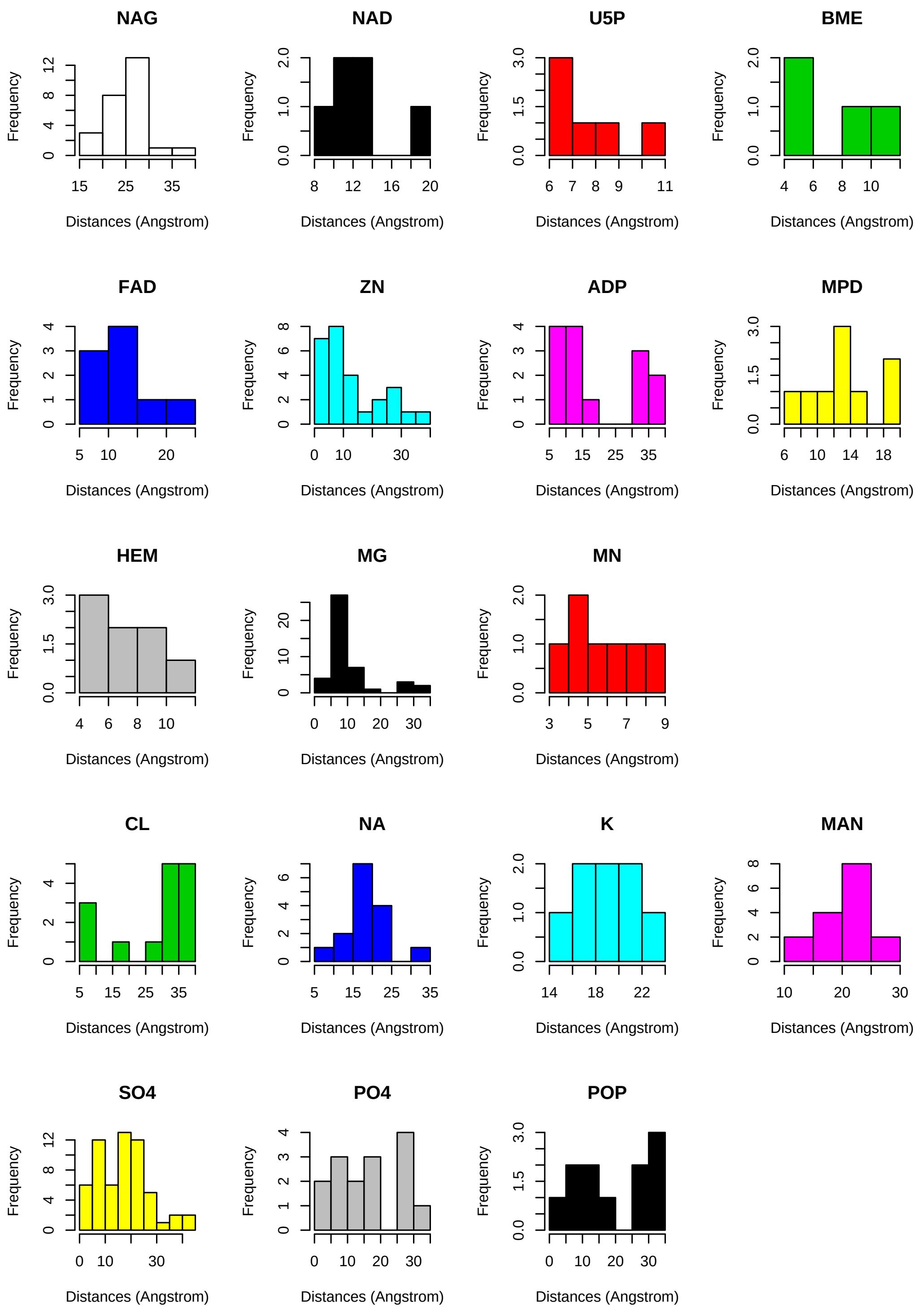


Figure 3.6: Histograms of the distances of the most frequent heterogens from catalytic residues in the PW dataset.

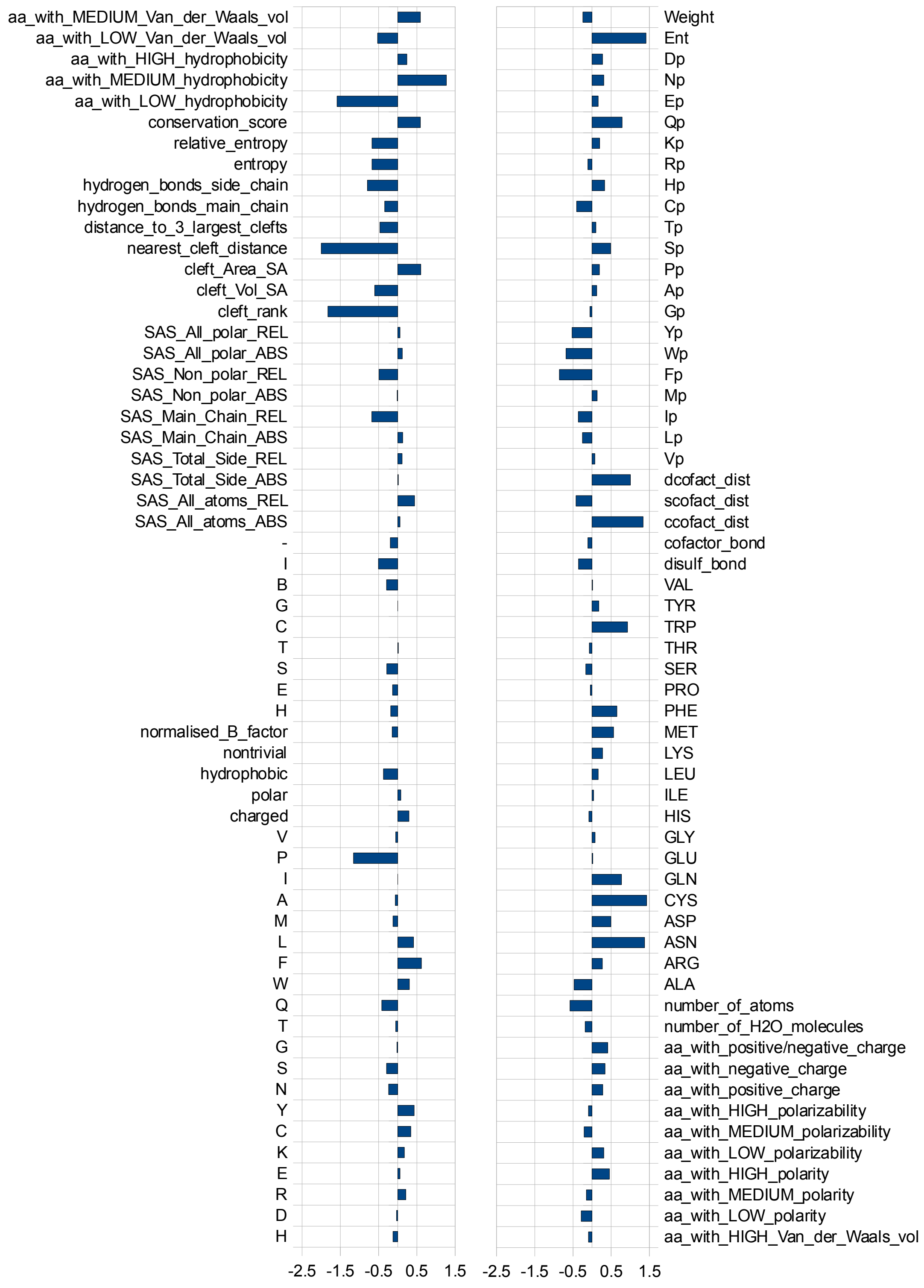
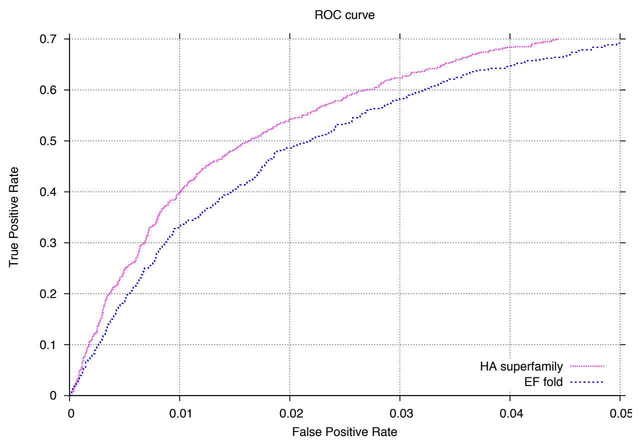
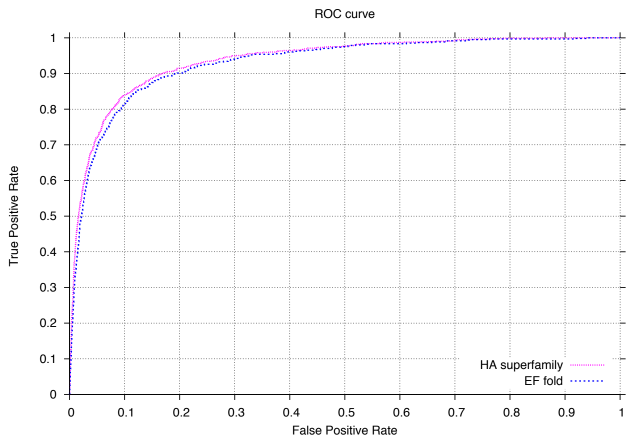


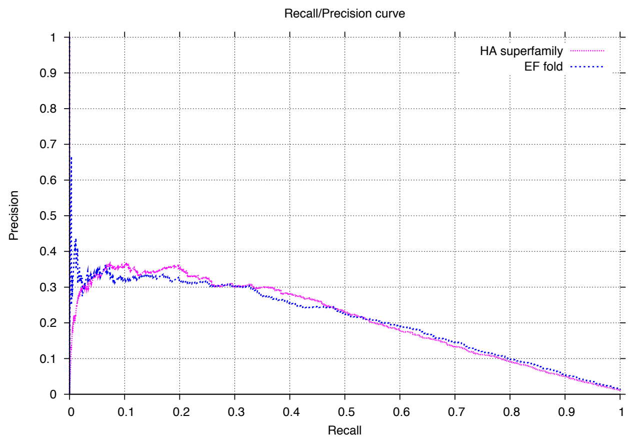
Figure 3.7: Vector of the feature weights $\vec{w}$ of a classifier trained on the PW dataset.



(a) Local ROC curve on the HA superfamily and EF fold dataset

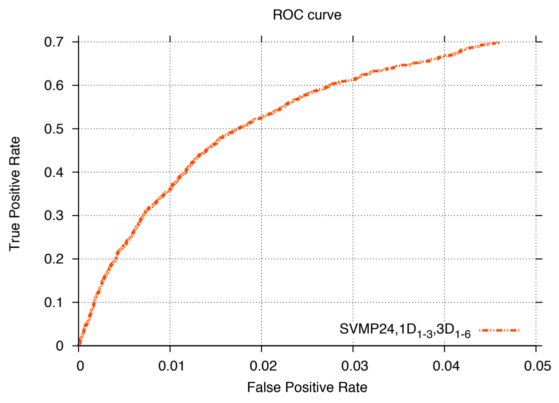


(b) Global ROC curve on the HA superfamily and EF fold dataset

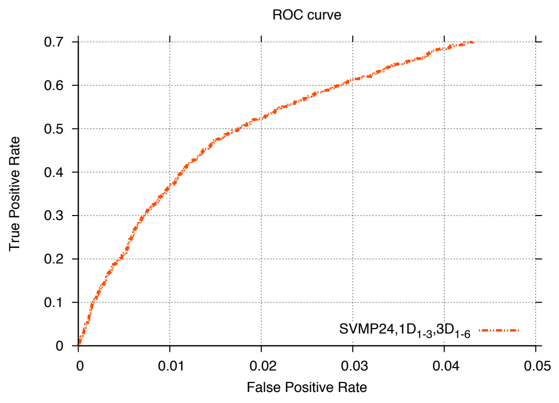


(c) Recall/Precision curve on the HA superfamily and EF fold dataset

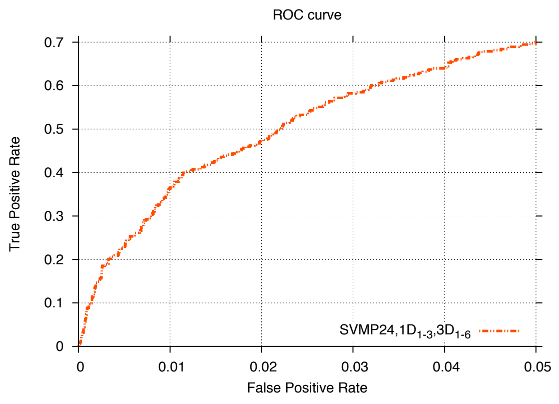
Figure 3.8: ROC and Recall/Precision curves of the predictions on two low homology benchmark datasets.



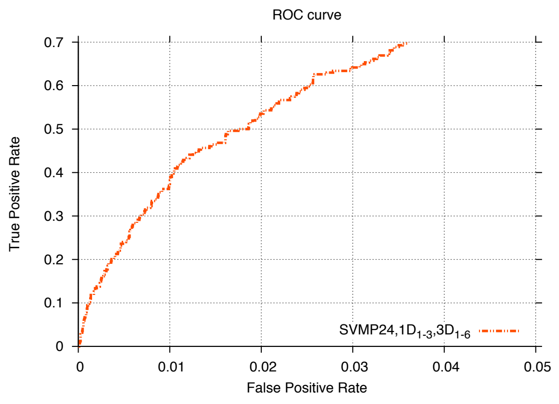
(a) Local ROC curve on the EF family dataset



(b) Local ROC curve on the EF superfamily dataset

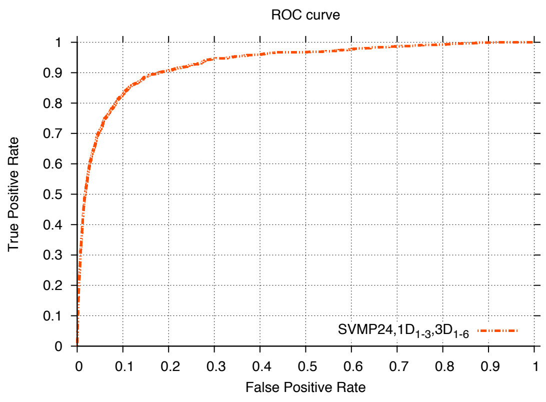


(c) Local ROC curve on the T-124 dataset

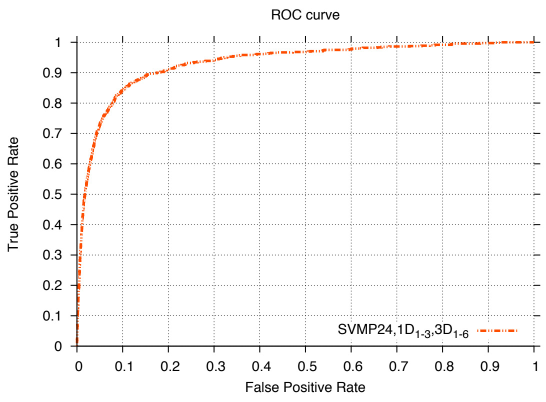


(d) Local ROC curve on the Petrova and Wu dataset

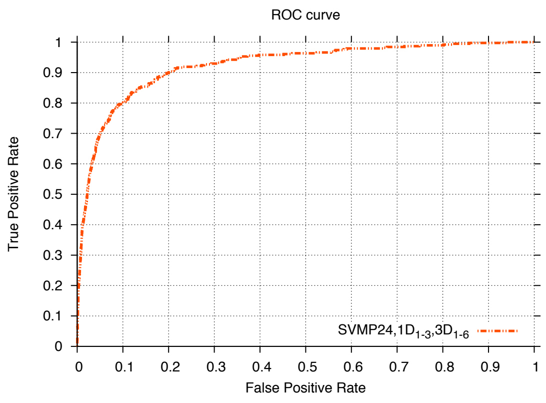
Figure 3.9: Local ROC curves of the predictions on different benchmark datasets.



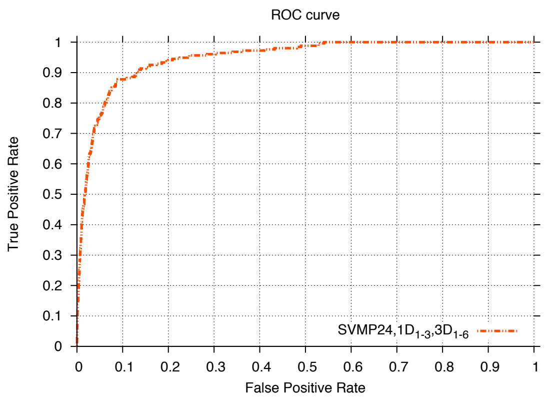
(a) Global ROC curve on the EF family dataset



(b) Global ROC curve on the EF superfamily dataset

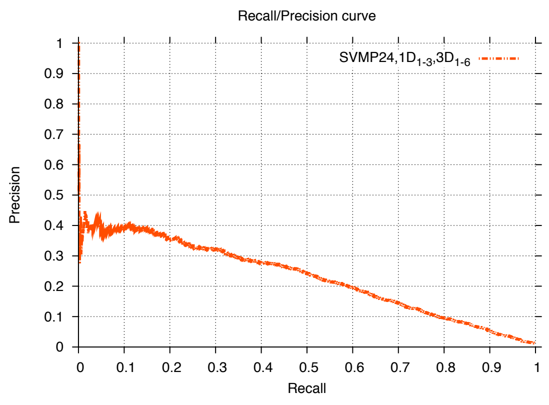


(c) Global ROC curve on the T-124 dataset

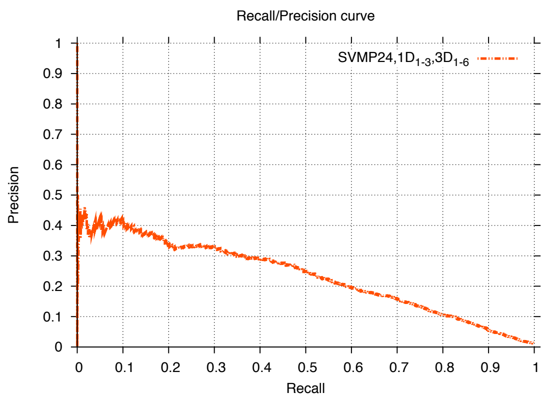


(d) Global ROC curve on the Petrova and Wu dataset

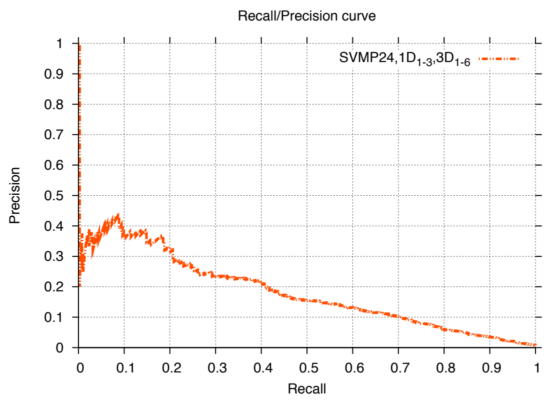
Figure 3.10: Global ROC curves of the predictions on different benchmark datasets.



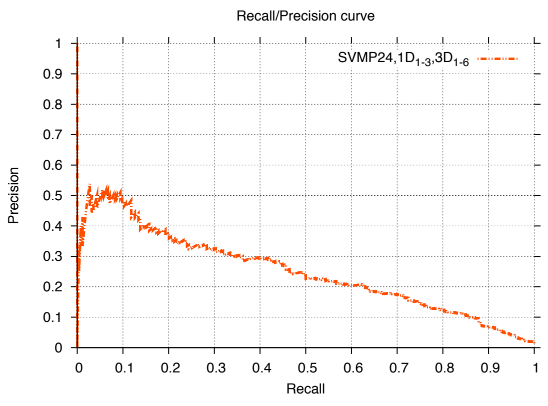
(a) Local ROC curve on the EF family dataset



(b) Local ROC curve on the EF superfamily dataset



(c) Local ROC curve on the T-124 dataset



(d) Local ROC curve on the Petrova and Wu dataset

Figure 3.11: Recall/Precision curves of the predictions on different benchmark datasets.

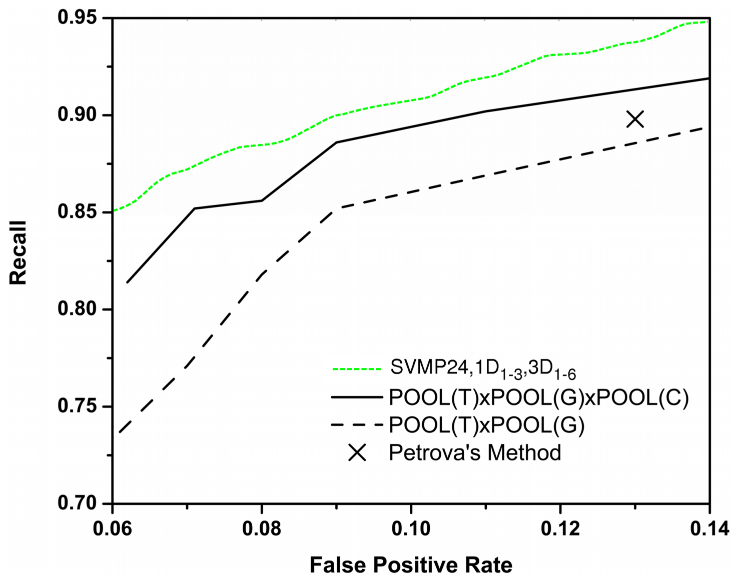


Figure 3.12: ROC curves superimposed with those reported in [134] on the PW dataset.

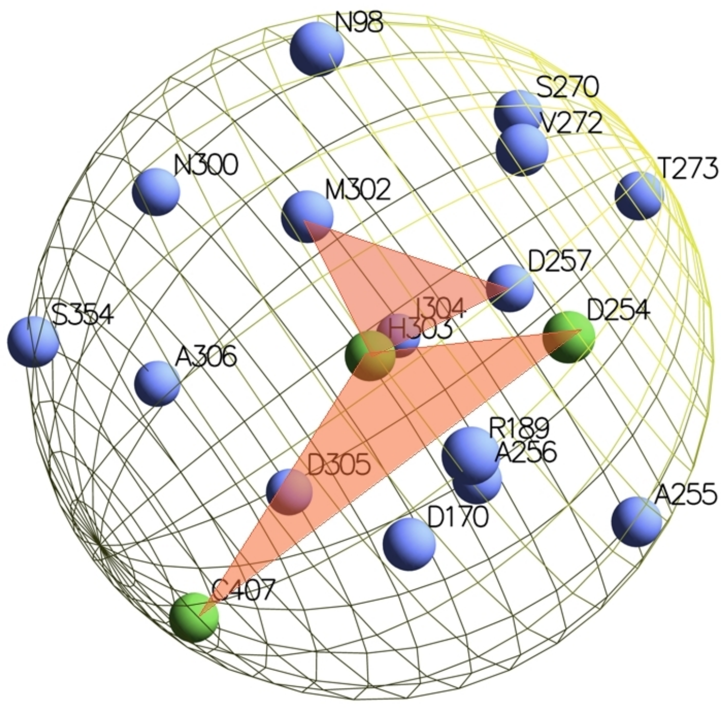


Figure 3.13: Two examples of triangular shapes extracted from the HYS303 three-dimensional neighbourhood.

# Chapter 4

# Toward a Collective Learning Approach

In the previous chapters the problem of active site prediction has been addressed as a binary classification task at the residue level: given a protein sequence, to predict for each residue whether it is involved in the active site.

This chapter addresses the problem of collectively predicting all the residues involved in functional sites [162]. The approach is also applied to the prediction of the geometry of metal binding sites. Indeed, the task is closely related to the identification of functional residues as already explained in Chapter 2. In the case of metal binding site prediction the problem is extended to determining the number of sites in the same protein and the residues involved in each of them.

The extremely challenging combinatorial problem here is formulated as a distance-based supervised clustering task [8], where training proteins are employed to learn a proper distance function between residues. The learnt pairwise distance is employed to turn instances into weighted graphs. A partial clustering is then returned by searching for maximum-weight cliques in the resulting protein graph representation. The partial clustering corresponds to a small set of densely connected components corresponding to candidate sites.

The approach is based on a stochastic local search algorithm for efficiently generating approximate solutions for the maximum-weight clique problem. The algorithm has a number of desirable features including automatic selection of the number of clusters — for example, the number of metal binding sites in apo-proteins is unknown —, natural handling of partial clusterings with many outliers and overlapping clusters, and scalability to large datasets.

The method achieves significant improvements over the current state-of-the-art in predicting catalytic sites from 3D structure in enzymes, where both node and edge weights are employed in order to exploit both local predictions and spatial constraints. Substantial improvements are also achieved in the related task of metal binding site prediction from sequence alone, over a previous structured-output approach proposed in [60].

## 4.1 Problem Description and Formalisation

Given a protein, the problem consists of detecting the number of catalytic or metal binding sites, collecting for each site the set of protein residues involved.

Metal binding sites are characterised by the set of protein atoms directly involved in binding the ion, called ligands, and the overall geometry of the site. Furthermore, the same protein often binds multiple ions, with typical numbers ranging from one to four. They tend to be rather specific in terms of possible ligands with cysteine (C), histidine (H), aspartic (D) and glutamic (E) acids being by far the most common ligands in transition metals. Cysteine and histidine residues are the vast majority of ligands in metal binding sites, while aspartic and glutamic acids are quite common in proteins and their relative binding frequency is thus very limited [107].

A more complex situation can be observed with alkali and alkaline-earth metals, which often bind proteins through the oxygen in backbone carbonyl groups.

Catalytic propensity is even less specific, given the number of different roles that a residue can play within the active site. Figure 4.1 reports the catalytic propensity of the whole set of amino acids showing that only few of them can be safely discarded. The previous results on the simpler binary classification task actually indicate that keeping all candidates produces slightly better results on average: the predictor occasionally manages to correctly predict rare amino acids as catalytic without significantly affecting precision.

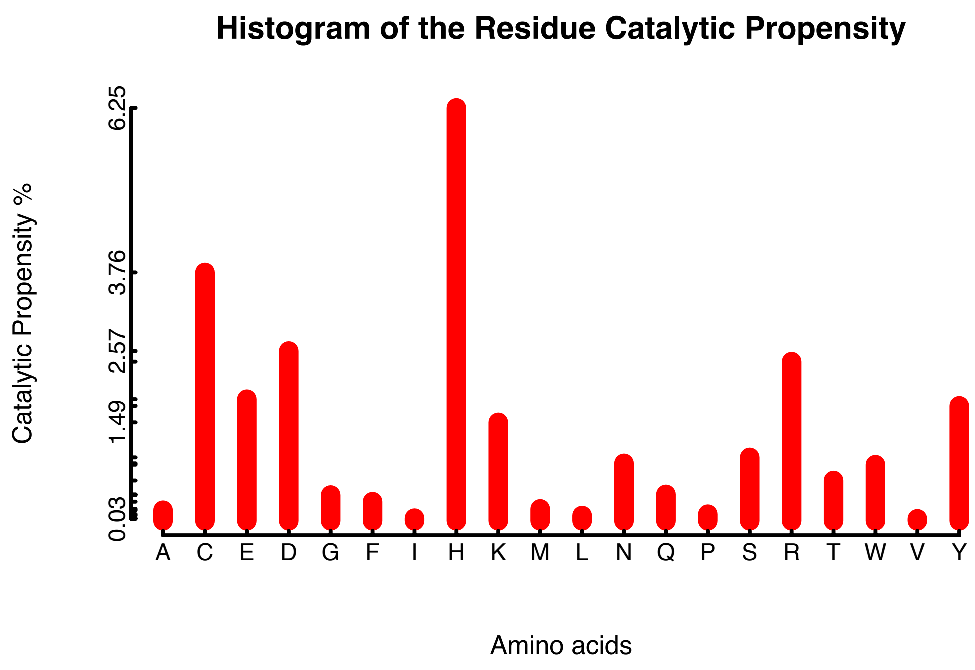


Figure 4.1: Histogram of the catalytic propensities of the residues in the experimental dataset *HA superfamily* (see experimental section for details).

Concerning the number of sites, metalloproteins usually contain between one and three sites, sometimes four and occasionally more. The coordination number of a bound ion, i.e. the total number of its ligands, varies from one to about eight depending on the metal. Values between two and four are the most frequent for transition metals. Figure 4.2 shows the metal binding geometry of the equine herpes virus-1 (PDB code 1CHC), where candidate ligands in $\mathcal{L} = \{C, H\}$ not binding any ion are marked in green. Contrarily to metal binding sites, enzymes tend to have a single catalytic site involving a larger number of residues, ranging from 1 to 9 in the experimental dataset we used. Multiple active sites can actually be found in some multimeric proteins, such as the 3-isopropylmalate dehydrogenase (PDB code 1A05). Figure 4.3 shows the active site of cloroperoxidase T (PDB code 1A7U and UniProtKB entry O31168) with seven residues corresponding to seven different amino acids involved. Note that proximity in sequence only partially relates to involvement in the same site, as the three-dimensional arrangement of the protein can bring quite distant residues closer as explained in Chapter 2. However, additional features contribute to characterise target residues, such as a conservation profile and the residue neighbourhood.

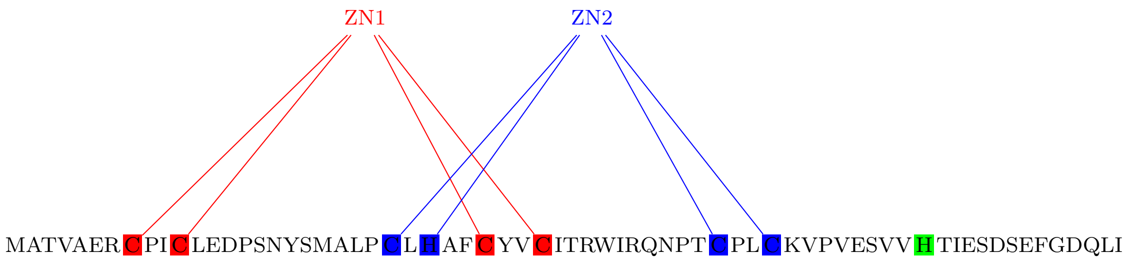

ZN1
ZN2
MATVAERCPICLEDPSNYSMALPCLHAFCYVCITRWIRQNPTCPLCKVPVESVVHTIESDSEFGDQLI

Figure 4.2: Sequence of the equine herpes virus-1 (PDB code 1CHC). Residues composing the metal binding sites are highlighted in different colours.

Given these premises, the problem is here formulated as a supervised clustering task. We provide a common formulation for both active site and metal binding site prediction. Slightly abusing terminology, we refer to residues involved in either type of site as *ligands*. While the two problems are treated as separate tasks in the experiments, they are indeed highly

```
MPFITVGQEN STSIDLYYED HGAGQPVVLI HGFPLSGHSW   40
ERQSAALLDA GYRVITYDRR GFGQSSQPTT GYDYDTFAAD   80
LNTVLETLDL QDAVLVGFSM GTGEVARYVS SYGTARIAKV  120
AFLASLEPFL LKTDDNPDGA APKEFFDGIV AAVKADRYAF  160
YTGFFNDFYN LDENLGTRIS EEAVRNSWNT AASGGFFAAA  200
AAPTTWYTDF RADIPRIDVP ALILHGTGDR TLPIENTARV  240
FHKALPSAEY VEVEGAPHGL LWTHAEEVNT ALLAFLAK
```

Figure 4.3: Sequence of the cloroperoxidase T (PDB code 1A7U and UniProtKB entry O31168). Residues composing the active site are highlighted in red.

correlated as metal binding sites can be part of a larger active site or can coordinate catalytic ions. In the latter case they are sometimes said to be "cocatalytic".

A protein sequence is represented as the set $x$ of its candidate ligands, that is residues belonging to $\mathcal{L}$. The output $y$ for the sequence is a subset of the powerset of $x$, i.e. $y \subseteq \mathcal{P}(x)$. In the case of active site prediction, if we limit the prediction to a single site the possible outputs are $|\mathcal{P}(x)| = 2^n$, where $n$ is equal to the number of residues in the protein $x$. For instance, the average sequence length in the UniProtKB/Swiss-Prot is 352 amino acids[1]. The problem is therefore quite challenging as the number of outputs is exponential in the size of the input. Outputs for proteins in Figures 4.2 and 4.3, for instance, would be represented as $\{\{c_1, c_2, c_4, c_5\}, \{c_3, h_1, c_6, c_7\}\}$ and $\{f_2, s_8, m_2, a_{14}, p_7, d_{18}, h_6\}$ respectively, assuming $\mathcal{L}$ is equal to $\{C, H\}$ for metal binding sites and $\mathcal{L}$ is the whole set of amino acids, for catalytic sites.

The desired output is thus a partial clustering of residues, where only predicted ligands are reported. Furthermore, at least for metal binding sites, clusters can overlap, as the same residue can simultaneously bind two ions, as happens for glutamic and aspartic acids with their two side-chain oxygen atoms. For comparison with previous approaches, experiments only

[1]source:http://www.expasy.org/sprot/relnotes/relstat.html

deal with non-overlapping clusters, but the approach can naturally handle overlaps, as described in the next section.

## 4.2 Distance-based Supervised Clustering with Maximum-weight Cliques

We opt for a distance-based supervised approach [8], where training instances are used to learn an appropriate distance (or similarity) measure to be later used in the clustering. The learning stage simply consists of training a pairwise classification function $F(x^i, x^j)$ predicting for each pair of residues $x^i$ and $x^j$ in $x$ whether they belong to the same site. Again, an SVM is employed as the underlying classification function.

Given a learnt similarity function $F$, we represent a set $x$ as a weighted graph, removing edges whose weight is below a certain threshold $\theta$ and rescaling remaining weights to be positive. A maximum-weight clique algorithm is then run on the graph in order to return a set of maximal cliques, i.e. cliques that are not contained in larger cliques, which correspond to the predicted sites. The rationale for the approach is that given a reasonable pairwise similarity measure, the algorithm should isolate few densely connected components which correspond to the desired solution while discarding most of the nodes in the graph. The algorithm can be asked to return a single large cluster, as typical of the active site prediction task, or a set of possibly overlapping maximal cliques, as for the metal binding site case, where the number of clusters cannot be specified *a priori*.

The described approach actually learns a separating hyperplane among paired feature vectors representing the protein residues. The distance of the vector $k$ of paired residues from the learnt hyperplane, i.e. the margin $\gamma^k$, is taken as a measure of likelihood that the pair belongs to the same site. This approach is applied when searching active and metal binding sites by only knowing the protein primary sequence. When the protein structural information is available, other considerations, i.e. spatial consideration, can be made for clustering functional or metal binding residues belonging to the same site. Therefore, in this case, we took a slightly different approach. Edges of the protein graph are weighted with the inverse of the Euclidean distance between the residues side-chain centroids (see Chapter 3). Furthermore, weights on the vertices are added, which correspond to the margins $\gamma^k$ of the structure-based functional residue classifier described in Chapter 3. The maximum-weight clique algorithm takes into consideration both edge and vertex weights during the search.

Note that the maximum-weight clique algorithm is independent of the supervised learning stage, and can be easily integrated in more complex supervised clustering approaches such as the structured-output formulation proposed in [57].

## 4.3 The Maximum-weight Clique Clustering Algorithm

Maximum Clique (MC) is a paradigmatic combinatorial optimisation problem with relevant applications in many areas; its weighted versions originate from fields such as computer vision, pattern recognition and robotics [6]. A survey on recent literature on Weighted Maximum Clique algorithms can be found in [114].

A heuristic algorithm is introduced for searching maximum-weight cliques in protein weighted graph, as discussed in the previous section. The algorithm is described for weighted edges only. Its extension for dealing with weights on both nodes and edges, as well as the case where weights are averaged on the number of nodes, is straightforward.

In the previous section we defined a learnt symmetric similarity function $F$ that maps each pair of residues onto a measure of likelihood that they belong to the same cluster. Given the set of protein residues $R$, a weighted undirected graph is defined as a triplet $G_\theta \equiv (R, E_\theta, F)$ where $R$ corresponds to the vertex set, the edge set $E_\theta \subseteq R \times R$ is defined by vertex pairs whose symmetric similarity function $F$ is above the threshold $\theta$

$$E_\theta = \big\{\{u,v\} \subset R : u \neq v \wedge F(u,v) \geq \theta\big\},$$

and the weight of every edge $e \in E_\theta$ is given by $F(e)$. From now on, subscript $\theta$ shall be removed for clarity.

A *clique* in graph $G$ is defined as a completely connected subgraph of $G$, i.e., any subset $R' \subseteq R$ such that for every pair of nodes $u, v \in R'$ the pair $\{u, v\}$ belongs to $E$. The *Edge-Weighted Maximum Clique Problem* requires to find the clique in $R$ that maximises the sum of weights:

$$R'_{\max} = \mathop{\arg\max}_{\substack{R' \subseteq R \\ R' \text{ clique in } G}} \sum_{u,v \in R'} F(u,v).$$

The introduced stochastic local search (SLS) algorithm (see [70] for a thorough introduction) is based on the Reactive Local Search optimisation heuristic for Maximum Clique [10](RLS-WMC, in the following WMC for short), with a novel dynamic behaviour adapted from [9].

The algorithm essentials are those of a Reactive Tabu Search applied to the weighted MC problem. The main procedure is sketched in Algorithm 4.1. Starting from an initial configuration (subset of vertices) $\bar{R} \subseteq R$, which corresponds to an empty clique (line 9), the search proceeds performing local moves by adding or removing vertices from the current candidate solution (lines 15-19). In this way, at each iteration $t$ only a set of feasible neighbouring solutions — cliques in the graph — is evaluated. A greedy choice is made among neighbouring solutions (see lines 2-3 in Algorithm 4.2). If a clique is found with an average weight greater than that of the current one, then it is taken as new current candidate solution

**Algorithm 4.1** The main section of WMC: the local search step is repeated and the best clique is returned

```
1: input: edge-weighted undirected graph $G = (R, E, F)$
2: output: best clique $\bar{R}$ found
3:
```

| Variable | Meaning |
|---|---|
| $t$ | Current iteration index |
| $T$ | Tabu tenure |
| $L_v$ | Last iteration when $v \in R$ was added/removed |
| $\bar{R}$ | Current configuration |
| $P$ | List of nodes that can be added to $\bar{R}$ |
| $w$ | Clique weight |
| $v$ | Chosen node |
| $a$ | Action to be taken (`Add` or `Drop`) |

```
4: procedure WMC($G$)
5:     for all $v \in R$ do
6:         $L_v \leftarrow -\infty$          ▷ initialise tabu attributes for each vertex in the graph
7:     end for
8:     $t \leftarrow 0$
9:     $\bar{R} \leftarrow \emptyset$          ▷ initialise current solution
10:    $P \leftarrow R$
11:    $w \leftarrow 0$
12:    repeat
13:        UpdateProhibition($\bar{R}, T$)          ▷ update tabu attributes
14:        $(v, a) \leftarrow$ ChooseNode($L, \bar{R}, P, T, t, G$)          ▷ choose best improving non-tabu neighbour
15:        if $a =$ Add then
16:            $\bar{R} \leftarrow \bar{R} \cup \{v\}$
17:        else
18:            $\bar{R} \leftarrow \bar{R} \setminus \{v\}$
19:        end if
20:        recompute $P$ and $w$ incrementally
21:        $L_v \leftarrow t$
22:        if too many iterations without improvements then
23:            Restart()          ▷ escaping mechanism against stagnation
24:        end if
25:        $t \leftarrow t + 1$
26:    until termination condition is met
27:    return $\bar{R}$
28: end procedure
```

(line 14). The algorithm repeats search steps until a termination condition is met.

Tabu attributes $L_v$ are associated to each one of the vertices in the graph. They store the last iteration in which the same vertex was part of a visited configuration. The aim of tabu attributes is to prohibit revisiting already visited configurations, for a certain period $T$, called *tabu tenure*. This prevents the algorithm from cycling between suboptimal solutions and getting trapped into local optima. The strategy implies the acceptance of worsening steps (see lines 4-7 in Algorithm 4.2) in the case in which the current solution quality can not be improved due for example to the prohibition.

The choice of the *tabu tenure* value has a strong impact on the capacity of the algorithm of diversifying or intensifying the search. Small values of $T$ tend to be insufficient for the system to efficiently escape local optima, while high values highly reduce the flexibility of the search procedure by reducing the number of eligible vertices. Rather than relying on an ideal value of $T$ as a function of the graph size and of its density, WMC adjusts it dynamically thanks to its reactive mechanism. To achieve this, a limited history of the search is maintained. If a configuration is visited too often, then the $T$ parameter is increased in order to improve the diversification capabilities of the algorithm. If, on the other hand, no configuration is revisited for a given time, $T$ is reduced. Further details on the dynamic adaptation of $T$ are available in [9].

Finally, to further improve diversification a restart mechanism is also provided for escaping from search stagnation: if the best solution is not improved in a fixed number of iterations, then the algorithm is restarted, so that new regions of the search space are possibly visited.

**Algorithm 4.2** The ChooseNode procedure: choose the non-prohibited node having the best chance to lead to better cliques in the future; if no nodes can be added, pick one for removal.

```
1: procedure ChooseNode($L, \bar{R}, P, T, t, G$)
2:     $S \leftarrow \left\{ w \in P : \begin{array}{c} L_w > t - T \wedge \\ \wedge\, w \text{ maximizes future expectations} \end{array} \right\}$
3:     $a \leftarrow$ Add
4:     if $S = \emptyset$ then
5:         $S \leftarrow \left\{ w \in \bar{R} : \begin{array}{c} L_i > t - T \wedge \\ \wedge\, w \text{ maximizes future expectations} \end{array} \right\}$
6:         $a \leftarrow$ Drop
7:     end if
8:     Pick $v \in S$
9:     return $(v, a)$
10: end procedure
```

## 4.4 Experimental Results

We experimented the proposed supervised learning approach on both tasks of predicting catalytic and metal binding sites from sequence information. For catalytic sites we considered also the prediction starting from sequence and structural information, relying on the previous results by the support vector classifier exploiting the residue structural neighbourhood as introduced in Chapter 3. The WMC algorithm is applied in cascade to the structure-based classifier for realising the collective prediction exploiting spatial information.

### 4.4.1 Predicting Active Sites

We focused on the *HA superfamily* dataset [37], the largest dataset employed as benchmark in the literature and in previous experiments in Chapter 3.

Given that most proteins contain a single active site, and the labelling found in the CSA [113] does not include information on different sites, we consider here a single site prediction setting. Common examples of multiple active sites are those of polymeric proteins in which a pair of specular sites is found at the interface of two identical chains. We plan to extract this additional information from known 3D structures in order to fully characterise overall geometry in the future.

For sequence-based prediction, we applied the proposed approach for learning a similarity function predicting whether two residues jointly participate in a certain active site. Pairs of residues are represented by concatenating their feature vectors (those described in Section 3.2.1), thus comparing residues according to their order in the sequence. This option was shown [60] to provide better results with respect to alternative approaches such as averaged pairwise comparisons, possibly because sequential ordering is relevant in characterising sites. Pairs were labeled positive if both residues bind to the same active site and negative otherwise, and an SVM was used as the pairwise classifier.

Following the same setting used for functional residue prediction on the *HA superfamily* dataset, we employed a 6 to 1 subsampling of negative (i.e. non-catalytic) residues, resulting in a 61/1 proportion of negative vs positive residue pairs. As a result of a model selection phase we employed a second degree polynomial kernel.

In building the weighted graph, edges having weight smaller than $\theta =$-0.9 are discarded, and remaining weights are rescaled to have positive values. The weight of each clique, i.e. the sum of its edge weights, was averaged over the clique dimension, i.e. the number of vertices in the clique. Following the site size distribution in training instances, we fixed the maximum size of cliques to six.

For structure-based prediction, edges were pruned for distances over $\theta =$14 Å. This threshold was chosen according to the distribution of distances between catalytic residues in the training set. The idea of constraining candidate solutions based on their pairwise 3D distances was actually used in the MBG prediction approach by Babor et al. [3] as an initial filtering stage. However the 3D constraint is much less stringent in catalytic sites, as shown by the quite large threshold (14 Å) we derived from data. Vertices weights encode catalytic propensity as predicted by the state-of-the-art support vector machine predictor described in Chapter 3. Vertices and edge weights were normalised in order to fall within the same range of values. In this case the weight of the clique is computed as the sum of edge and vertices weights divided by the number of vertices of the clique.

Experimental comparisons with the "local" approach described in Chapter 3 are shown in Table 4.1, where the protein-level precision, recall and $F_1$ measures averaged across folds are reported.

| | **SVM** | | | **SVM+WMC** | | |
|---|---|---|---|---|---|---|
| | P | R | $F_1$ | P | R | $F_1$ |
| **sequence-based predictor** | 20 ± 4 | 59 ± 7 | 25 ± 4 | 22 ± 2 | 41 ± 4 | 27 ± 3 ● |
| **structure-based predictor** | 23 ± 3 | 65 ± 6 | 28 ± 3 | 35 ± 7 | 43 ± 7 | 34 ± 6 ● |

Table 4.1: Comparison of the results (performance ± st.d.) obtained in active site prediction. A bullet indicates that the performance differences are statistically significant ($p < 0.05$).

The SVM+WMC approach achieves significant improvements at $p < 0.05$ in both sequence-based and structure-based predictions according to a paired Wilcoxon test. Note that the average protein-level $F_1$ of the local predictor is quite lower than the $F_1$ computed from average protein-level precision and recall. This happens because the local SVM produces rather unbalanced predictions, either maximising recall with low precision or (more rarely) vice versa, and for a number of proteins it outputs completely wrong predictions. The SVM+WMC approach is much more stable and balanced in its predictions. Note also that the improvement in $F_1$ is not simply due to a better choice of the decision threshold with respect to the standard local approach. The best $F_1$ value which could be obtained with local sequence-based predictions by optimising the threshold (on the test set) is just 0.256. Results from the structured-based prediction significantly improve the current state-of-the-art thanks to an effective use of the spatial geometry information. In particular, the algorithm finds cliques that discard many of the classifier false positives.

### 4.4.2 Predicting Geometry of Metal Binding Sites

The method is also tested on the task of predicting metal binding sites in metalloproteins. We used the same setting described in [60], for predicting their geometry from the sequence and the same dataset of 199 metalloproteins, for allowing a comparison. In [60] a supervised structured output learning technique is proposed. Each residue is represented by 242 features encoding conservation profiles from multiple alignments in a window of 11 residues centred on the target residue (see also Section 3.2.1 for details on conservation profile computation). Performance are evaluated on 30 random 80/20 train/test splits. We employed a setting analogous to the case of sequence-based active site prediction, with pairs of residues represented as ordered pairs of features vectors.

All parameters concerning the SVM and the maximum weighted clique algorithm described below were selected by an inner-fold cross-validation on the training set of the first split and kept fixed for all remaining folds. As a result of this model selection phase, we employed a second degree polynomial kernel and a cost factor $j = 3$. The same thresholding and rescaling of the weighted graph in sequence-based active site prediction is applied in this case.

For metal binding site prediction, the maximum clique algorithm enumerates all the maximal cliques of size up to four.

The algorithm returns the first four non-overlapping solutions with at most four residues.

We present here a set of measures including those reported in [60]. Note that we are not trying to predict the identity of an ion (e.g. the “first” zinc, the “second” iron or so), but only the subset of residues which jointly bind the same one. Thus, when evaluating the quality of a certain clustering, we assign each ion to the cluster containing the highest number of its true ligands (if any). An equivalent approach was employed in [60]. $P_E$, $R_E$, and $F_E$ are the precision, recall, and $F_1$ of the correct assignment between a ligand and a metal ion. $P_S$, $R_S$, and $F_S$ are the precision, recall, and $F_1$ of the correct prediction of binding sites, i.e., how many sites are entirely correctly predicted over the total number of sites in the chain. $P_B$, $R_B$, and $F_B$ are the precision, recall, and $F_1$ of the correct prediction of the bonding state of the residues in the chain, i.e. regardless of which ion they actually bind. Tables 4.2 and 4.3 report the mean and standard deviation of these performance measures averaged over the 30 splits. The breakdown of these measures for proteins binding different numbers of metal ions (i.e. from 1 to 4) is also reported.

Our SVM+WMC approach achieves significant improvements over the previous structured-output approach in edge, site and bonding state prediction, as measured by paired Wilcoxon tests ($p < 0.05$).

The most significative improvement over [60] lies in the number of sites entirely correctly predicted. The overall $P_S$, $R_S$, and $F_S$, is consistently better for any number of metal ions in the protein.

| **# sites** | **SVM + WMC** | | | **[60]** | | |
|---|---|---|---|---|---|---|
| | $P_E$ | $R_E$ | $F_E$ | $P_E$ | $R_E$ | $F_E$ |
| **any** | $79 \pm 3$● | $59 \pm 5$● | $62 \pm 5$● | $66 \pm 5$ | $52 \pm 4$ | $53 \pm 4$ |
| 1 | $84 \pm 4$ | $73 \pm 7$ | $73 \pm 6$ | $66 \pm 7$ | $58 \pm 6$ | $57 \pm 6$ |
| 2 | $70 \pm 8$ | $33 \pm 5$ | $42 \pm 6$ | $67 \pm 7$ | $44 \pm 9$ | $48 \pm 9$ |
| 3 | $70 \pm 15$ | $22 \pm 8$ | $32 \pm 11$ | $69 \pm 19$ | $24 \pm 13$ | $32 \pm 12$ |
| 4 | $42 \pm 30$ | $16 \pm 13$ | $23 \pm 18$ | $42 \pm 31$ | $20 \pm 19$ | $26 \pm 22$ |
| | $P_S$ | $R_S$ | $F_S$ | $P_S$ | $R_S$ | $F_S$ |
| **any** | $42 \pm 7$● | $30 \pm 7$● | $31 \pm 7$● | $20 \pm 7$ | $17 \pm 6$ | $16 \pm 6$ |
| 1 | $50 \pm 8$ | $41 \pm 9$ | $41 \pm 9$ | $25 \pm 10$ | $22 \pm 8$ | $22 \pm 8$ |
| 2 | $25 \pm 14$ | $8 \pm 7$ | $11 \pm 9$ | $15 \pm 9$ | $7 \pm 7$ | $7 \pm 7$ |
| 3 | $23 \pm 32$ | $4 \pm 7$ | $5 \pm 11$ | $0 \pm 2$ | $0 \pm 1$ | $0 \pm 2$ |
| 4 | $9 \pm 21$ | $3 \pm 6$ | $5 \pm 9$ | $2 \pm 7$ | $1 \pm 5$ | $1 \pm 5$ |
| | $P_B$ | $R_B$ | $F_B$ | $P_B$ | $R_B$ | $F_B$ |
| **any** | $88 \pm 3$● | $63 \pm 5$ | $67 \pm 4$● | $79 \pm 4$ | $64 \pm 6$ | $64 \pm 4$ |
| 1 | $84 \pm 4$ | $73 \pm 7$ | $73 \pm 6$ | $74 \pm 5$ | $68 \pm 7$ | $65 \pm 6$ |
| 2 | $92 \pm 8$ | $45 \pm 6$ | $58 \pm 7$ | $88 \pm 5$ | $60 \pm 11$ | $66 \pm 10$ |
| 3 | $100 \pm 0$ | $34 \pm 12$ | $49 \pm 15$ | $98 \pm 5$ | $38 \pm 22$ | $50 \pm 20$ |
| 4 | $67 \pm 45$ | $25 \pm 18$ | $36 \pm 25$ | $65 \pm 44$ | $32 \pm 28$ | $40 \pm 31$ |

Table 4.2: Comparison on the metalloproteins dataset. The means and standard deviations are computed on the 30 random splits. A bullet indicates that the performance differences are statistically significant ($p < 0.05$).

| | **# sites** | | | | |
|---|---|---|---|---|---|
| | any | 1 | 2 | 3 | 4 |
| **SVM + WMC** | $27 \pm 6$● | $40 \pm 9$ | $1 \pm 4$ | $0 \pm 0$ | $0 \pm 0$ |
| [60] | $14 \pm 6$ | $20 \pm 8$ | $3 \pm 7$ | $0 \pm 0$ | $0 \pm 0$ |

Table 4.3: Experimental results on the metalloproteins dataset. $A_G$ is the accuracy at a chain level, i.e., the number of entire configurations correctly predicted. A bullet indicates that the performance differences are statistically significant ($p < 0.05$).

# Chapter 5

# CatANalyst - Catalytic Residue Predictor

In this Chapter CatANalyst [159] will be described CatANalyst is a web server for predicting catalytic residues from sequence and structural information. The server is freely accessible at the following web address:

`http://catanalyst.disi.unitn.it`

since the beginning of 2010. The web server provides sequence- or structure-based predictions depending on the available information and relies on the underlying SVM classifiers described in Chapter 3. Details about the server realisation are reported in the following sections.

## 5.1 CatANalyst Web Server Architecture

The CatANalyst web server is implemented in Python and C languages. Some core modules for PDB file parsing and information extraction are written in C language and integrated with other python written modules for querying external servers and managing other server functionalities.

The CatANalyst web-server architecture highly decouples management of the user sessions and web-interface (front-end) from the elaboration system (back-end), which includes input/output job pools, the scheduler and the elaboration engine (see Figure 5.1). The latter incorporates the most computationally intensive modules of the server. This architecture confers scalability and allows for the server to be easily expanded in the future to distribute the server load on different machines.

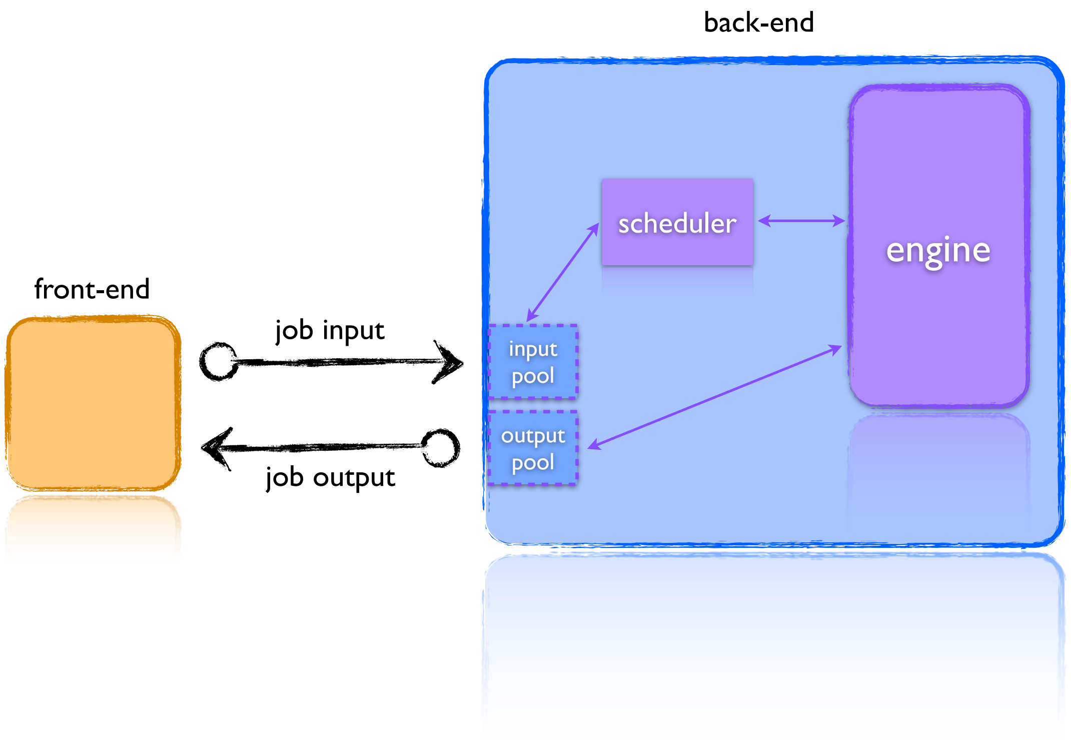


Figure 5.1: CatANalyst web server architecture.

The front-end uses mod-python, an apache module that embeds the Python interpreter within the Apache server. Front-end modules manage the graphical web-interface and sessions associated to the submitted jobs. The two main functionalities are:

(a) manage and send to the back-end a job request;

(b) display the prediction results.

The back-end is composed of three main parts:

- the input/output pools
- the scheduler
- the elaboration engine

Input and output pools are shared by both the back-end and the front-end and contain respectively the input data associated to the queued jobs and the result of the elaboration for each job.

The scheduler implements a simple policy. A job queue in the input pool is chosen according to an assigned prior probability as reported in Table 5.1. Each one of the four queues is identified by a mode (interactive or batch) and the kind of input data (primary or tertiary structure). Prior probabilities on the queues tend to favour interactive jobs that are computationally less intensive as those asking for a sequence-based prediction, therefore trying to maximise the server throughput. Within each queue the priority is given according to a First Come First Served policy for minimising the waiting time.

| | **Input Data** | |
|---|---|---|
| **Modes** | *sequence* | *structure* |
| *interactive* | 0.33 | 0.27 |
| *batch* | 0.22 | 0.18 |

Table 5.1: CatANalyst's scheduler policy and job queues.

### 5.1.1 The Elaboration Engine

The Elaboration Engine (EE) main sub-systems are:

- system of feature extraction from primary and tertiary structure;
- prediction system.

After an initial setup, needed for creating job folders and some intermediate files, the EE starts the elaboration with a feature extraction step.

Feature extraction varies depending on the type of input data available. If the input is a protein primary structure, only sequential features will be extracted, while if the input is a protein tertiary structure, both sequential and structural features will be extracted.

Starting from the protein primary sequence the name and type of the residue and a multiple alignment profile computed using Blast+[1] [28] (see also Section 3.2.1 and Table 3.1), are used for building the residue vectorial representation. From the tertiary structure, features are autonomously generated by modelling spherical regions around candidate residues and computing statistics on the neighbourhood properties (see Section 3.2.3). Whenever available in the PDB file, information on close heterogen molecules is also extracted. Additional features are extracted by querying a pool of well-known servers: solvent accessibility features from Naccess [71], relative position on the protein surface features from CASTp [17], hydrogen bonds from Molmol [84], and secondary structure features from DSSP [75].

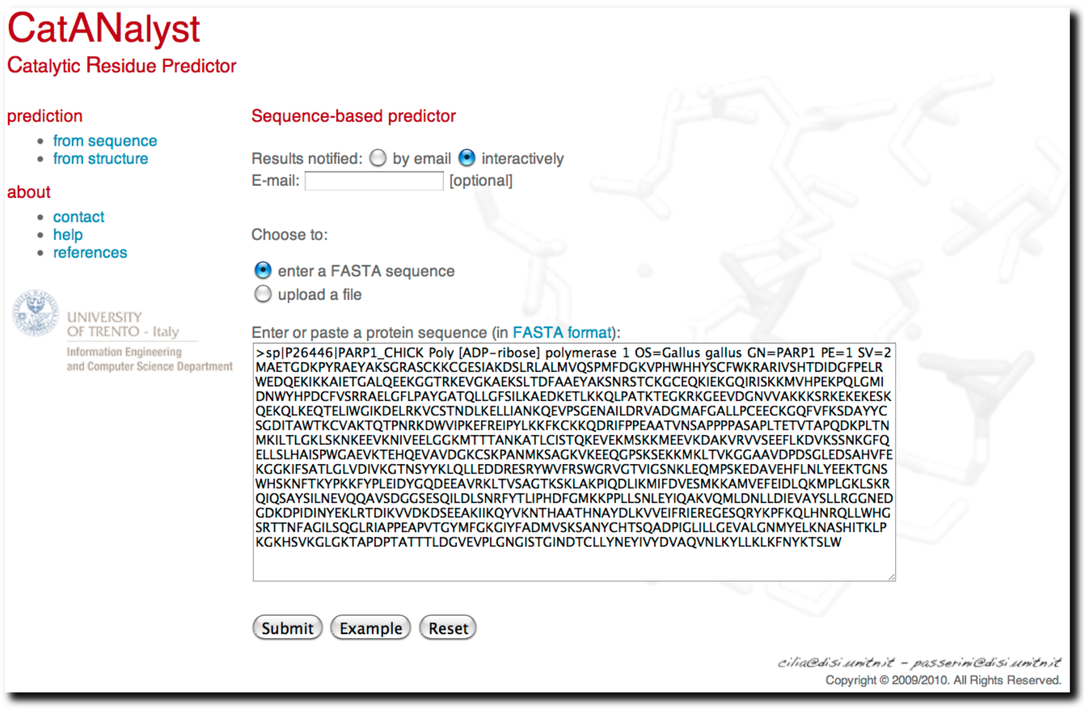
Once the feature vectors corresponding to each residue of a input protein are built, the sequence or structure based support vector classifier is used for predicting whether a residue has the characteristics for being considered a putative catalytic residue or not.

## 5.2 Querying CatANalyst

### 5.2.1 Job Submission

CatANalyst provides two different forms for submitting jobs to the sequence-based predictor (Figure 5.2) or the structure-based predictor (Figure 5.3). In the former case, the user is asked to provide as input a protein sequence

[1]version 2.2.22


CatANalyst
Catalytic Residue Predictor
prediction
from sequence
from structure
about
contact
help
references
UNIVERSITY OF TRENTO - Italy
Information Engineering and Computer Science Department
Submit
Example
Reset

Sequence-based predictor

Results notified: by email [✓ interactively]

E-mail: [optional]

Choose to:

[✓ enter a FASTA sequence]

upload a file

Enter or paste a protein sequence (in FASTA format):

```
>sp|P26446|PARP1_CHICK Poly [ADP-ribose] polymerase 1 OS=Gallus gallus GN=PARP1 PE=1 SV=2
MAETGDKPYRAEYAKSGRASCKKCGESIAKDSLRLALMVQSPMFDGKVPHWHHYSCFWKRARIVSHTDIDGFPELR
WEDQEKIKKAIETGALQEEKGGTRKEVGKAEKSLTDFAAEYAKSNRSTCKGCEQKIEKGQIRISKKMVHPEKPQLGMI
DNWYHPDCFVSRRAELGFLPAYGATQLLGFSILKAEDKETLKKQLPATKTEGKRKGEEVDGNVVAKKKSRKEKEKESK
QEKQLKEQTELIWGIKDELRKVCSTNDLKELLIANKQEVPSGENAILDRVADGMAFGALLPCEECKGQFVFKSDAYYC
SGDITAWTKCVAKTQTPNRKDWVIPKEFREIPYLKKFKCKKQDRIFPPEAATVNSAPPPPASAPLTETVTAPQDKPLTN
MKILTLGKLSKNKEEVKNIVEELGGKMTTTANKATLCISTQKEVEKMSKKMEEVKDAKVRVVSEEFLKDVKSSNKGFQ
ELLSLHAISPWGAEVKTEHQEVAVDGKCSKPANMKSAGKVKEEQGPSKSEKKMKLTVKGGAAVDPDSGLEDSAHVFE
KGGKIFSATLGLVDIVKGTNSYYKLQLLEDDRESRYWVFRSWGRVGTVIGSNKLEQMPSKEDAVEHFLNLYEEKTGNS
WHSKNFTKYPKKFYPLEIDYGQDEEAVRKLTVSAGTKSKLAKPIQDLIKMIFDVESMKKAMVEFEIDLQKMPLGKLSKR
QIQSAYSILNEVQQAVSDGGSESQILDLSNRFYTLIPHDFGMKKPPLLSNLEYIQAKVQMLDNLLDIEVAYSLLRGGNED
GDKDPIDINYEKLRTDIKVVDKDSEEAKIIKQYVKNTHAATHNAYDLKVVEIFRIEREGESQRYKPFKQLHNRQLLWHG
SRTTNFAGILSQGLRIAPPEAPVTGYMFGKGIYFADMVSKSANYCHTSQADPIGLILLGEVALGNMYELKNASHITKLP
KGKHSVKGLGKTAPDPTATTTLDGVEVPLGNGISTGINDTCLLYNEYIVYDVAQVNLKYLLKLKFNYKTSLW
```

cilia@disi.unitn.it - passerini@disi.unitn.it

Copyright © 2009/2010. All Rights Reserved.

Figure 5.2: Input form for the sequence-based CatANalyst prediction.

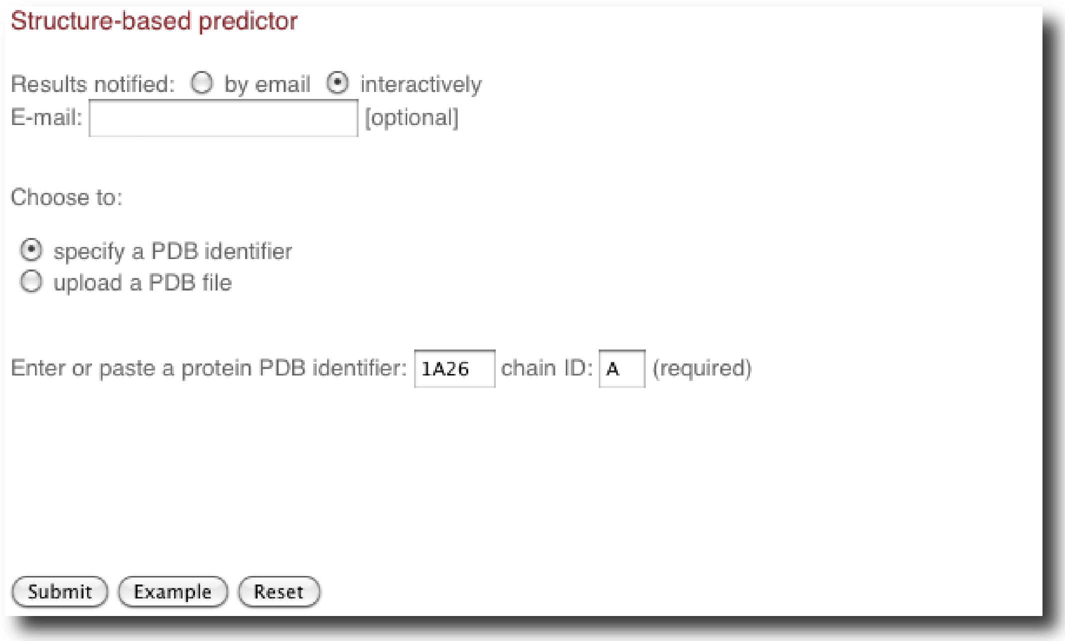
in FASTA format. In the latter case, the user submits as input a PDB identifier of the protein structure and a chain identifier. Alternatively, she can directly upload a file in PDB format, always specifying the protein chain to be analysed.

Two possibilities are offered to the user: (a) to use the CatANalyst in an interactive way, i.e. waiting for the results being delivered on the browser page, or (b) to ask for an e-mail notification once the results of the elaboration become available.

Details on the web server usage can be found in the web-site help pages[2].

An example that automatically fills the input forms is also available on the web-site by clicking on "Example" button (see Figure 5.2 and 5.3).

[2]http://catanalyst.disi.unitn.it/help


Structure-based predictor

Results notified: by email [✓ interactively]
E-mail: [optional]

Choose to:

[✓ specify a PDB identifier]
upload a PDB file

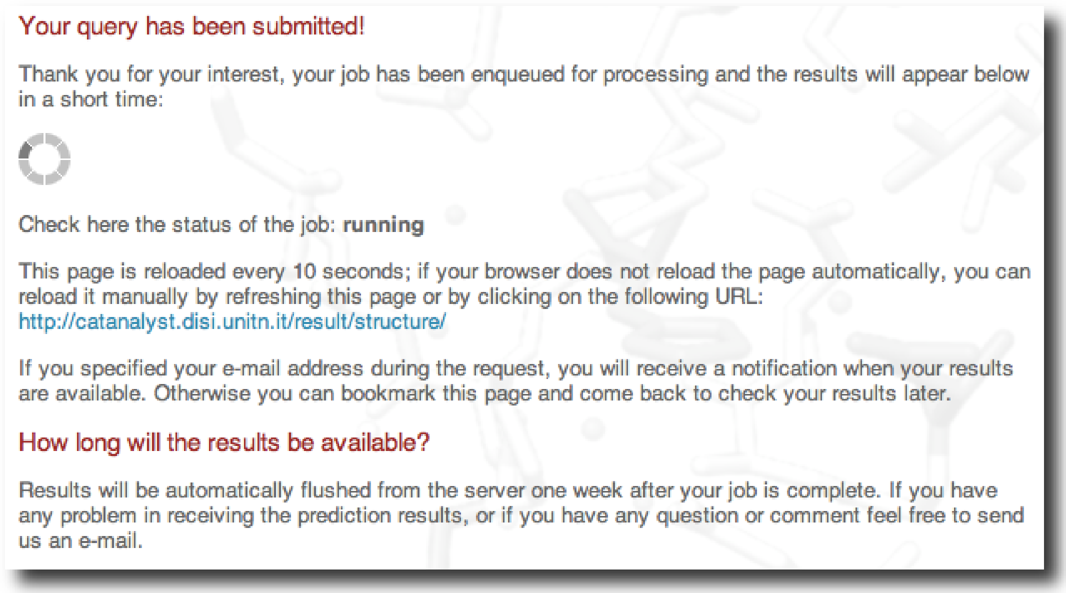
Enter or paste a protein PDB identifier: 1A26 chain ID: A (required)

Figure 5.3: Input form for the structure-based CatANalyst prediction.

### 5.2.2 Job Elaboration

Once the user submits a job, a new page is shown to her. The page is automatically refreshed every 10 seconds and reports the current job status (queued or running) (see Figure 5.4). When the job elaboration is completed and the results are available they will be displayed on the same page.

The user can bookmark the page and check the results later when they will become available. Alternatively she can leave the browser window open and check the status of her job until completion. If she asked for an e-mail notification, she will be notified as soon as the results will be available by an e-mail containing the link to follow for visualising the results.



Your query has been submitted!

Thank you for your interest, your job has been enqueued for processing and the results will appear below in a short time:

Check here the status of the job: **running**

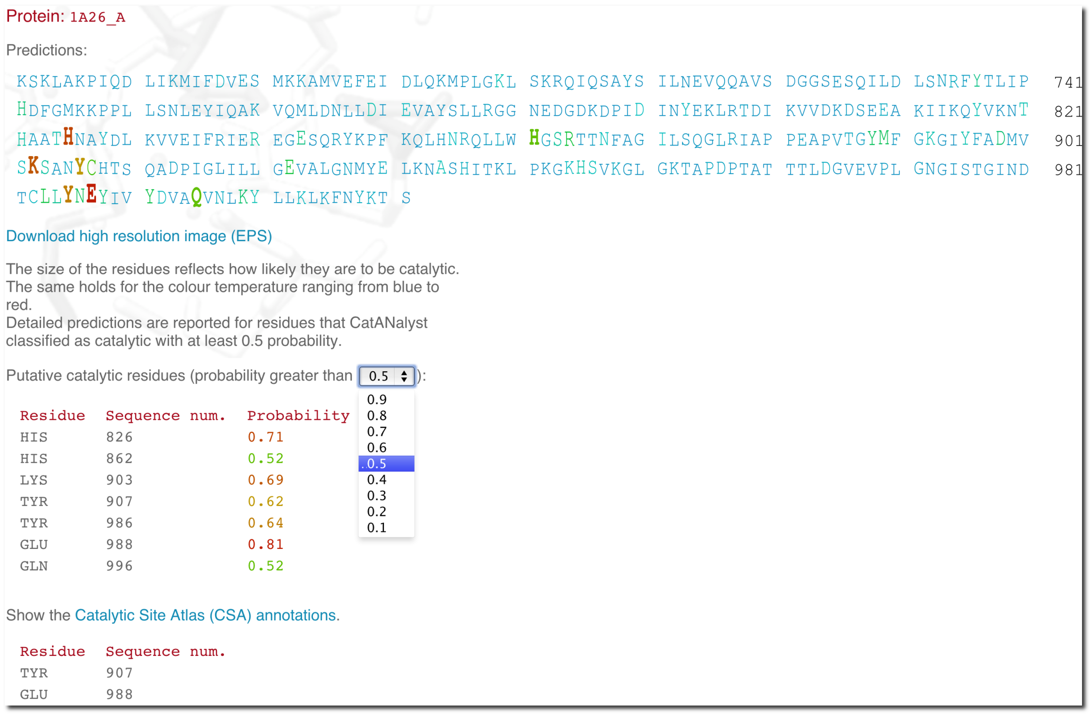
This page is reloaded every 10 seconds; if your browser does not reload the page automatically, you can reload it manually by refreshing this page or by clicking on the following URL: http://catanalyst.disi.unitn.it/result/structure/

If you specified your e-mail address during the request, you will receive a notification when your results are available. Otherwise you can bookmark this page and come back to check your results later.

How long will the results be available?

Results will be automatically flushed from the server one week after your job is complete. If you have any problem in receiving the prediction results, or if you have any question or comment feel free to send us an e-mail.

Figure 5.4: Intermediate elaboration page showing the job status.



Protein: 1A26_A

Predictions:

```
KSKLAKPIQD LIKMIFDVES MKKAMVEFEI DLQKMPLGKL SKRQIQSAYS ILNEVQQAVS DGGSESQILD LSNRFYTLIP   741
HDFGMKKPPL LSNLEYIQAK VQMLDNLLDI EVAYSLLRGG NEDGDKDPID INYEKLRTDI KVVDKDSEEA KIIKQYVKNT   821
HAATHNAYDL KVVEIFRIER EGESQRYKPF KQLHNRQLLW HGSRTTNFAG ILSQGLRIAP PEAPVTGYMF GKGIYFADMV   901
SKSANYCHTS QADPIGLILL GEVALGNMYE LKNASHITKL PKGKHSVKGL GKTAPDPTAT TTLDGVEVPL GNGISTGIND   981
TCLLYNEYIV YDVAQVNLKY LLKLKFNYKT S
```

Download high resolution image (EPS)

The size of the residues reflects how likely they are to be catalytic.
The same holds for the colour temperature ranging from blue to red.
Detailed predictions are reported for residues that CatANalyst classified as catalytic with at least 0.5 probability.

Putative catalytic residues (probability greater than 0.5):

| Residue | Sequence num. | Probability |
|---|---|---|
| HIS | 826 | 0.71 |
| HIS | 862 | 0.52 |
| LYS | 903 | 0.69 |
| TYR | 907 | 0.62 |
| TYR | 986 | 0.64 |
| GLU | 988 | 0.81 |
| GLN | 996 | 0.52 |

Show the Catalytic Site Atlas (CSA) annotations.

| Residue | Sequence num. |
|---|---|
| TYR | 907 |
| GLU | 988 |

Figure 5.5: Web page showing the results of the CatANalyst predictions.

### 5.2.3 Result Visualisation

CatANalyst outputs the prediction results by rendering the protein residues with different colour temperatures and sizes, reflecting how likely they are to be catalytic (see Figure 5.5 for an example). Probabilities of being catalytic are also shown for residues predicted with more than 0.5 confidence. To this aim, SVM margins are rescaled in the [0,1] range passing them through a sigmoid function. A different confidence threshold can be selected if desired, as well as probabilities for the entire sequence (see Figure 5.5). It is also possible to visualise the annotations reported in the Catalytic Site Atlas [3] (CSA) [113] if available, and readily compare them with the CatANalyst predictions.

## 5.3 Performance Evaluation

In order to train and test the online version of the predictors, we collected the set of enzymes contained in PDBselect and annotated in the CSA. We employed the 505 enzymes deposited in PDB before January 2008 for training and the 88 newly deposited ones for testing, resulting in 137,116 train and 24,849 test residues respectively. Results are reported in Table 5.3. Consistently with the previous comparisons, we employed a 6:1 subsampling of negative to positive examples on the training set. The performance obtained are consistent with those obtained with a 10-fold cross-validation on other benchmark datasets (compare results in Table 5.3 with those reported in Chapter 3).

Note that the model we choose for CatANalyst tends to favour recall with respect to precision. CatANalyst allows to shift the prediction threshold as explained in Section 5.2.3. In Figure 5.6 we report the Recall/Precision curves for the sequence-based (blue) and structure-based (green)

[3] version 2.2.11

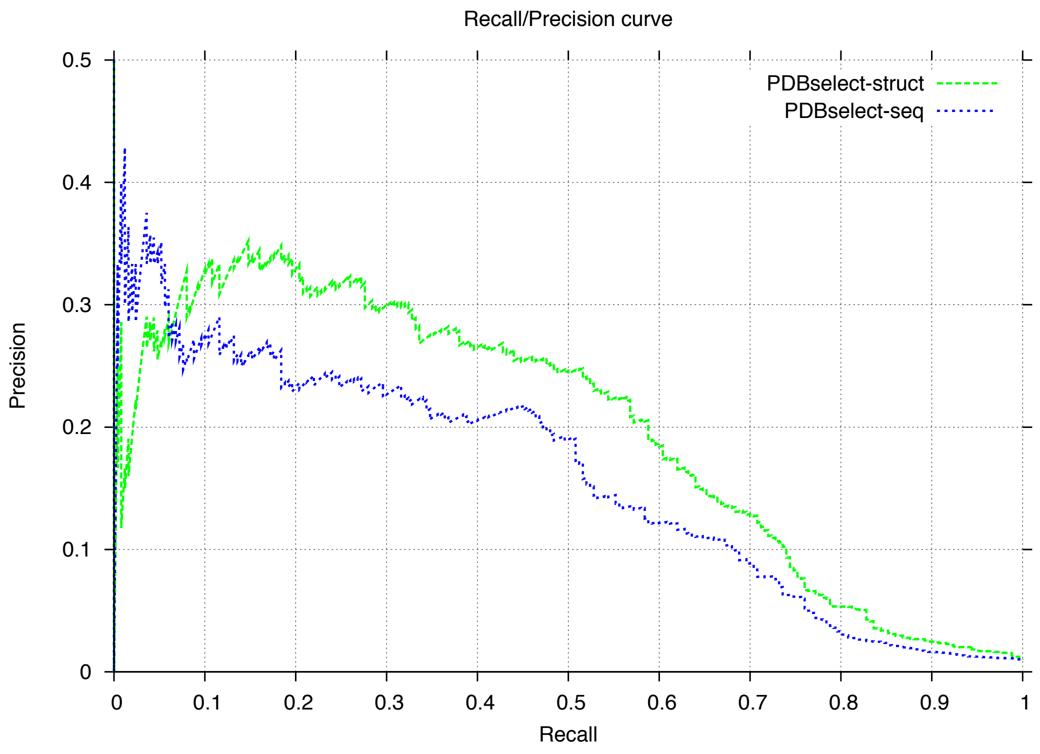


Figure 5.6: Recall/Precision curves of the CatANalyst online sequence-based (blue) and structure-based (green) predictors on the PDB-select dataset.

predictors. The curves can be easily compared to those we reported in Chapter 3 on other benchmark datasets. Note that a performance worsening of the structure-based predictor with respect to the sequence-based one can be observed at very low levels of recall (below 0.07). This is due to a slight positive bias on residues, especially HIS or ASP having a high catalytic propensity, that bind catalytic cofactors (especially zinc ions). These residues are not considered as having a direct role in the catalysis according to the CSA annotation rules. This observation suggests to develop models able to jointly predict the different roles of catalytic and binding site residues.

As a case study we tested the structure-based predictor on an amidase (SsAH) [45] whose structure has not been resolved. We used as input to CatANalyst a predicted model of the tertiary structure the authors

**Performance %**

| **CatANalyst** | P | R | FPR | $F_1$ | $MCC$ |
|---|---|---|---|---|---|
| *PDBselect-seq* | 16.1 | 51.6 | 2.7 | 24.6 | 27.6 |
| *PDBselect-struct* | 18.3 | 61.6 | 2.8 | 28.2 | 32.4 |

Table 5.2: Precision (P), recall or sensitivity (R), False Positive Rate (1-specifity), $F_1$ measure and Matthews Correlation Coefficient ($MCC$) results for test proteins in PDBselect (rows one and two).

in [154] were confident with. CatANalyst predictions include the known catalytic residues (K96, S195, S171) [45]. It also confirms the experimental results shown in previous work [153], for example a possible role of R197, K209 and D228 in affecting the active site, and identifies an additional candidate residue (E142), whose putative role could be verified by site-directed mutagenesis.

## 5.4 Statistics

The CatANalyst server is online since the beginning of December 2009 and it has been tested on thousands of different protein sequences and structures from the PDB during the development. Six research groups and companies outside of the author's group have been involved in the testing phase.

CatANalyst web-site has currently about three visits per-day. Since its online publication it has received more than 900 visits, and submissions from more than twenty countries around the world.

## 5.5 Computational Time Issues

Many of the CatANalyst tasks of feature extraction can take a long time depending on many different factors. We observed that one of the most costly tasks in terms of latency is the computation of the conservation profiles.

Conservation profiles encoding evolution-based similarity among sequences in a multiple alignment are computed by using the Position-Specific Iterative Blast Search (PSI-Blast) [28] on a database of non redundant protein sequences (as explained in Chapter 3).

In order to reduce the latency in the delivery of the answer to a queried protein we are precomputing the conservation profiles for a large number of protein sequences of the UniProtKB [131]. Currently more than 8000 profiles have been precomputed.

## 5.6 Work In Progress

The CatANalyst structure-based classifier shows a slight positive bias on residues having a high catalytic propensity that bind catalytic cofactors. These residues have no direct role in the catalysis according to the CSA annotation, as explained in Chapter 2. On the other hand, information on nearby heterogens helps in identifying functional residues with low catalytic propensities. Planned extensions include the joint prediction of both catalytic and binding site residues and the collective prediction of residues belonging to the same catalytic site by incorporating the approach described in Chapter 4.

# Part II

# Mining Relevant Features from Mutation Data

# Chapter 6

# Learning from Mutations

In this second part of the manuscript the focus is on the mining of relevant relational features from enzyme mutation data. It shows that the use of the mined features is not only important for improving the understanding of an enzyme function but also for predicting the evolution of viral enzyme mutants. The proposed approaches have a general validity and can be applied straightforwardly to proteins other than enzymes.

## 6.1 Introduction and Background

*Advances in molecular genetics reveal that many diseases stem from specific protein defects.*

***David Whitford in "Protein Structure and Function"***

Random mutagenesis is a technique that aims at generating a library of mutants of the same enzyme (the *wild type*) to be screened. A mutant has incurred mutations, i.e. changes in the DNA sequence, with respect to its *wild type*. In the present study the interest is on the so called mis-sense mutations, in other words in mutations that change the protein amino acid. Mutations that, due to the redundancy of the Genetic Code, do not change the amino acid, are called silent or synonyms or same-sense mutations.

The mutations induced by random mutagenesis can lead to either inactive mutants or more convenient mutants showing an improved biological activity or they can be mutations that do not affect the enzyme activity (silent mutations). Protein mutation data are not only characterised by the mutant sequences. They typically include mutant activities assayed on several groups of structurally related substrates. By analysing common patterns of mutation and their correlation with the experimentally measured mutant activity, it is possible, for instance, to formulate hypotheses on the functional sites and the amino acids involved in them [153].

The sequence-based predictor described in Chapter 3 has been used for predicting functional residues in mutants of the same enzyme — an amidase (SsAH) [45]— produced by random mutagenesis. Mutant functional residue predictions were analysed by correlating them with those of the wild type enzyme. An example is shown in Figure 6.1. 36 mutants and their activities on 14 substrates belonging to three structural groups were screened. By coupling and validating the computational analysis with spatial considerations on a predicted 3D model of the enzyme tertiary structure (see Figure 6.2), residues in positions 173 and 209 were highlighted as important for affecting the catalytic site.

This analysis helped to gain insights about the putative role of some residues in affecting the catalytic site. However, the case study highlighted the need of a suitable approach for screening a large number of mutants and mining relevant features from the related data. The mined features should highlight the properties of protein functional sites and of the residues involved in them. Therefore, relational and statistical relational learning approaches are here investigated for mining enzyme mutation data.

Random mutagenesis is not only used with the mere purpose to gain insights on a protein function or analyse a protein structure. It is also used in protein engineering applications. Indeed, the design of novel proteins with

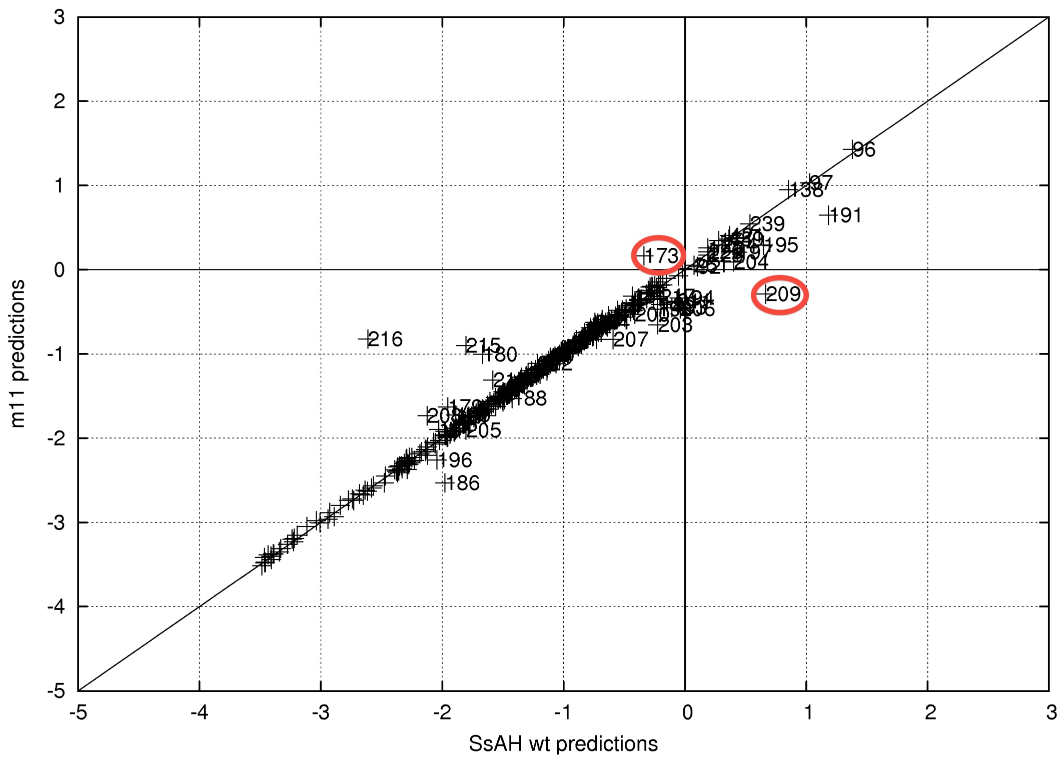


Figure 6.1: Correlation plot of the prediction margins of the residues in the wild type enzyme (x axis) and one of its mutants (y axis).

useful function passes through the understanding of the structure-function relationship and the analysis functional sites. Here is also proposed a relational learning approach that starting from the mined relevant features builds novel instances of the learnt concept. An example of learnt concept is “a mutant resistant to a certain inhibitor”.

The analysis of mutations proved to be extremely important in molecular genetics studies aiming at developing effective cures to a number of diseases stemming from specific protein defects [143].

Diseases like the cystic fibrosis, cancer and Human Immunodeficiency Virus (HIV) infection all depend in some way on specific mutations taking place into specific proteins — for the cystic fibrosis even a single mutation over a total of 1500 amino acids. The HIV case is an emblematic example of infection that encompasses many aspects of protein structure and function.

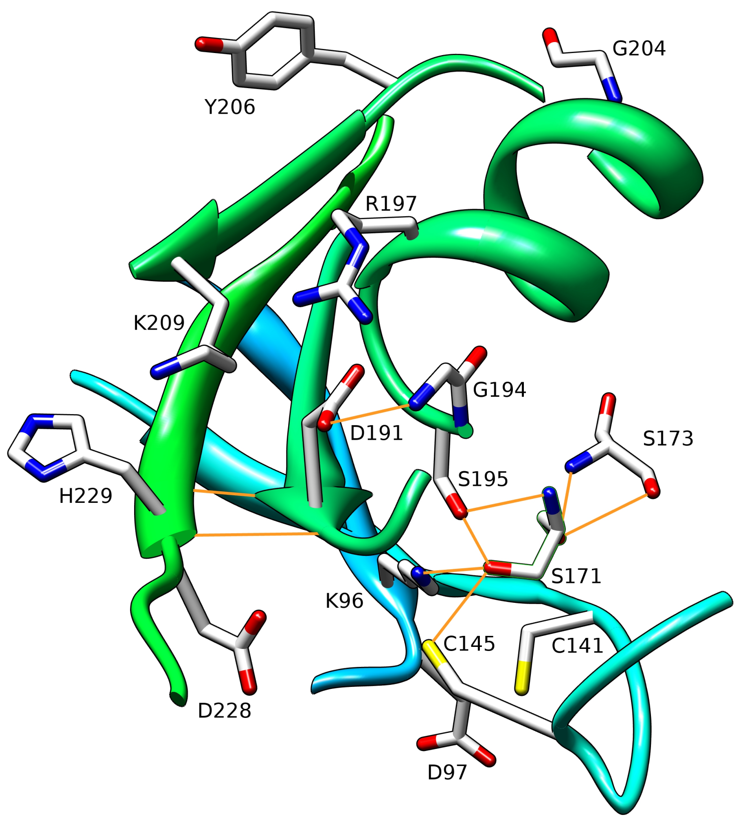


Figure 6.2: Portion of the predicted 3D model of SsAH including the active site.

The virus enters white blood cells through the interaction of specific viral proteins with receptors on the cell surface. Once in the cell, through the action of specific enzymes, the virus integrates viral DNA into the host cell for its replication.

The reverse transcriptase (RT) is the DNA polymerase enzyme that transcribes RNA into DNA, allowing it to be integrated into the genome of the host cell and replicated along with it, and is therefore crucial for the virus propagation. Stopping an HIV infection requires the understanding of the structure and functioning of these viral enzymes. Thanks to its high mutation rate, the virus is able to quickly develop resistance to specific drugs. The analysis of mutations and the discovery of those residues that are necessary for the proper functioning of virus proteins are extremely important. Based on this information more effective inhibitors can be designed and the virus evolution can be possibly anticipated. Even a single mutation can confer the mutant resistance to one or more drugs by modifying the inhibitor target site on the protein and consequently avoiding its interaction with the drug. HIV mutants typically have multiple mutations and understanding the role of each one of them with respect to the development of the resistance is quite difficult.

Many types of RT inhibitors have been designed for fighting against HIV infection, which rely on quite different mechanisms. The NonNucleoside RT Inhibitors (NNRTIs) and the NonCompetitive RT inhibitors (NCRTIs) inhibit the reverse transcriptase by binding to the enzyme active site, therefore directly interfering with the enzyme function. A Nucleoside RT Inhibitor (NRTI) is instead incorporated into the newly synthesised viral DNA for preventing its elongation. Finally the Pyrophosphate Analogue RT Inhibitor (PARTI) targets the pyrophosphate binding site and it is employed, as part of a salvage therapy, on patients in which the HIV infection shows resistance to the other classes of antiretroviral drugs. Each of these four classes of inhibitors includes a number of specific drugs or substrates (see Table 7.1 in Chapter 7). HIV mutation data typically consist of a number of HIV protein mutants whose resistance has been evaluated with respect to several groups of related inhibitors.

Figure 6.3 shows an example of the inhibition of the reverse transcriptase. The Nevirapine drug, an NNRTI, by binding just at the back of the active site, changes the active site shape interfering with the RT function.

Recently the research for halting HIV is going into the direction of studying HIV controllers, i.e. patients that are infected by HIV but do not get the Acquired Immune Deficiency Syndrome (AIDS). Knowing how these patients "naturally" fight off AIDS could help in providing that same resistance to other patients who has contracted HIV. Analysing and comparing

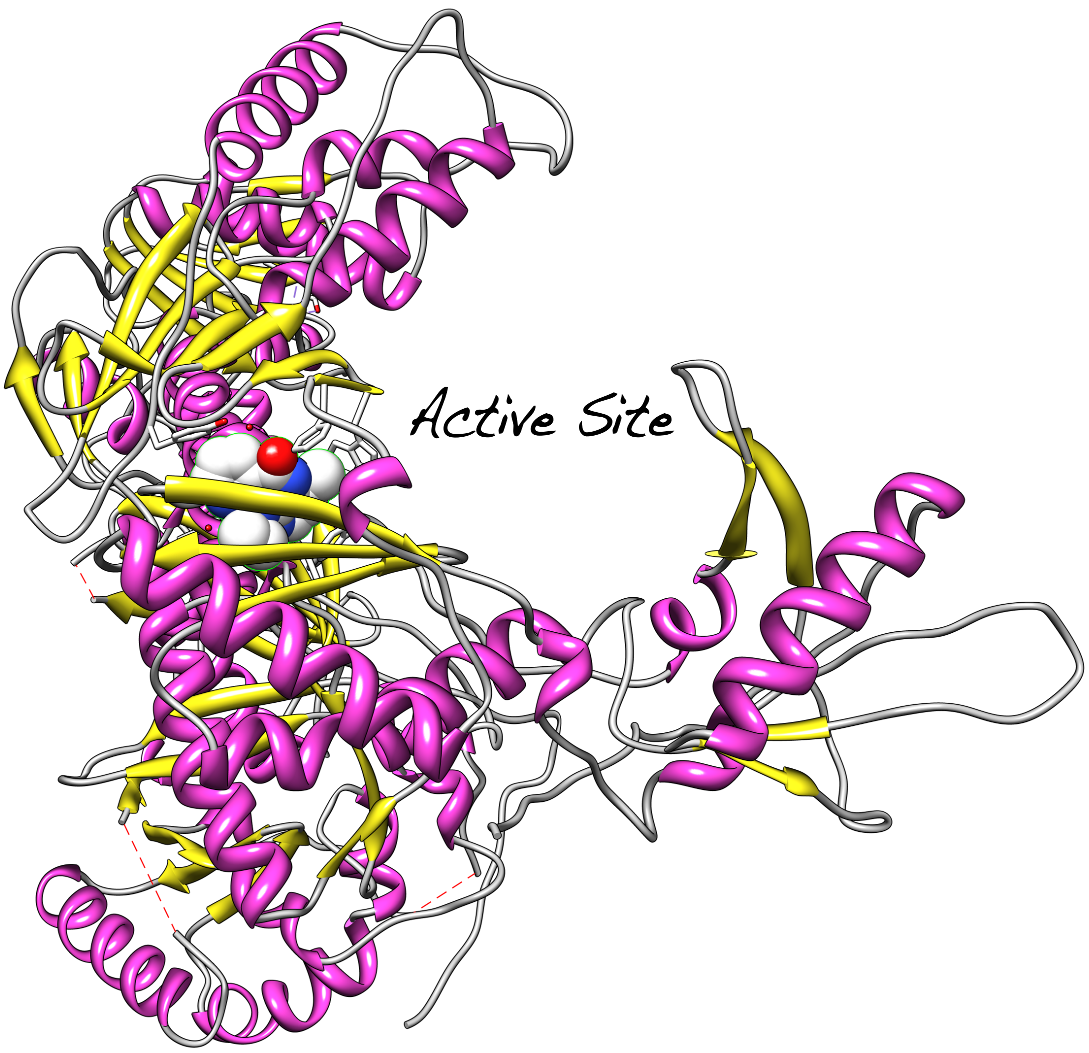


Figure 6.3: Reverse transcriptase inhibition by Nevirapine (PDB code 1VRT).

mutations of collected human genetic samples in [127] some genetic variations have been discovered. All of them are related to a protein HLA-B which is essential in the immune system process of destroying infected cells. This again underlines the importance of analysing and studying mutations and the need of developing tools for mining the relevant protein variations and their characteristics.

In this part of the thesis, two different approaches are proposed for "learning from mutations". The focus is on resistance of HIV RT to drug treatment. The first approach has the primary aim to mine relevant features from mutation data, which can explain the mechanism involved in the mutant mutated response to a drug. A statistical relational approach is proposed for learning relational features.

The catalytic activity of an enzyme and its mutants is usually evaluated on a number of different substrates (drugs). Exploiting the information on related classification tasks can be useful for improving predictive performance. Therefore, a multitask learning setting is employed for exploiting the availability of resistance data for multiple inhibitors. The proposed approach extends the notion of multitask learning in a relational setting to a hierarchical multitask classification, providing the ability to learn rules that are themselves hierarchical. The second proposed approach exploits relational learnt rules modelling resistance mutations. Its aim is to generate novel mutations that can confer resistance to an inhibitor and therefore anticipate the virus evolution.

## 6.2 State of the Art

Many machine learning methods have been applied in the past to mutation data for predicting single point mutations on protein stability changes [32, 33, 40] and the effect of mutations on the protein function [104] [26] or HIV drug susceptibility [118]. All but one of the above mentioned approaches are based on SVMs or neural networks. In [118] the performance of five different statistical learning methods (for example, decision trees, neural networks and support vector regression) are compared in the prediction of mutant drug susceptibility. An extensive review of approaches for the prediction of drug resistance mutations can be found in [30].

Albeit being effective binary classification algorithms, SVMs are not ideal for mining relevant features or general rules that can be readily interpreted by human experts. A first attempt of learning rules from mutation data, can be found in [120]. In this work relational associations among mutations are extracted by means of an ILP learner.

Statistical relational learning approaches offer not only the ability to learn human interpretable rules, typical of ILP algorithms, but also the incorporation of statistical robustness principles needed to handle noise, from the statistical learning theory. kFOIL [86, 87] is a statistical relational learner that combines techniques from inductive logic programming —specifically, the FOIL algorithm [115]— with kernel methods.

Multitask learning [34] is an active research area and various techniques have been proposed in the literature. The underlying idea is that of introducing a form of parameter sharing among tasks. In feed-forward neural networks [34], this is achieved by learning a common hidden layer for the tasks, while in kernel machines a matrix encoding the tasks relationship is included in the regularisation term [55]. In a fully Bayesian framework, hierarchical Bayesian models [11] introduce a common prior distribution for the task-specific models, favouring parameter sharing among related tasks. These models allow for a great flexibility in modelling different levels of tasks relatedness. Bakker and Heskes [5] for instance, generalise standard multitask feed-forward neural networks with prior distributions over the output layer weights. In the task gating approach, they allow for both task clustering, by using a mixture model for the prior distribution, and task-specific mixing proportions for the clustering. However, the number of clusters needs to be pre-specified according to the available domain knowledge, or determined by model selection strategies. Non-parametric Bayesian approaches have gained increasing popularity in recent years as they allow to overcome this problem, by sampling distributions over parameters rather than parameters directly. Dirichlet processes [56] have been employed for multitask learning [147] in order to automatically select the appropriate number of clusters for task-dependent parameters. Approaches to learn an entire latent hierarchy of tasks have been also proposed [47, 102]. These approaches, however, assume that task models can be eventually represented as parameter vectors.

In a full relational setting, either domain knowledge allows to explicitly encode relationships between tasks, or one needs to resort to multitask relational structure learning, and specifying priors over task-dependent structures is quite challenging. The same observation can be made on the quite different setting considered in Chapter 7 in which hybrid statistical-logical models made up of first-order clauses are learnt. It would be quite challenging to devise prior distributions for such discrete structures. Indeed, multitask structure learning itself has received little attention in the statistical relational learning setting. A notable exception is the work by Deshpande et al. [52] on learning multitask probabilistic relational rule sets. Their approach assumes that the prior information shared among tasks is a set of rule *prototypes*. Task-specific rules are derived from these prototypes by a generative probabilistic process involving prototype selection and modification steps or rule generation from scratch. From the point of view of learning hierarchies of concepts in relational domains, the works on infinite relational models [78, 146] are also worth mentioning. Here latent indicator variables on entities or concepts are introduced on top of the relational structure, allowing to cluster them using Dirichlet process priors. The approach has also been extended [121] to discover a hierarchical structure of concepts. While these models do not perform structure learning in terms of logical hypotheses, their ability to infer arbitrary clustering structures is an appealing feature to be integrated in a multitask learning approach.

kFOIL has been recently extended for dealing with multitask learning [87]. It exploits multitask information for learning a general-to-specific hierarchy of logical hypotheses, represented as relational kernel functions, to be coupled with appropriate task-specific weights. The multitask kFOIL was the first attempt to address multitask kernel learning in a statistical relational learning context.

In the ILP community, multitask learning has been tackled in the form of learning several related concepts simultaneously. Different approaches have been pursued. In [116], the assessment of candidate clauses on the primary task is augmented with the performance of similar rules on a secondary task. Furthermore, a scenario resembling multitask learning has been studied in [46], where (sub)structures of concepts already learnt are used as building blocks when learning a new concept. A further related scenario is that of *repeat learning* and *multiple predicate learning* [50, 80], where an ILP learner has to discover a series of related concepts drawn from some (initially unknown) distribution. Moreover, *predictive clustering trees* have been used in an ILP setting. These trees can be used in a multitask setting, where predictions for several tasks are made at every leaf [18, 19].

# Chapter 7

# Hierarchical Multitask kFOIL

This chapter proposes a novel approach for mining relevant features from enzyme mutation data [155]. The proposed statistical relational learning approach implements a simple hierarchical extension of the multitask kFOIL learning algorithm. The algorithm addresses the limitations of learning a single relational structure for multiple tasks, by taking a hierarchical approach. It first learns a core logic representation common to all tasks, and then refines it by specialisation on a per-task basis. The approach can be easily generalised to a deeper hierarchy of tasks. A task clustering algorithm is also proposed in order to automatically generate the task hierarchy. The approach is validated on the problem of drug-resistance mutation prediction, as well as on a protein structural classification problem [156]. Experimental results show the advantage of the hierarchical version over both single and multi task alternatives and its potential usefulness in providing explanatory features for the domain. Task clustering allows to further improve performance when a deeper hierarchy is considered. A major advantage of the adopted strategy is the ability to provide explanations for the learnt models which are themselves hierarchical: a subset of relational features relevant to all tasks can be identified together with more specific task-dependent ones.

## 7.1 Introduction

kFOIL [86] is a statistical relational learner that, by taking a discriminative viewpoint, greedily learns a relational kernel representation with high discriminant power for a certain task. The algorithm was recently [87] extended to deal with multitask problems by learning a shared relational representation which is discriminative for multiple tasks simultaneously. However, learning a single common representation prevents the model from discovering task-specific features, a problem affecting multitask neural networks as well. Moreover learning a single common representation can be highly suboptimal if tasks relatedness is not high [94]. We propose a simple extension where the common representation is viewed as an initial structure which is further specialised on a per-task basis. This simple approach can be generalised to a deeper hierarchical process of refinements whenever a hierarchical clustering of the tasks is available or can be learnt from data.

The hierarchical kFOIL algorithm is applied to a dataset of HIV resistance mutations [120]. The dataset reports wild type and mutations of the reverse transcriptase, a viral protein which is essential for the success of the viral propagation. A mutation can confer the mutant resistance to one or more drugs, for instance by modifying the inhibitor target site on the protein. Due to the high mutation rate of viruses, mutants typically have multiple mutations, ranging from 6 to 90 on this dataset. A possible problem in this setting is predicting which drugs a certain mutant is resistant to. This can be naturally addressed as a multitask learning problem, where each drug is a single task. However, the relationship between tasks is not necessarily strong as different drugs can target different sites in the protein. Indeed, plain multitask learning will sometimes result in a performance worsening with respect to single task in this setting, especially when drug classes are considered as tasks [120]. On the other hand, the proposed hierarchical refinement approach succeeds in combining the advantages of the two methods, being always at least as good as either alternative. Task clustering allows to further improve performance when a finer grain of tasks is obtained by considering individuals drugs as tasks instead of drug classes. The hierarchical multitask approach is also applied to a protein structure classification problem, within the SCOP [72] hierarchy. In a setting in which quite distantly related tasks are combined, the hierarchical approach is always significantly better than both single task and multitask alternatives. Nonetheless, the main concern here is not the predictive performance itself, but rather the ability of the method to provide insights into the reasons for a certain resistance. From this viewpoint, multitask and hierarchical approaches have an additional advantage: the models learnt can be inspected in order to relate general and task-specific features.

## 7.2 kFOIL: Learning Relational Kernels

This section briefly reviews the original kFOIL algorithm [86] and its multitask extension [87].

### 7.2.1 The kFOIL Algorithm

kFOIL [86] is a statistical relational learning approach that combines techniques from inductive logic programming — specifically, the FOIL algorithm [115] — with kernel methods. ILP systems such as FOIL learn a first-order logical hypothesis for the target concept. This approach has several advantages, including that a large and flexible space of hypotheses is considered, and that the final model is readily interpreted by human experts. On the negative side, ILP methods do not always incorporate statistical robustness principles needed to handle noise, and searching in a large, discrete space of hypotheses can be challenging. In contrast, kernel-based learning methods are well-suited to deal with noisy training data, and learning reduces to a much simpler convex optimisation problem for which globally optimal solutions can be found. It is possible to apply kernel methods to relational data by manually defining a kernel function for relational instances and using this function in standard kernel-based learners. However, this approach lacks the flexibility of an ILP-style hypothesis search, as relevant relational features have to be encoded *a priori* in the kernel function, rather than being discovered automatically from data. It also limits the amount of domain insight obtainable from the learnt model, as no new relational features are induced during learning.

kFOIL integrates the clause search of ILP approaches with kernel methods, by learning a set of interpretable first-order clauses from data that define a relational kernel function. Thus, ILP methods are used as a form of structure learning to induce a suitable kernel function from data. This approach has the appealing potential of combining the advantages of both approaches, namely the flexibility and interpretability of ILP-style clause search and the robustness of kernel-based learners.

The simplest way to introduce a relational kernel function $k(x_1, x_2)$ based on a set $H = \{c_1, ..., c_n\}$ of first-order clauses is to propositionalise the examples $x_1$ and $x_2$ using $H$ and then employ existing kernels on the resulting vectors, which we will refer to as the *feature space representation* of the examples. We will thus map each example $x$ onto a vector $\phi(x)$ over $\{0, 1\}^n$ with $\phi(x)_i = 1$ if the $i$-th clause $c_i \in H$ covers the example $x$, and $\phi(x)_i = 0$ otherwise. Figure 7.1 shows an example of the propositionalising function $\phi(x)$ and feature space representation of examples m261 and m1636 in the HIV resistance domain. Each mutant is mapped to a vector with one entry for each clause, evaluating to one if the clause fires on the example (i.e. the clause and the background knowledge logically entail

**Algorithm 7.1** kFOIL algorithm.

```
1: procedure kFOIL($H_0, \mathcal{D}, \mathcal{B}$)
2:     Initialise $H \leftarrow H_0$
3:     repeat
4:         Initialise $c := p(X_1, \cdots, X_n) \leftarrow$
5:         repeat
6:             $c := \arg\max_{c' \in \rho(c)} S(H \cup \{c'\}, \mathcal{D}, \mathcal{B})$
7:         until stopping criterion
8:         $H := H \cup \{c\}$
9:     until stopping criterion
10:    return $H$
11: end procedure
```

it) and zero otherwise. A linear kernel in this representation amounts to counting the number of rules firing on both examples. For details on the logic representation employed see Section 7.5.1.

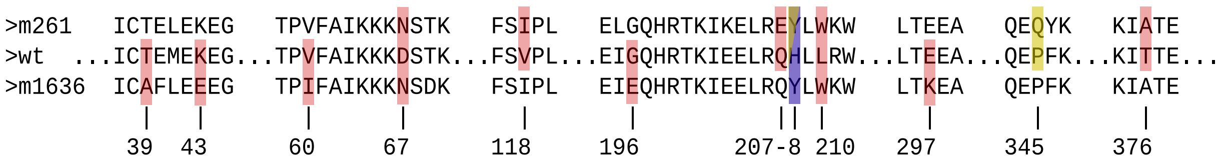
kFOIL integrates ILP and kernel-based learning by solving an integrated optimisation problem given by (see also Section 1.4.3):

$$\max_{H \in \mathcal{H}} S_O \left( \arg\max_{f \in \mathcal{F}_H} S_I(f, \mathcal{D}, \mathcal{B}), \mathcal{D}, \mathcal{B} \right) \tag{7.1}$$

Here, $\mathcal{H}$ denotes the logical hypothesis space under consideration, i.e. the set of all possible sets of clauses. $\mathcal{H}$ is defined using language bias declarations similar to those employed in FOIL and other ILP algorithms. $\mathcal{F}_H$ denotes the model space of a kernel machine working on the feature space representation given by $H$. $S$ denotes a scoring function that measures the predictive performance of $f(x; H, \mathcal{B})$ on the training data $\mathcal{D}$, and $\mathcal{B}$ the available logical background knowledge. $S_O$ and $S_I$ are the scoring functions used for hypothesis and function learning respectively.

Learning the hypothesis $H$ requires to solve the outer optimisation problem.

kFOIL follows the well-known FOIL algorithm [115] for learning a set of clauses from data. The search procedure is sketched in Algorithm 7.1. In



```
>m261   ICTELEKEG   TPVFAIKKKNSTK   FSIPL   ELGQHRTKIKELREYLWKW   LTEEA   QEQYK   KIATE
>wt  ...ICTEMEKEG...TPVFAIKKKDSTK...FSVPL...EIGQHRTKIEELRQHLLRW...LTEEA...QEPFK...KITTE...
>m1636  ICAFLEEEG   TPIFAIKKKNSDK   FSIPL   EIEQHRTKIEELRQYLWKW   LTKEA   QEPFK   KIATE
          |   |        |       |         |         |            || |      |       |       |
         39  43       60      67       118       196         207-8 210   297     345     376
```

🟦 `mut(A,h,C,y),position(C,208)`

🟨 `mut(A,B,C,y),position(C,208),`
`mut(A,D,E,q),color(red,D)`

🟥 `mut(A,l,C,w),mut(A,D,E,i),`
`mut(A,F,G,a), mut(A,H,I,e),`
`correlated_mut(A,J,I),`
`mut(A,K,L,e),same_type(K,F),`
`mut(A,M,N,y),mut(A,O,P,n)`

$$\phi(\texttt{m261}) = \begin{pmatrix} 1 \\ 1 \\ 1 \end{pmatrix} \quad \phi(\texttt{m1636}) = \begin{pmatrix} 1 \\ 0 \\ 1 \end{pmatrix}$$

$$k(\texttt{m261},\texttt{m1636}) = \langle \begin{pmatrix} 1 \\ 1 \\ 1 \end{pmatrix}, \begin{pmatrix} 1 \\ 0 \\ 1 \end{pmatrix} \rangle = 2$$

Figure 7.1: Example of a relational kernel function for the HIV resistance mutation database. An alignment between the wild type and two mutants is reported with colours highlighting positions which jointly satisfy the corresponding clause. The resulting feature space representation and mutants similarity for a linear kernel are reported in the lower right part.

an outer loop, kFOIL repeatedly searches for clauses that score well with respect to the data set and the current hypothesis, and adds them to the current hypothesis. The initial hypothesis $H_0$ is the empty set in the standard algorithm, but it will be a partial model when the procedure will be used in the hierarchical version of the algorithm described in Section 7.3. In the inner loop, kFOIL greedily searches for a clause that scores well. To this aim, it employs a general-to-specific hill-climbing search strategy. Let $p/n$ denote the predicate that is being learnt. Then the most general clause, which succeeds on all examples, is "$p(X_1, ..., X_n) \leftarrow$". This clause serves as a starting point for the hill-climbing search. The set of all refinements of a clause $c$ within the language bias is produced by a *refinement operator* $\rho(c)$. Clauses are greedily refined until a stopping criterion is met, and the highest-scoring clause encountered during the search is added to the hypothesis $H$. The joint optimisation of the hypothesis $H$ and kernel machine $f$ reflected in Equation (7.1) and Algorithm 7.1 means that kFOIL falls into the framework of *dynamic propositionalisation* [85]. This is in contrast to *static propositionalisation* approaches that decouple propositionalisation of the data and propositional learning.

Several scoring functions $S(H \cup \{c'\}, \mathcal{D})$ for candidate hypotheses have been proposed. One class of scoring functions uses the performance of a kernel machine trained on the feature space representation of the data $\phi(D)$ obtained from $H \cup \{c'\}$, by setting

$$S(H, \mathcal{D}, \mathcal{B}) = S_O\left(\arg\max_{f \in \mathcal{F}_H} S_I(f, \mathcal{D}, \mathcal{B}), \mathcal{D}, \mathcal{B}\right). \tag{7.2}$$

These measures require that the statistical learner is trained for each candidate clause, and its performance on the training set is reported. An efficient alternative consist of using kernel target alignment (KTA) [44], defined as:

$$S(H, \mathcal{D}, \mathcal{B}) = \frac{\langle K, yy^T \rangle_F}{\sqrt{\langle K, K \rangle_F \langle yy^T, yy^T \rangle_F}} \tag{7.3}$$

where $K$ is the kernel matrix resulting from $H$, $y \in \{-1, 1\}^m$ is the target vector for $m$ examples, $y^T$ is the transpose of $y$, and the Frobenius product is defined as $\langle M, N \rangle_F = \sum_{ij} M_{ij} N_{ij}$. Intuitively, the alignment measures how the kernel adheres to a "perfect" kernel, scoring one and minus one for examples belonging to the same and different classes respectively, and is thus an indication of the performance a kernel machine can reach using it.

As a stopping criterion, the original FOIL algorithm stops when it fails to find a clause that covers additional positive examples. As an equally simple stopping criterion, learning in kFOIL is stopped when there is no improvement in score between two successive iterations.

### 7.2.2 Multitask kFOIL

The original kFOIL algorithm has been recently extended to a multitask setting [87]. Multitask learning in kFOIL is based on sharing the learnt feature representation (or, equivalently, the relational kernel function) across tasks. This can be achieved by learning a single joint set of clauses for all tasks under consideration, such that all task-specific kernel machines trained on this representation achieve good performance in their respective prediction tasks. In a multitask setting, Algorithm 7.1 is thus adapted by replacing the single-task scoring function $S(H, \mathcal{D}, \mathcal{B})$ by an appropriate multitask scoring function, which is obtained as a combination of single-task scoring functions on the individual tasks. Assume that $S(H, \mathcal{D}, \mathcal{B})$ is an (outer) scoring function as given by Equation (7.2) or Equation (7.3), and that $\mathcal{D}_1, ..., \mathcal{D}_M$ are the available training data for $M$ tasks $T_1, ..., T_M$. A simple but effective multitask scoring function is obtained by averaging single-task scores. That is, replacing Equation (7.2) with

$$S(H, \mathcal{D}, \mathcal{B}) = \frac{1}{M} \sum_{t=1}^{M} S_O \left( \arg\max_{f \in \mathcal{F}_H} S_I(f, \mathcal{D}_t, \mathcal{B}), \mathcal{D}_t, \mathcal{B} \right), \tag{7.4}$$

or equivalently, replacing Equation (7.3) with

$$S(H, \mathcal{D}, \mathcal{B}) = \frac{1}{M} \sum_{t=1}^{M} \frac{\langle K, y_t y_t^T \rangle_F}{\sqrt{\langle K, K \rangle_F \langle y_t y_t^T, y_t y_t^T \rangle}} \tag{7.5}$$

where $y_t$ is the label vector of task $t \in 1, ..., M$.

Experimental results presented in [87] show several advantages of multitask learning in the proposed setting. First, generalisation performance is improved in some cases, confirming the advantages of multitask learning observed in the literature [34]. Second, in this setting learning a shared clause set for multiple tasks leads to a more compact representation of the learnt concept, as the multitask clause set is significantly smaller than

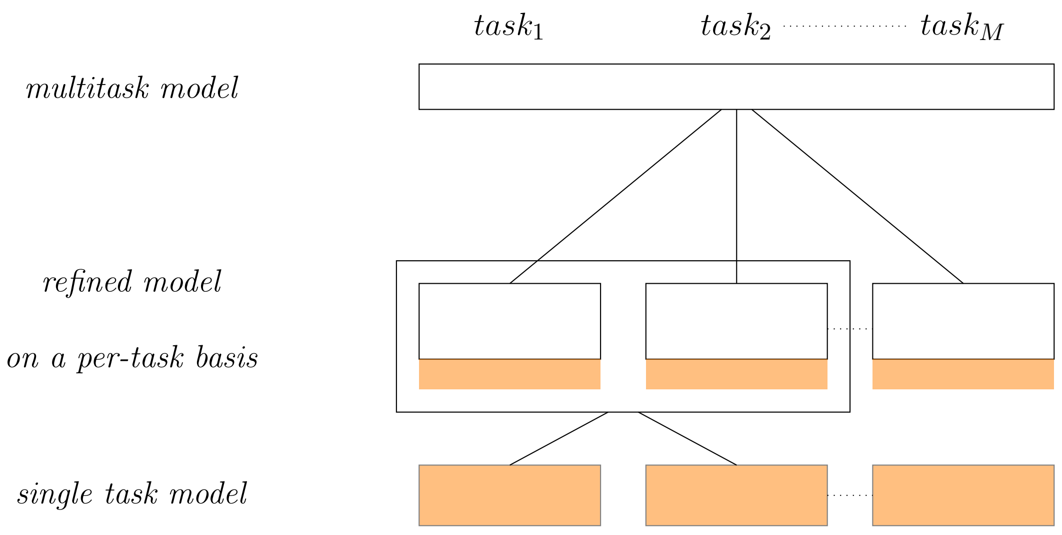


Figure 7.2: The hierarchical learning approach.

the union of the task-specific clause sets. In terms of the learnt similarity/kernel function, this yields a generic definition of similarity that is shared between tasks and should be easier to interpret than a number of task-specific similarity functions. Finally, multitask learning also results in significant computational savings.

## 7.3 Hierarchical kFOIL

Learning a representation which is common to multiple related tasks is a way to reduce the risk of overfitting the data [11]. However, it prevents the algorithm to learn specific task-dependent features, which can be harmful for some of the other tasks. Furthermore, it assumes a high relatedness among tasks, and performances can be badly affected when relatedness is not that high. We propose a simple extension to the multitask kFOIL learning algorithm dealing with this problem.

A hierarchical approach to multitask learning is taken (see Figure 7.2): the algorithm first learns a representation which is common to all tasks

**Algorithm 7.2** Hierarchical kFOIL algorithm.

```
1: procedure HIERARCHICALKFOIL($\{\mathcal{D}_1, \ldots, \mathcal{D}_M\}, \mathcal{B}$)
2:     Initialise $H_0 \leftarrow$ KFOIL($\emptyset, \{\mathcal{D}_1, \ldots, \mathcal{D}_M\}, \mathcal{B}$)    ▷ compute initial representation
3:     for all $\mathcal{D}_t \in \{\mathcal{D}_1, \ldots, \mathcal{D}_M\}$ do
4:         $H_t \leftarrow$ KFOILREFINE($H_0, \mathcal{D}_t, \mathcal{B}$)    ▷ compute task-dependent refinements
5:     end for
6:     return $\{H_1, \ldots, H_M\}$
7: end procedure
```

by using the multitask kFOIL; then such initial representation is refined separately for each task, leading to task-dependent final representations obtained as extensions of a common core. Algorithm 7.2 shows the resulting hierarchical kFOIL learning system, where $\mathcal{D}_1, \ldots, \mathcal{D}_M$ are the datasets for the $M$ different tasks. For simplicity we assumed a common background knowledge $\mathcal{B}$, but it is straightforward to replace it with task-specific background knowledge, as well as a task-specific language bias defining the clause refinement operator $\rho$.

The refinement stage is described in Algorithm 7.3 and consists of two steps. First, each clause from the initial representation is further refined guided by the task-specific score. If the (possibly) refined clause fails to improve the score, it is not added to the task-specific model. Then, the model from the previous step is enlarged by creating novel clauses using the plain kFOIL procedure. In principle, when refining general clauses, specialising the initial representation is not the only available option. We also implemented a search operator that considers both specialisation and generalisations of the current clause, by greedily adding or removing a single literal. As this did not change results significantly in the experimental settings, section 7.5 only reports the results obtained with the specialisation operator.

A detailed analysis of computational complexity for both single task and multitask kFOIL is reported in [87], showing the increase in efficiency

**Algorithm 7.3** kFOIL refinement algorithm.

```
1: procedure kFOILRefine($H_0, \mathcal{D}, \mathcal{B}$)
2:     Initialise $H \leftarrow H_0$
3:     for all $c \in H_0$ do                          ▷ refinement of existing clauses
4:         $H \leftarrow H \setminus \{c\}$
5:         repeat
6:             $c \leftarrow \arg\max_{c' \in \rho(c)} S(H \cup \{c'\}, \mathcal{D}, \mathcal{B})$
7:         until stopping criterion
8:         if score improvement then
9:             $H \leftarrow H \cup \{c\}$
10:        end if
11:    end for
12:    $H \leftarrow$ kFOIL($H, \mathcal{D}, \mathcal{B}$)                 ▷ search for novel clauses
13:    return $H$
14: end procedure
```

of the latter especially when KTA scoring is employed. The additional complexity of the refinement stage depends on the number of single-task clauses added: the more related the tasks are, the more the multitask clauses will already explain them and the refinement size will be limited.

The hierarchical kFOIL algorithm can be easily generalised to multiple levels of refinements whenever tasks can be naturally aggregated into a hierarchical structure. When the existence of a hierarchy can be guessed but its structure is unknown, the approach can be combined with a task-clustering algorithm as suggested by Thrun and O'Sullivan [132].

## 7.4 Task Clustering

The rationale of multitask learning is that predictive performance on a certain task should improve if information from related tasks can be exploited. However, performance worsening can also be experienced when trying to transfer information between unrelated tasks. A *selective* transfer [132] approach would be advisable. In such approach the transfer occurs only among tasks related one to each other. This requirement can be combined with the potential advantages of a multi-level representation by pursuing a hierarchical clustering approach. We basically adapt the task-clustering algorithm of Thrun and O'Sullivan [132] to our setting in which multitask learning is seen as a way to improve performance among training tasks rather than on novel test ones, and the clustering structure is used to learn interpretable hierarchical models and discover hierarchical rules relating tasks and groups of tasks.

Most clustering approaches rely on a measure of similarity (or distance) among instances. Intuitively, two tasks are similar if they have the same output over the same (or similar) examples. Whenever few (or no) examples are shared between tasks, we cannot just rely on them in order to compute a reliable measure of similarity. However, we can simply fill missing labels using predicted ones. Assume $\mathcal{D}$ is the overall set of training instances, and $\mathcal{D}_i$ its subset having labels for task $i$. We compute missing task-dependent labels for $\mathcal{D} \setminus \mathcal{D}_i$ as those predicted by a model for task $i$ trained on $\mathcal{D}_i$. Repeating the procedure for all tasks, we compute pairwise task similarity as:

$$sim(i,j) = \frac{1}{|\mathcal{D}|} \sum_{x \in \mathcal{D}} \delta(\hat{f}_i(x), \hat{f}_j(x)) \tag{7.6}$$

where $\hat{f}_k(x)$ is the true task-specific label $y_k$ for $x$ if available, or the predicted one $f_k(x)$ otherwise. Learning a model for each task requires providing a representation for instances. We rely on the initial common representation $H_0$ (see Algorithm 7.2) and use it to derive task-dependent models, pairwise similarities and resulting hierarchical clustering. We employed an agglomerative hierarchical clustering approach with average pairwise similarity between elements for cluster similarity. Once the clustering structure is available, a multi-level refinement can be applied. Note however that the amount of transfer and the granularity of the refinement structure can be adaptively adjusted to the problem at hand. At each node in the clustering structure, two choices are possible, depending on the similarity between the node children: 1) learn a node-dependent model, and refine it for each child; this would be the default strategy to achieve a fully hierarchical model; 2) learn a node-dependent model, and use it within each child; this option would reduce the granularity of the hierarchical structure by merging together nodes containing similar tasks. The choice between these two options can be made separately for each choice point, by specifying a similarity threshold $\theta_{max}$, or jointly by constraining the size of the representational structure (e.g. three levels from the root to the task-specific models). Algorithm 7.4 reports the pseudo-code of the hierarchical kFOIL procedure with task clustering. The initial representation $H_0$ obtained by running the plain multitask kFOIL on all tasks is used to infer a hierarchical clustering rooted at $C_0$. The representation $H_0$ is then refined on a per-node basis for each child of $C_0$, using the subset of $\mathcal{D}$ containing examples for tasks in $C_i$. $\textsc{sim}(C_0)$ returns the similarity between the children of $C_0$, needed to decide on the amount of transfer. Algorithm 7.5 describes the node-dependent refinement for a generic node $C$. If the similarity value $s$ between the node and its siblings is below $\theta_{max}$, or the node contains a single task, a node-refined representation is generated calling the kFOIL refinement algorithm, implementing case 1 of node choices previously described. Otherwise, no node-wise refinement is performed (case 2) and the parent model is directly passed to the children.

**Algorithm 7.4** Hierarchical clustering kFOIL algorithm.

```
1: procedure HierarchicalClustKFOIL($\mathcal{D}, \mathcal{B}$)
2:     Initialise $H_0 \leftarrow$ kFOIL($\emptyset, \mathcal{D}, \mathcal{B}$)          ▷ compute initial representation
3:     $C_0 \leftarrow$ HierarchicalClustering($H_0, \mathcal{D}, \mathcal{B}$)          ▷ compute clustering
4:     for all $C_i$ children of $C_0$ do          ▷ compute node-dependent refinements
5:         $\mathcal{D}_i \leftarrow$ subset of $\mathcal{D}$ involving tasks in $C_i$
6:         ClusterNodeKFOIL($H_0, C_i, \text{sim}(C_0), \mathcal{D}_i, \mathcal{B}$)
7:     end for
8: end procedure
```

**Algorithm 7.5** kFOIL algorithm for cluster node.

```
1: procedure ClusterNodeKFOIL($H_0, C, s, \mathcal{D}, \mathcal{B}$)
2:     if $s \leq \theta_{max} \vee$ isLeaf($C$) then
3:         $H \leftarrow$ kFOILRefine($H_0, \mathcal{D}, \mathcal{B}$)
4:     else
5:         $H \leftarrow H_0$
6:     end if
7:     for all $C_i$ children of $C$ do
8:         $\mathcal{D}_i \leftarrow$ subset of $\mathcal{D}$ involving tasks in $C_i$
9:         ClusterNodeKFOIL($H, C_i, \text{sim}(C), \mathcal{D}_i, \mathcal{B}$)
10:    end for
11: end procedure
```

## 7.5 Experimental Evaluation

We evaluated our hierarchical approach on datasets from two domains, namely HIV drug resistance mutations and SCOP protein structure classification hierarchy. In all experiments the KTA (see Equation 7.3) was used as scoring function for guiding the search of the kFOIL algorithm, as it is more efficient even if less effective [87] in general than measures based on a trained kernel machine. A simple linear kernel was employed in order to maximise the understandability of the learnt models. Note that, rather than pushing the performance of the learning algorithm by fine tuning its parameters, we are more interested in comparing the respective advantages of the different approaches and evaluate their explanatory power. The Hierarchical kFOIL program is freely available online [1]. The package also includes all the prediction results and learnt models of the experiments reported in the following sections.

### 7.5.1 Predicting Drug-Resistance of Mutants

Viruses are characterised by a very high mutation rate, which allows them to quickly develop drug-resistant strains. A single- or multiple-point mutation can confer the mutant resistance to one or more drugs, for instance by modifying the inhibitor target site on the protein. Predicting the drug-resistance of mutants can be of valuable help in designing more effective drugs, especially if interpretable models can be provided. This can be addressed as a multitask learning problem, where each drug is a single task. However, the relationship between tasks is not necessarily strong as different drugs can target different sites in the protein. A hierarchical approach seems a natural candidate in this setting. We focused on HIV, both for the impact of the virus and the availability of annotated databases of mutants.

#### The HIV Resistance Mutations Datasets

We experimented on a dataset of mutations from the Los Alamos National Laboratories (LANL) HIV resistance database[2]. The dataset was derived in [120] and is composed of 2,339 mutants of the HIV RT. Richter et al. [120] formulated the learning problem as a mining task and applied a relational association rule miner to derive rules relating different mutations and their resistance properties. We take a slightly different approach here and provide supervision at the mutant rather than mutation level. A mutant is considered resistant to a drug if it contains at least one observed resistance

[1]http://www.disi.unitn.it/~passerini/software/HkFOIL.tgz
[2]http://www.hiv.lanl.gov/content/sequence/RESDB/

mutation to that drug. We derived two different versions of the dataset, considering resistance to drug classes or specific drugs respectively.

**drug class resistance dataset** We selected the three classes of drugs: (a) NonNucleoside RT Inhibitors (NNRTI); (b) NonCompetitive RT inhibitors (NCRTI); (c) Pyrophosphate Analogue RT Inhibitors (PARTI). In the dataset 1081 mutants are labelled as resistant to NNRTI, 75 to NCRTI and 53 to PARTI. We ignored the Nucleoside RT Inhibitors (NRTI) since all the mutants in this dataset had at least one mutation conferring resistance to that class of drugs.

**drug resistance dataset** We used all the four classes of drugs from the original dataset, thus including the NRTI class. We specialised the labelling on a single-drug basis, extracting this information directly from the LANL HIV resistance database when available. Table 7.1 reports the drugs belonging to the four inhibitor classes and the number of mutants resistant to each of them. In this way we deepen the hierarchy at the single drug level with the aim of testing the usefulness of the hierarchical clustering approach.

#### Background Knowledge

A relational knowledge base is built for the domain at hand. Table 7.2 summarises the predicates included as a background knowledge. We represented the amino acids of the wild type with their positions in the primary sequence (`aa/2`) and the specific mutations characterising them (`mut/4`). Target predicates were encoded as resistance of the mutant to a certain drug (`res_against/2`).

Additional background knowledge was included in order to highlight characteristics of residues and relationships between mutations:

| Drug class | Drug | # mutants |
|---|---|---|
| NRTI | Zidovudine (azt) | 2211 |
| | Lamivudine (3tc) | 1356 |
| | Abacavir (abc) | 422 |
| | Zalcitabine (ddc) | 336 |
| | Didanosine (ddi) | 280 |
| NNRTI | Efavirenz (efv) | 883 |
| | Nevirapine (nvp) | 833 |
| | Delavirdine (dlv) | 796 |
| | ADAMII | 114 |
| | Trovirdine | 113 |
| NCRTI | MSK-076 | 75 |
| PARTI | foscarnet | 53 |

Table 7.1: Hierarchical task structure for the HIV resistance mutation datasets with associated number of instances.

| **Background Knowledge Predicates** | |
|---|---|
| `aa(Pos,AA)` | indicates a residue in the wild type sequence |
| `mut(Mutant,AA,Pos,AA1)` | indicates a mutation: mutant identifier, position and amino acids involved, before and after the substitution |
| `res_against(Mutant,Drug)` | indicates whether a mutant is resistant to a certain drug |
| `color(Color,AA)` | indicates the type of a natural amino acid |
| `same_type(R1,R2)` | indicates whether two residues are of the same type |
| `same_type_mut(Mutant,Pos)` | indicates a mutation to an amino acid from the same type |
| `different_type_mut(Mutant,Pos)` | indicates a mutation changing the type of residue |
| `correlated_mut(Mutant,Pos1,Pos2)` | indicates whether two mutations are correlated (see the text for the details) |

Table 7.2: Summary of the HIV drug resistance background knowledge facts and rules.

`color/2` indicates the type of the natural amino acids according to the colouring proposed in [130] and also reported in Table 7.3. For example the magenta class includes basic amino acids as lysine and arginine while the blue class includes acidic amino acids as aspartic and glutamic acids.

`same_type/2` indicates whether two residues belong to the same type, i.e.

| **Colour Class** | **Amino Acids** | **Description** |
|---|---|---|
| red | AVFPMILW | small and/or hydrophobic and/or aromatic |
| blue | DE | acidic |
| magenta | RK | basic |
| green | STYHCNGQ | hydroxyl and/or polar and/or basic |

Table 7.3: Amino acid types encoded in colour classes.

a change from one residue to the other conserves the type of the amino acid.

`same_type_mut/2` indicates that a residue substitution at a certain position does not modify the amino acid type with respect to the wild type. For example mutation d123e conserves the amino acid type while mutation d123a does not (i.e. `different_type_mut/2` holds for it).

`correlated_mut/3` states that two mutations are correlated. We considered two mutations in different positions along the primary sequence to be correlated, when they compensate reciprocally for the substitutions of the amino acids. This predicate captures simple cases like the two mutations d123a and a321d, and more complex correlations in which the changes involve not exactly the same residue but residues of the same type, like d123a and a321e or d123a and v321e.

### Hierarchical Multitask Experiments

The hierarchical kFOIL algorithm is evaluated on the drug class resistance dataset, with three tasks corresponding to the three drug classes. Results are compared with the two alternatives of: 1) considering three separate single task learning problems; 2) considering a single common multitask learning problem with no per-task refinement. We performed 3-fold cross-validation procedures stratified at the single-task level to ensure a good balancing between positive and negative examples for each learning task. The area under the ROC curve (AUC) was employed as measure of performance in all experiments.

We experimented with two variants of the language bias guiding the learner, by varying the constraints on the use of the `mut/4` predicate. In the first variant (V1) the learner can extend a clause by using the predicate `mut/4` with the position variable already instantiated, and thus scoring it according to the mutants that have a mutation in that position. In the second variant (V2), in the predicate `mut/4` the position variable is not instantiated while the variable corresponding to the mutated form of the residue is instantiated instead. In the first case the search space of the learner will contain predicates like `mut(Mutant,a,123,Rnew)` with a specific position instantiated, while in the second case it will contain predicates like `mut(Mutant,Rold,Position,k)` where k indicates a change resulting in a lysine. The rationale for considering the two variants is that the former will tend to learn more specific clauses involving relationships between point-wise mutations, as for the association rules discovered in [120]. Conversely, the latter variant will be biased to learn possibly suboptimal but more general and hopefully more interesting mutation rules, trying to discover higher level patterns relating different mutations.

Table 7.4 reports the results of the two variants V1 and V2. Different behaviours can be detected for different drug classes. Overall, multitask learning achieves comparable results with respect to a standard single task approach in both variants, being twice significantly better and once significantly worse than the alternative. The result suggests that for this dataset we are not always able to take a real advantage from the multitask learning approach, possibly because classes of drugs can be quite unrelated by targeting different binding sites. By adding a refinement stage on a per-task basis, we succeed in improving the results with respect to both single task

| | V1 | | | V2 | | |
|---|---|---|---|---|---|---|
| Drug class | single task | multitask | hierarchical | single task | multitask | hierarchical |
| NNRTI | 0.95○ | 0.77 | 0.96○ | 0.68 | 0.66 | 0.71○● |
| NCRTI | 0.95 | 0.99 | 0.99● | 0.88 | 0.86 | 0.88 |
| PARTI | 0.81 | 0.95● | 0.98● | 0.65 | 0.84● | 0.90● |

Table 7.4: Summary of the hierarchical multitask experiments (KTA scoring). Statistical tests for the significance of the differences in AUC were computed for the methods within each language bias variant using the two-tailed Hanley-McNeil test [68] (p=0.05). A bullet (●) indicates that the method is significantly better than the single task approach, while a circle (○) indicates a significant improvement over the multitask one. All other differences are not statistically significant.

and multitask approaches. Hierarchical kFOIL is never significantly worse than any of the two alternatives, while being significantly better than at least one of them in five out of six cases.

#### Discussion and Rule Interpretation

Concerning the language bias variants, we can observe from Table 7.4 that as expected the instantiation of a specific position (V1) gives better results compared to searching more general rules (V2). The former models can exploit the fact that mutants resistant to the same drug often share mutations in the same positions. These could be located in the protein binding site or its vicinity, but drug resistance could also be conferred more indirectly by other conformational modifications.

Figure 7.3 shows an example of hierarchy of learnt clauses in the V1 variant setting. The root clause `mut(A,g,196,B),color(blue,B)` was learnt in the multitask models of all folds. The clause states that a mutation in position 196 changing a glycine into an acidic residue (aspartic or glutamic acid) can be important for a mutant to develop resistance to the three kinds of drugs analysed. This could provide hints for understanding how the binding works and is affected by the surrounding residues. Moreover it underlines the potential of the approach also in other contexts: for example to gain insights on the wild type protein function and on its active site starting from its mutants usually obtained by random mutagenesis. We are currently pursuing such research direction on an amidase. When inspecting the models resulting from the single task refinements, we found that for the NNRTI task the above-mentioned clause is not extended. Additional mutations are instead included in the extended clauses for the NCRTI and the PARTI tasks. This could suggest that in those cases the drug-specific resistance results from the combination with one or two other mutations.

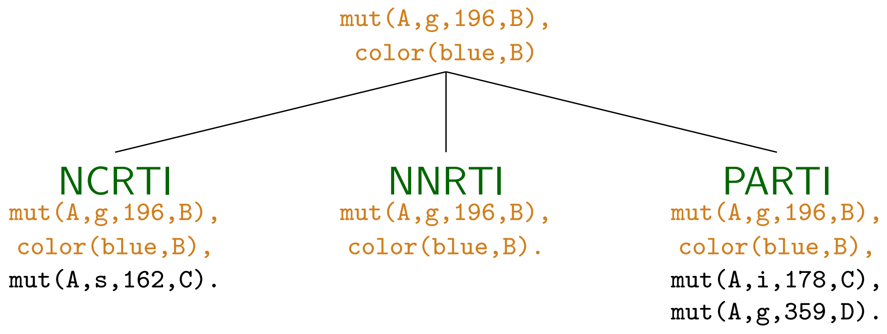


Figure 7.3: Example hierarchy of learnt clauses.

Some mutations, like the mutation of the asparagine in position 348, appear in multitask model and in all the single task models (in all the folds) with the addition of at most one correlated mutation. This seems to suggest that such mutation is important for the mutant resistance to the three drug classes. Interestingly the refined model further enriches the corresponding clause by correlating the mutation with up to three other mutations.

The learnt rules generated by kFOIL for NNRTIs, NCRTIs and PARTIs can be compared with those reported by the rule miner used in [120] which are not explicitly linked to the resistance to NRTIs. These rules correlate mutations in position 41 and 215, sometimes also with other mutations in position 277 and 293. kFOIL identified the correlation between the mutations at positions 215 and 41. This correlation was previously observed to be related to resistance to NRTIs [91]. kFOIL identifies the same correlation for the resistance to the quite different other classes of inhibitors. In particular the two mutated positions were found among the rules for the resistance to PARTIs where additional mutated position are highlighted: 67 and 376 .

The learnt models in the variant V2 contain clauses like

```
mut(A,B,C,w),mut(A,D,E,i),mut(A,F,G,l),mut(A,H,I,d),mut(A,J,K,a)
```

or

```
mut(A,B,C,g),position(C,135),mut(A,D,E,m),
                    correlated_mut(A,E,F),position(F,138)
```

which combine a quite large set of mutations, with the latter clause including an explicitly correlated pair of mutations. As a further example the refined model on the PARTI task suggests, in all folds, the clause:

```
mut(A,B,C,w),different_type_mut(A,C)
```

which highlights a mutation into a tryptophan that completely changes the type of amino acid in the mutated position. Note that the need for multiple mutations in order to induce a change in the phenotype has recently found confirmation [140] in experimental studies on molecular phenotypes. A suggested interpretation [140] states that "neutral mutations prepare the ground for later evolutionary adaptation". While this is far from being a confirmation for the specific patterns found by the algorithm, the obvious limitation of learning techniques focusing on single point mutations alone is an additional stimulus for this research direction.

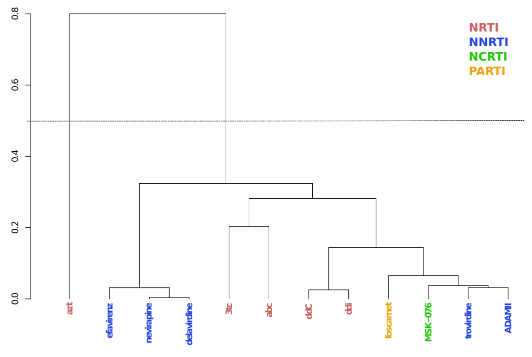
#### Hierarchical Clustering Experiments

The task clustering approach for hierarchical multitask learning is evaluated on the drug resistance dataset, which provides a deeper structure of tasks as information on both drug classes and specific drugs are available. Results are compared with the three alternatives of: 1) considering three separate single task learning problems; 2) considering a single common multitask learning problem with no per-task refinement; 3) learning a per-task refinement of an initial common multitask model as in the previous section. In order to feed the clustering algorithm with an accurate estimate of tasks similarities, the focus is on the variant of the language bias providing the better results (V1). The validation procedure is the same as for the hierarchical multitask experiments.

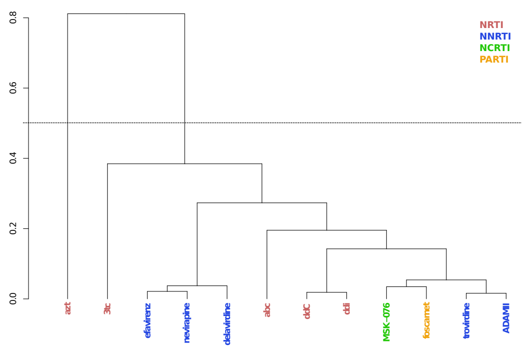
Figure 7.4 shows the dendrograms of the clusterings obtained in each one of the folds of the cross-validation. Drug classes are highlighted with different colours: red for drugs belonging to the NRTIs, blue for those belonging to NNRTIs, orange for PARTIs and green for NCRTIs. Note that the hierarchical clustering obtained is rather stable across folds. Interestingly, Zidovudine (azt) is consistently separated from the rest of the drugs and merged only at the root. A possible explanation for this behaviour is that the large number of examples available for the drug (see Table 7.1) makes the kernel machine employed in kFOIL more focused on the task and less predictive for different drugs. Indeed, multitask learning already achieves an AUC of 0.95 for azt (see Table 7.5), which is rather high considered that the resulting binary classification problem is the most balanced.

A dotted line in Figure 7.4 indicates the value of the similarity threshold employed ($\theta_{max} = 0.5$): a single intermediate level of refinement is obtained in all folds between the shared multitask model at the root and the per-task refinements at the leaves.

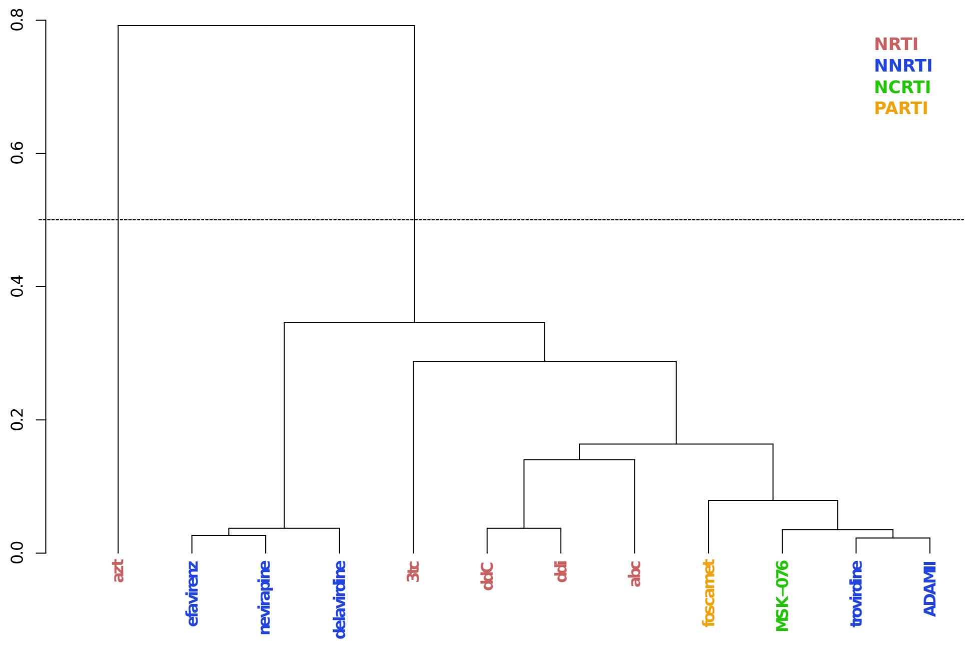
Table 7.5 reports AUC results for the different learning strategies for the V1 variant of the language bias. The problem of using a plain multitask approach is even more severe when considering single drugs from all



(a) fold 0



(b) fold 1



(c) fold 2

Figure 7.4: Hierarchical clustering dendrograms. Specific drugs belonging to the same class are coloured respectively in red for NRTI, blue for NNRTI, orange for PARTI, green for NCRTI.

classes, as the single task alternative is significantly better in six out of twelve cases and never significantly worse. Again, a per-task refinement of the multitask model allows to recover the single-task performance. However, only in one case (azt) the refinement is significantly better than the single-task alternative. This seems to indicate a poor overall contribution of task transfer, possibly because of the difference between target sites for the four inhibitor classes. Indeed, such inhibitors rely on quite different mechanisms. A further refinement level guided by the clustering procedure allows to increase to four the number of cases in which a hierarchical approach is significantly better than the single-task alternative. Furthermore, in one of these cases the approach is significantly better than the shallow hierarchical multitask model too. Note that in one case (foscarnet) the hierarchical clustering is actually significantly worse than the shallow approach. However, in this case a plain single task model is significantly better than both hierarchical versions[3]. This drug has very few examples (53), indicating a possible sub-optimality of the hierarchical versions when insufficient data are available.

### Discussion and Rule Interpretation

By mining the learnt rules some interesting examples of hierarchical concept learning can be found: the clause `mut(A,w,88,B)` learnt in the multitask setting is specialised at the first refinement stage into the clause

```
mut(A,w,88,B),mut(A,C,D,E),color(blue,C),color(green,E),
```

in which an additional rather high-level mutation description is included. In the second refinement stage, the per-drug one, the clause is further specialised depending on the specific drug under consideration. In the case

[3]Note that the multitask approach for foscarnet is not significantly better than the hierarchical ones even if its AUC value is the same as the single-task one. This is due to the fact that in the latter case predictions are highly related to the hierarchical ones, and the Hanley-McNeil test [68] considers such relatedness in computing the critical ratio.

| Drug Class | Drug | multitask | single task | hierarchical multitask | hierarchical clustering |
|---|---|---|---|---|---|
| NRTI | azt | 0.95 | 0.93 | 0.97●○ | 0.97●○ |
| | 3tc | 0.76 | 0.92○ | 0.91○ | 0.94●○◇ |
| | abc | 0.92 | 0.95○ | 0.96○ | 0.95○ |
| | ddC | 0.94 | 0.92 | 0.94 | 0.94● |
| | ddi | 0.95 | 0.94 | 0.94 | 0.95 |
| NNRTI | efavirenz | 0.93 | 0.94○ | 0.95○ | 0.96●○ |
| | nevirapine | 0.92 | 0.95○ | 0.96○ | 0.96○ |
| | delavirdine | 0.93 | 0.96○ | 0.97○ | 0.97○ |
| | ADAMII | 0.87 | 0.90 | 0.91 | 0.90 |
| | trovirdine | 0.91 | 0.96○ | 0.96○ | 0.97○ |
| NCRTI | MSK-076 | 0.99 | 0.98 | 0.97 | 0.99 |
| PARTI | foscarnet | 0.88 | 0.88◇⋆ | 0.86⋆ | 0.84 |

Table 7.5: Summary of the hierarchical clustering experiments (KTA scoring) with the V1 version of the language bias. AUC values are reported, together to the results of the statistical tests for the significance of AUC differences computed using the two-tailed Hanley-McNeil test [68] (p=0.05). A symbol after an AUC value indicates that the corresponding method is significantly better than: single task (●), multitask (○), hierarchical multitask (◇), hierarchical clustering (⋆). All other differences are not statistically significant.

of Lamivudine (3tc), for instance, the final clause is:

```
mut(A,w,88,B),mut(A,C,D,E),color(blue,C),color(green,E),mut(A,t,400,F)
```

that is, a specific position is selected in combination with the previously characterised mutations.

Another interesting example is given by the clause

```
mut(A,t,215,y),mut(A,g,196,B),
```

which is initially refined into `mut(A,t,215,y),mut(A,g,196,B),mut(A,t,357,C)`. When refining on a single task basis for the NRTI drug Abacavir (abc), for instance, the previously observed [141] correlation between mutations in positions 215 and 41 is recovered:

```
mut(A,t,215,y),mut(A,g,196,B),mut(A,t,357,C),mut(A,m,41,D),mut(A,i,293,E).
```

Correlation of mutations in position 41, 215 and 293 was also highlighted by the relational rule miner used in [120]. Here we are able to be a bit more specific and relate the correlation to resistance to a particular drug (abc).

Among the kFOIL generated rules many surveillance mutations [12] indicated for the resistance to NRTI and NNRTI can be found. Those mutations often appear in correlation with other mutations that potentially participate in conferring the resistance to a class of inhibitors or to a specific inhibitor. For instance, the mutations in position 184 that were shown to be involved in 3tc resistance appear jointly with a mutation in position 69. In abc resistance the same 184 mutated position appears with a mutation in position 10 in the generated model.

### 7.5.2 Protein Structure Classification

After being sequenced, a protein folds in the 3D space assuming a specific native conformation. Large regularities can be observed in these conformations, from the local arrangements into secondary structure units, alpha helices and beta strands, to their aggregation into domains. A number

| All-$\alpha$ | N | All-$\beta$ | N | $\alpha/\beta$ | N | $\alpha+\beta$ | N |
|---|---|---|---|---|---|---|---|
| DNA 3-helical | 30 | Ig beta-sandwich | 45 | $\beta/\alpha$ (TIM)-barrel | 55 | Ferredoxin-like | 26 |
| EF hand-like | 14 | Tryp ser proteases | 21 | Rossmann-fold | 21 | Zincin-like | 13 |
| Globin-like | 13 | OB-fold | 20 | P-loop | 14 | SH2-like | 13 |
| 4-Helical cytokines | 10 | SH3-like barrel | 16 | Periplasmic II | 13 | beta-Grasp | 12 |
| Lambda repressor | 10 | Lipocalins | 14 | $\alpha/\beta$-Hydrolases | 12 | Interleukin | 9 |

Table 7.6: Number of examples (N) in each of the folds in the SCOP dataset. Folds are grouped by fold classes: All-$\alpha$, All-$\beta$, $\alpha/\beta$ and $\alpha+\beta$.

of hierarchies of protein 3D structures have been created, based on evolutionary and/or structural considerations. A protein structure classification task in this setting consists of automatically assigning a protein structure to the correct class, relying on information like the arrangement of its secondary structure units.

SCOP [72] is a manually curated hierarchy based on both structural and evolutionary relationships between proteins. Turcotte et al. [137] extracted a dataset made of the five most populated folds of each of the four main classes (see Table 7.6). They learnt a set of rules characterising each of the folds with respect to the other folds of the same class, in a binary classification setting. The problem was later generalised [39] to full multiclass classification at the fold-class level. The following experiments consider this last setting.

#### Background Knowledge

The background knowledge encoded in [137] is used for representing the three-dimensional structure information of the proteins. Table 7.7 reports the background knowledge predicates divided into three classes: global knowledge, which encodes global characteristics of the protein domains, namely, the number of residues and the number and type of secondary structure elements; local knowledge, which encodes local information of a single protein secondary structure element (SSE); relational knowledge, introducing relationships between secondary structure elements and their properties.

| **Background Knowledge Predicates** | |
|---|---|
| *global background knowledge* | |
| `len_interval(Min,Domain,Max)` | indicates that the number of amino acids composing a Domain is between Min and Max |
| `nb_alpha_interval(Min,Domain,Max)` | indicates that the number of $\alpha$-helices composing a Domain is between Min and Max |
| `nb_beta_interval(Min,Domain,Max)` | indicates that the number of $\beta$-strands composing a Domain is between Min and Max |
| *local background knowledge* | |
| `unit_len(SSE,Value)` | indicates the length of an SSE as `very_low`, `lo`, `hi`, `very_high` |
| `unit_aveh(SSE,Value)` | indicates the average hydrophobicity of an SSE as `very_low`, `lo`, `hi`, `very_high` |
| `unit_hmom(SSE,Value)` | indicates the hydrophobic moment of an SSE as `very_low`, `lo`, `hi`, `very_high` |
| `has_pro(SSE)` | indicates whether an SSE contains a proline. |
| *relational background knowledge* | |
| `adjacent(Domain,SSE1,SSE2,N,TypeSSE1,TypeSSE2)` | indicates that the N-th (position along the chain) secondary structure element SSE1 of type TypeSSE1 is followed by SSE2 of type TypeSSE2. The type can be `h` or `e`. Helices and strands are numbered separately |
| `coil(SSE1,SSE2,Len)` | indicates that there are Len residues between two SSEs |

Table 7.7: Summary of the SCOP background knowledge.

### Hierarchical Multitask Experiments

Single task problems here consist of discriminating a fold against the other folds in the same fold class. A common multitask learning problem can be devised by jointly addressing all 20 single tasks. The two alternatives are compared with the hierarchical approach. Adhering to the experimental setting in [39], 5-fold cross validation procedures are performed and multiclass classification accuracies at the fold-class level are reported.

Experimental results are summarised in Table 7.8. Single task learn-

| Fold class | single task | multi task | hierarchical |
|---|---|---|---|
| all-$\alpha$ | 0.66○ | 0.45 | 0.69○● |
| all-$\beta$ | 0.78○ | 0.65 | 0.87○● |
| $\alpha/\beta$ | 0.55○ | 0.50 | 0.59○● |
| $\alpha+\beta$ | 0.59○ | 0.46 | 0.67○● |

Table 7.8: Summary of the hierarchical multitask experiments for the SCOP dataset. Multiclass accuracies at the fold-class level are reported, averaged over 5 folds. The results of a two-tailed paired t-test for the significance of accuracy differences are also shown (p=0.05). A symbol after an accuracy value indicates that the corresponding method is significantly better than single task (●) or multitask (○) respectively.

ing is always significantly better than the multitask approach. This is rather expected in this setting, as different fold classes have quite different structural characteristics. This is a clear example where plain multitask learning is badly harmful because of the limited relationship between tasks. On the other hand, the hierarchical approach is always significantly better than both multitask and single task alternatives. That is, it succeeds in collecting the useful information coming from loosely related tasks, to be effectively refined on a per-task basis. Indeed, only a fraction of the multitask clauses are actually retained, or further specialised, in the refined models. Note that a deeper hierarchical structure could be conceived by learning a common representation at the fold-class level, to be further refined. This is a special case in which the multitask problem is actually a multiclass one. While single tasks are definitely related in this case, there is no increase in the training set size as all examples appear in all tasks. We experienced a performance degradation when including this additional level of the hierarchy. The results are roughly comparable to those reported in [39]. Better results are achieved on the two most populated fold classes, and worse on the other two. However, a sound comparison cannot be conducted because of the different learning setting. The set of rules

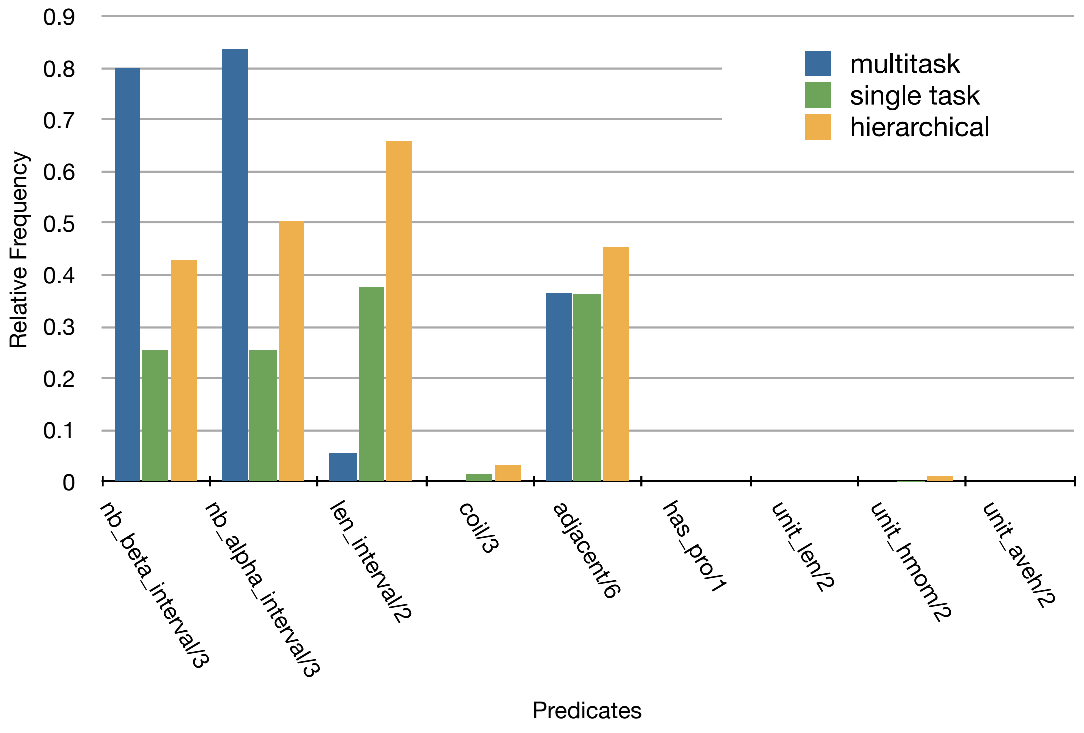


Figure 7.5: Relative frequency of the background knowledge predicates in the learnt models.

employed in [39] were learnt on the entire dataset, and only their weights were learnt and evaluated with a cross-validation procedure.

### Discussion and Rule Interpretation

Figure 7.5 highlights the different roles of global, relational and local background knowledge predicates (on the $x$ axis) in the learnt models. On the $y$ axis we report the relative frequency of occurrence of each predicate in the ten best clauses for the different settings (multitask, single task and hierarchical).

As expected the multitask learning produces models mainly including global background knowledge predicates, especially those characterising the number and type of secondary structure elements. Indeed these are the main features characterising the different fold classes. In single task learning models and refined models, relational background knowledge predicates gain more relevance. The results also confirm the observation in [38] that local information, related to the hydrophobicity (unit_aveh/2 and unit_hmom/2) and the presence of a proline in the SSE (has_pro/1), has a quite marginal role.

To give a more detailed idea of the learnt clauses interpretation, we report an example of hierarchical rule. At the root level, the multitask model includes the following clause:

```
nb_beta_interval(0,A,3),nb_alpha_interval(1,A,18).
```

The clause is refined at the single-task level both by adding global information on domain lengths and relational one for SSE pairs. Examples of the former type of refinement include the 4-Helical cytokines fold from the All-$\alpha$ class:

```
nb_beta_interval(0,A,3),nb_alpha_interval(1,A,18),len_interval(104,A,145).
```

and the Ferredoxin-like for the $\alpha + \beta$ one:

```
nb_beta_interval(0,A,3),nb_alpha_interval(1,A,18),len_interval(58,A,72).
```

Relational background knowledge is used with regard to $\beta$-strands in the Interleukin fold from the $\alpha + \beta$ class:

```
nb_beta_interval(0,A,3),nb_alpha_interval(1,A,18),adjacent(A,B,C,2,e,e).
```

and to $\alpha$-helices in the EF-hand like fold (All-$\alpha$).

```
nb_beta_interval(0,A,3),nb_alpha_interval(1,A,18),
                              nb_alpha_interval(3,A,5),adjacent(A,B,C,1,h,h).
```

Note that in this last clause the number of $\alpha$-helices is also restricted to a range from three to five. The domain should then contain two consecutive helices at the beginning of the polypeptide chain. Figure 7.6 shows an example of EF-hand like protein structure (PDB code 1CNP) that respects the above rule. The two consecutive $\alpha$-helices are highlighted in red.

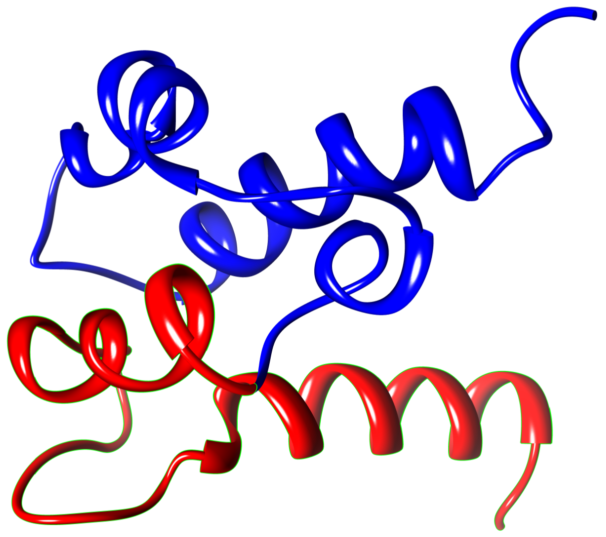


Figure 7.6: Example of EF-hand like protein structure (PDB code 1CNP). The first two consecutive $\alpha$-helices are highlighted in red.

# Chapter 8

# A Relational Learning Approach Toward Protein Engineering

The mining of relevant features from protein mutation data has its first aim in understanding the properties of functional sites, for instance, which residues are more likely to have a functional role. The same mined information can be used to engineer mutants of a protein with an improved activity on a certain substrate or resistance to a certain inhibitor.

Rational design is an engineering technique modifying existing proteins by site directed mutagenesis. It assumes the knowledge or intuition about the effects of specific mutations on the protein function. The process typically involves extensive trial-and-error experiments and is also used with the aim of improving the understanding mechanisms of a protein behaviour. In the case of viral proteins this is particularly important for taking the necessary counter-measures against the development of drug resistance.

In this chapter first steps are moved towards the use of a relational learning approach to protein engineering [160]. The focus is on learning relevant single mutations that can affect the behaviour of a protein. Predicting the effect of single mutations helps reducing the dimension of the search space for rational design.

In the proposed approach an ILP learner is trained to extract general relational rules describing mutations relevant to a certain behaviour (e.g. drug resistance of HIV). The learnt rules are then used to infer novel potentially relevant mutations. The approach is applied for learning relevant mutations of the HIV reverse transcriptase (RT). In the case of the HIV drug resistance, building artificial mutants that can resist to the inhibitor could help to early predict the virus evolution and thus design more effective drugs. HIV drug resistance mutation data are abundant and collected in well structured databases like the Los Alamos National Laboratories (LANL) HIV resistance database[1] and the Stanford HIV Drug Resistance Database [117].

Many machine learning methods have been applied in the past to mutation data for predicting single point mutations on protein stability changes [33] and the effect of mutations on the protein function [104] [26] or drug susceptibility [118]. At the author's knowledge this is the first approach proposal for learning relational features of mutations affecting a protein behaviour and use them for generating novel relevant mutations. Furthermore, even if the focus of the experimental evaluation is on single point mutations, the approach can be quite straightforwardly extended to multiple point mutations. Conversely, the up-mentioned predicting approaches would immediately blow up for the explosion in the number of candidate configurations to evaluate. The approach is tested on a dataset of HIV drug resistance mutations, and compared to a baseline random generator of mutations.

The promising preliminary results suggest that the proposed approach for learning mutations has a potential in guiding mutant engineering, as well as in predicting virus evolution in order to try and devise appropriate countermeasures. It is also worth noticing that it can be quite easily generalised to learning mutants characterised by more complex rules correlating

[1]http://www.hiv.lanl.gov/content/sequence/RESDB/

multiple mutations.

## 8.1 Algorithm Overview

The aim of the approach is to learn a first-order logic hypothesis for the target concept, i.e. mutation conferring resistance to a certain drug, and use it to infer novel mutations consistent with such hypothesis. For instance, a simple hypothesis like

`res_against(A,ncrti) ← mutation(A,C),close_to_site(C)`

would indicate that a mutation `C` in the proximity of a binding site confers to mutant `A` resistance against a `ncrti`. First-order clauses can thus be interpreted as relational features that characterise the target concept. The main advantage of using a logic-based approach with respect to other machine learning techniques is the expressivity and interpretability of the learnt models. Models can be readily interpreted by human experts and provide direct explanations for the predictions.

In the following the proposed approach is described. Starting from HIV RT mutation data a relational knowledge base is built. By using an ILP learner, mining of relevant relational features for modelling mutant resistance is performed. Then, on the basis of the learnt relational rules a set of candidate mutations satisfying them is generated.

### 8.1.1 Background Knowledge

A relational knowledge base is built for the domain at hand. Differently from the approach proposed in Chapter 7 here the focus is on single mutations rather than on the entire mutant. Future extension of the approach will generalise again to the whole mutant properties for capturing also relations among multiple mutations. Table 8.1 summarises the predicates included as a background knowledge. We represented the amino acids of the

| **Background Knowledge Predicates** | |
|---|---|
| aa(Pos,AA) | indicates a residue in the wild type sequence |
| mut_prop(MutationID,AA,Pos,AA1) | indicates a mutation: mutation identifier, position and amino acids involved, before and after the substitution |
| res_against(MutationID,Drug) | indicates whether a mutation is resistant to a certain drug |
| color(Color,AA) | indicates the type of a natural amino acid |
| same_type(R1,R2) | indicates whether two residues are of the same type |
| same_type_mut(MutationID, Pos) | indicates a mutation to a residue from the same type |
| different_type_mut(MutationID, Pos) | indicates a mutation changing the type of residue |
| close_to_site(Pos) | indicates whether a specific position is close to a binding or active site if any |
| location(L,Pos) | indicates in which fragment of the primary sequence the amino acid is located |
| catalytic_propensity(AA,CP) | indicates whether an amino acid has a high, medium or low catalytic propensity |
| mutated_residue_cp(Pos, CPold, CPnew) | indicates how, in a mutated position, the catalytic propensity has changed (e.g. from low to high) |

Table 8.1: Summary of the background knowledge facts and rules.

wild type with their positions in the primary sequence (aa/2) and the specific mutations characterising them (mut_prop/4). Target predicates were encoded as resistance of the mutation to a certain drug (res_against/2).

Additional background knowledge was included in order to highlight characteristics of residues and relationships between mutations. The following three predicates are already described in the previous chapter. Their descriptions are reported below again for convenience and readability:

color/2 indicates the type of the natural amino acids according to the colouring already described in the previous chapter (see Table 7.3). For example the magenta class includes basic amino acids as lysine and arginine while the blue class includes acidic amino acids as aspartic and glutamic acids.

same_type/2 indicates whether two residues belong to the same type, i.e. a change from one residue to the other conserves the type of the amino acid.

`same_type_mut/2` indicates that a residue substitution at a certain position does not modify the amino acid type with respect to the wild type. For example mutation d123e conserves the amino acid type while mutation d123a does not (i.e. `different_type_mut/2` holds for it).

Other background knowledge facts and rules were added in order to express structural relations along the primary sequence and catalytic propensity of the involved residues:

`close_to_site/1` indicates whether a specific position is distant less than 5 positions from a residue belonging to a binding or active site. In this specific case, the background theory incorporates knowledge about a metal binding site and a heterodimerisation site.

`location/2` indicates in which fragment of the primary sequence the amino acid is located. Locations are numbered from 0 by dividing the sequence into a certain number of fragments.

`catalytic_propensity/2` indicates whether an amino acid has a high, medium or low catalytic propensity according to [7].

`mutated_residue_cp/3` indicates how, in a mutated position, the catalytic propensity has changed (e.g. from low to high).

### 8.1.2 The Approach

The proposed approach is sketched in Figure 8.1. The first step is the learning phase, in which an ILP learner is fed with a logical representation of the data $\mathcal{D}$ (the mutations experimentally observed to confer resistance to a certain inhibitor) and of the domain knowledge $\mathcal{B}$ we want to incorporate. After a training stage, it returns a first-order logical hypothesis $H$ for the concept of mutation conferring resistance to a certain drug.

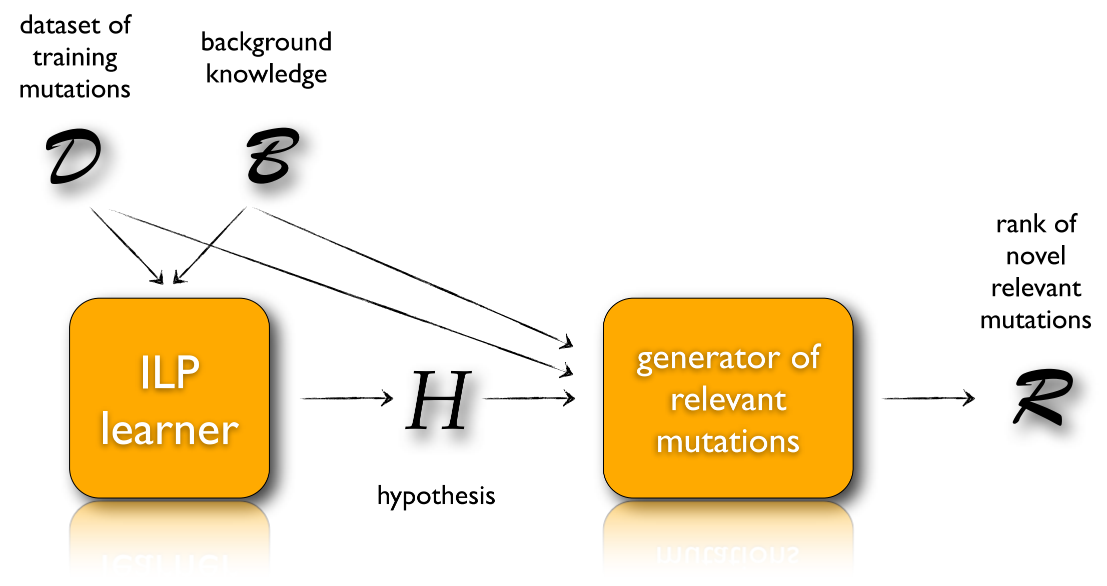

dataset of training mutations
background knowledge
$\mathcal{D}$
$\mathcal{B}$
rank of novel relevant mutations
ILP learner
$H$
hypothesis
generator of relevant mutations
$\mathcal{R}$

Figure 8.1: Schema of the mutation engineering algorithm.

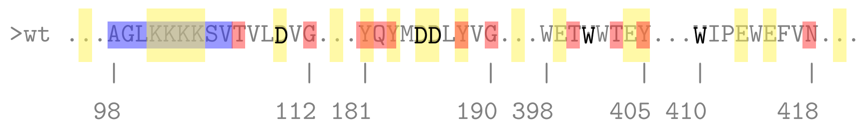
The hypothesis is derived using the Aleph (A Learning Engine for Proposing Hypotheses) ILP system[2]. Aleph allows also to learn from positive example only. This is the most suitable approach in this case as the positive examples are the mutations experimentally proved to confer resistance to a drug but no safe claim can be made on the other mutations if there is no sufficient evidence due for example to the lack of an exhaustive set of laboratory experiments.

Aleph incrementally builds a hypothesis covering all positive examples guided by a Bayesian evaluation function, described in [98], scoring candidate solutions according to an estimate of the Bayes' posterior probability that allows to tradeoff hypothesis size and generality.

In Figure 8.2 an example of learnt hypothesis covering a set of examples is shown. The learnt hypothesis models the ability of a mutation to confer the resistance to NCRTIs and is composed of three first-order clauses, each

[2]http://www.comlab.ox.ac.uk/activities/machinelearning/Aleph/aleph.html

one covering different sets of mutations of the wild type as highlighted in colours: blue for the first clause, yellow for the second and red for the third one. Some mutations are covered by more than one clause as shown by the colour overlaps.


>wt ...AGLKKKKSVTVLDVG...YQYMDDLYVG...WETWWTEY...WIPEWEFVN...
98
112
181
190
398
405
410
418

🟦 `mut_prop(A,B,C,D),location(11.0,C)`
🟨 `mut_prop(A,B,C,D),mutated_residue_cp(C,high,small)`
🟥 `mut_prop(A,B,C,D),color(green,B),close_to_site(C)`

Figure 8.2: Example of learnt hypothesis for the NCRTI task with highlighted mutations covered by the hypothesis clauses.

The built background knowledge incorporates information about the RT metal binding site, which is composed of the aspartic acids D110, D185 and D186 and, about the heterodimerisation site composed of W401 and W414 (in bold in Figure 8.2).

The second step is the generative phase, in which the learnt hypothesis is employed to find novel mutations that can confer drug resistance to an RT mutant. A set of candidate mutations can be generated by using the Prolog inference engine starting from the rules in the learnt model. The rules are actually constraints on the characteristics that a mutation of the wild type should have in order to confer resistance to a certain inhibitor.

Algorithm 8.1 details the mutation generation procedure. We here assume, for simplicity, to have a model $H$ for a single drug class. The procedure works by querying the Prolog inference engine for all possible variable assignments that satisfy the hypothesis clauses. In order to generate novel mutations, the `mut_prop/4` predicate is modified here in order to return all legal mutations instead of only those appearing in the training instances.

The set of mutations identified by the variable assignments and not present in the training set is ranked according to a scoring function $S_M$ before being returned by the algorithm. Here $S_M$ is defined as the number of clauses in $H$ that a candidate mutation $m$ satisfies.

**Algorithm 8.1** Algorithm for novel relevant mutations discovery.

```
1: input: dataset of training mutations $\mathcal{D}$, background knowledge $\mathcal{B}$, learnt model $H$
2: output: rank of the most relevant mutations $\mathcal{R}$
3: procedure GenerateMutations($\mathcal{D}, \mathcal{B}, H$)
4:     Initialise $\mathcal{D}_{\mathcal{M}} \leftarrow \emptyset$
5:     $A \leftarrow$ find all assignments $a$ that satisfy at least one clause $c_i \in H$
6:     for $a \in A$ do
7:         $m \leftarrow$ mutation corresponding to the assignments $a \in A$
8:         $score \leftarrow S_M(m)$                ▷ number of clauses $c_i$ satisfied by $a$
9:         if not $m \in \mathcal{D}$ then          ▷ discard mutations observed in the training set
10:            $\mathcal{D}_{\mathcal{M}} \leftarrow \mathcal{D}_{\mathcal{M}} \cup \{(m, score)\}$
11:        end if
12:    end for
13:    $\mathcal{R} \leftarrow$ RankMutations($\mathcal{D}_{\mathcal{M}}, \mathcal{B}, H$)     ▷ rank relevant mutations
14:    return $\mathcal{R}$
15: end procedure
```

Referring to the model in Figure 8.2 some generated candidate mutations with highest score are: 101P, 102A, 103F, 103I, 104M, 181I, 181V, 183M, 188L, 188F where for example the notation 101P indicates a change of the wild type amino acid, located in position 101, into a proline (P). This list includes also known HIV RT surveillance mutations [12].

## 8.2 Experimental Evaluation

Experimental results follow with a comparison to a baseline random generator of mutations.

### 8.2.1 Dataset

The approach is applied to the same dataset of mutations already used in [120] for extracting relational rules among mutations. The dataset is derived from the Los Alamos National Laboratories (LANL) HIV resistance database[3] and reports mutations of the HIV RT. Richter et al. [120] formulated the learning problem as a mining task and applied a relational association rule miner to derive rules relating different mutations and their resistance properties.

This dataset is the same used in Chapter 7 for learning a relational model of mutant resistance with the hierarchical kFOIL algorithm. Here the dataset is used for inferring rules that can characterise a single mutation as resistant to a certain class of RT inhibitors. Those drug classes include: a) Nucleoside RT Inhibitors (NRTI); b) NonNucleoside RT Inhibitors (NNRTI); c) NonCompetitive RT inhibitors (NCRTI); d) Pyrophosphate Analogue RT Inhibitors (PARTI). The final dataset is composed of 164 mutations labeled as resistant over a set of 581 observed mutations (extracted from 2339 mutants). Among the 164 mutations, 95 are labeled as resistant to NRTI, 56 to NNRTI, 5 to NCRTI and 8 to PARTI.

### 8.2.2 Performance Evaluation

The dataset of mutations is divided into a training and a test set (70/30) in a stratified way, which means by preserving, both in the train and test set, the proportion of examples belonging to one of the four drug classes. The resulting training set is composed of a total of 116 mutations while the test set is composed of 48 mutations.

The ILP learner is trained on the training set and the set of mutations generated using the learnt model is evaluated on the test set. The evalua-

[3]http://www.hiv.lanl.gov/content/sequence/RESDB/

| | *Mean recall % on 30 splits* | | | |
|---|---|---|---|---|
| | **Algorithm** | **Random Generator** | *Mean n. generated muts* | *n. test muts* |
| NNRTI | 86 ● | 58 | 5201 | 17 |
| NRTI | 55 ● | 46 | 5548 | 28 |
| NCRTI | 16 | 8 | 1042 | 1 |
| PARTI | 49 | 39 | 3425 | 2 |

Table 8.2: Statistical comparisons of the performance of the proposed algorithm with an algorithm generating mutations at random. The average recall has been computed for each one of the learning tasks over the 30 splits by averaging recall over 30 repeated runs of the two algorithms. Results of a paired Wilcoxon test ($\alpha = 0.05$) on the statistical significance of the performance differences are also reported. A black bullet indicates a statistical significant improvement of the algorithm over a random generator.

tion procedure takes the generated mutations and computes its enrichment in test mutations by counting how many generated mutations are actually observed in at least one test example. The recall of the approach is compared with the recall of an algorithm that chooses at random a set (of the same cardinality) of possible mutations among all legal ones. The *Recall* here corresponds to the ratio between the number of generated test set mutations and the total number of test set mutations. Nothing can be said on mutations that are not in the test set because those mutations have not been assayed in "wet lab" experiments for their ability to confer drug resistance. Results averaged on 30 random splits of the dataset are reported in Table 8.2.

On each split, 30 runs of the algorithm and of the random generation algorithm are performed, in each one of the different learning tasks (NNRTI, NRTI, NCRTI and PARTI). Columns 3 and 4 also report for each task the mean number of generated mutations over the 30 splits and the number of test set mutations for reference. The statistical significance of the performance differences between the two algorithms is evaluated by paired Wilcoxon tests on the averaged recall reported on each split. A confidence level $\alpha$ of 0.05 is employed.

The improvement of the algorithm with respect to the random generation of mutations is statistically significant on the NRTI and NNRTI tasks, which are the tasks on which we can learn the hypothesis from a largest set of training examples.

Figure 8.3 shows the trend of the mean recall over all splits when cutting the number of generated mutations (from 1 generated mutation to more than 8000). The advantage of the approach remains quite stable when reducing the set of candidates, producing almost nine times more test mutations than random in the 100 highest scoring ones for NNRTI (Figure 8.4 shows the trend of the recall curves of the plot in Figure 8.3(a)(a) for the highest ranked 150 mutations).

## 8.3 Discussion

Finally a model is learnt on the whole set of training mutations in order to generate a single set of mutations for further inspection. Below five examples of novel mutations are reported, which have the highest rank for each one of the tasks:

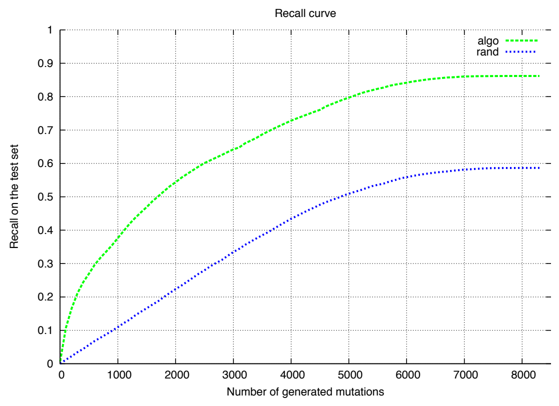
**NNRTI** 90I 98I 103I 106P 179I

**NRTI** 60A 153M 212L 229F 239I

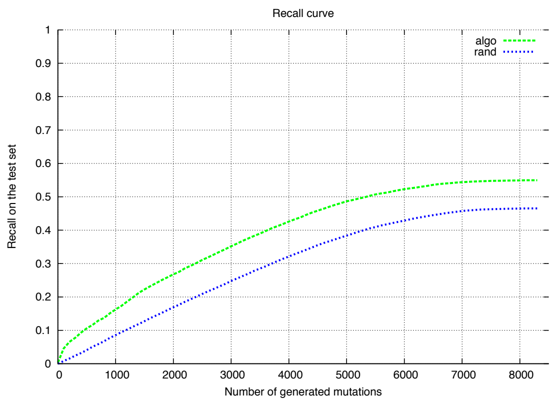
**NCRTI** 183V 183L 188V 188F 188I

**PARTI** 84R 86E 88Y 88V 89N

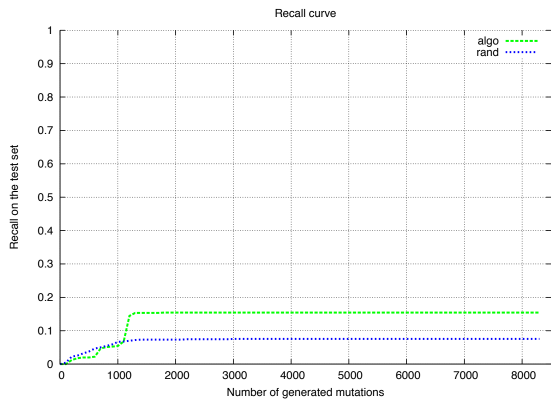
In [51], the authors found a set of novel mutations conferring resistance to efavirenz and nevirapine, which are NNRTIs. The proposed mutation generation algorithm partially confirmed their findings. Mutation 90I was ranked high (the model contains five clauses and all of them are satisfied,



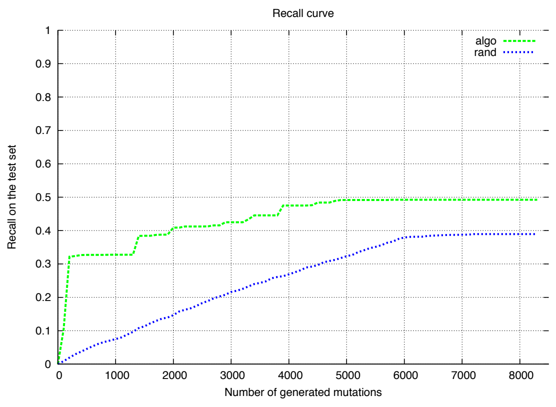
(a) NNRTI



(b) NRTI



(c) NCRTI



(d) PARTI

Figure 8.3: Mean recall by varying the number of generated mutations.

5/5), mutation 101H was generated with a rank of 3/5, mutations 196R and 138Q with rank 1/5, while mutation 28K was not generated at all by the proposed system as a candidate for conferring resistance to NNRTI.

### Example of learnt hypothesis

An example of learnt hypothesis is reported in Table 8.3.

For instance, according to the model, among the features a mutation should have for conferring resistance to a NNRTI, there is the change of a basic (magenta) residue of the wild type, e.g. lysine or arginine, into a

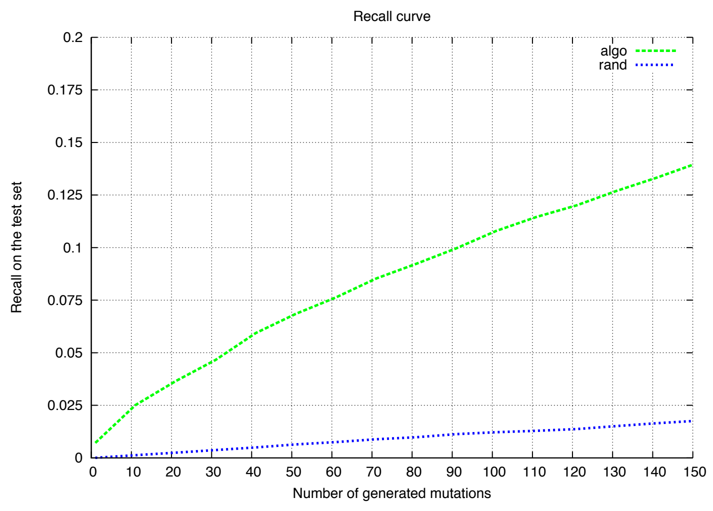


Figure 8.4: Detail of the mean recall curves of the NNRTI task for a number of generated mutations below 150.

residue with completely different phisico-chemical characteristics (rule 16).

Another example for the resistance to NNRTI is that a non conserved mutation is present in positions between 98 and 106 of the wild type sequence (rule 8).

## 8.4 Future Work

Albeit preliminary, the results suggest that the proposed approach for learning mutations has a potential in guiding mutant engineering, as well as in predicting virus evolution in order to try and devise appropriate countermeasures. A more detailed background knowledge, possibly including 3D information whenever available, is necessary in order to further focus the set of generated mutations, and possibly post-processing stages involving mutant evaluation by statistical machine learning approaches [33]. In the next future, plans are to generalise the proposed approach to jointly generate sets of related mutations shifting the focus from single point mutations to entire mutants.

```
1-res_against(A,nrti) ← mut_prop(A,B,C,D),color(red,D)
2-res_against(A,nrti) ← mut_prop(A,B,C,D),color(red,D),color(red,B)
3-res_against(A,nrti) ← mut_prop(A,B,C,D),location(7.0,C),mutated_residue_cp(C,medium,medium)
4-res_against(A,nrti) ← mut_prop(A,B,C,D),location(7.0,C)
5-res_against(A,nrti) ← same_type_mut(A,B)
6-res_against(A,nrti) ← mut_prop(A,B,C,D),mutated_residue_cp(C,medium,high)
7-res_against(A,nrti) ← mut_prop(A,B,C,D),mutated_residue_cp(C,medium,small)
8-res_against(A,nnrti) ← different_type_mut(A,B),location(11.0,B)
9-res_against(A,nnrti) ← mut_prop(A,B,C,D),mutated_residue_cp(C,small,small)
10-res_against(A,nnrti) ← same_type_mut(A,B)
11-res_against(A,nnrti) ← mut_prop(A,B,C,D),aminoacid(B,v)
12-res_against(A,nnrti) ← mut_prop(A,B,C,D),location(15.0,C)
13-res_against(A,nnrti) ← mut_prop(A,B,C,D),aminoacid(D,i)
14-res_against(A,nnrti) ← mut_prop(A,B,C,D),location(11.0,C)
15-res_against(A,nnrti) ← mut_prop(A,B,C,D),color(red,D)
16-res_against(A,nnrti) ← mut_prop(A,B,C,D),color(magenta,B),different_type_mut(A,C)
17-res_against(A,nnrti) ← mut_prop(A,B,C,D),location(21.0,C)
18-res_against(A,ncrti) ← mut_prop(A,B,C,D),location(11.0,C)
19-res_against(A,ncrti) ← mut_prop(A,B,C,D),mutated_residue_cp(C,high,small)
20-res_against(A,ncrti) ← mut_prop(A,B,C,D),color(green,B),close_to_site(C)
21-res_against(A,parti) ← mut_prop(A,B,C,D),location(9.0,C)
```

Table 8.3: Example of learnt model.

As already discussed in Chapter 7, multiple mutations are often required in order to affect a protein function. The debated neutral network theory claims that neutral mutations are required as intermediate steps to effective ones.

The prediction of single point mutations does not consider the joint effect of multiple mutations. Trying all the possible combination of mutations is computationally unfeasible and anyhow there are not enough data for applying a similar approach.

An approach like the one proposed in Chapter 7 that learns a model for characterising a mutant as resistant to a certain inhibitor, can be used for generating novel protein mutants that are most probable to develop resistance against the same inhibitor. In this way, a full protein engineering approach can be realised. In the specific case of HIV, engineering such novel mutants can be useful for anticipating the virus evolution toward the resistance to a drug and taking measures of prevention.

```
>m542  PISPIET   FAIKKKSSS   PLDKDFRKY   ELREHLLKWGFY   EIQKQGPGQWT   IVGAETF
>wt    PISPIET...FAIKKKDST...PLDEDFRKY...ELRQHLLRWGFT...EIQKQGQGQWT...IVGAETF
>m2012 PISPIET   FAIKKKDST   PLDESFRKY   KLREHLLRWGFT   EVQKQGPDQWT   IPGAETY
       *******   ******.*:   ***:.****   :**:***:***    *:**** .***   * ****:
                       -         -          -                 -
                       |         |          |                 |
                      67        123        207               334

mut(A,B,C,p),pos(C,334),correlated_mut(A,D,E),pos(D,207),residue_type(A,E,blue).

                      67        122                    328
                       |        |                       |
                       -        -                       -
       PISPIET...FAIKKKDST...PLDNDFRKY...ELREHLLRWGFT...EIQKQGPGQWT...IVGAETF
       PISPIET...FAIKKKDST...PLDEDFRKY...ELRDHLLRWGFT...QIQKQGPGQWT...IVGAETF
       PISPIET...FAIKKKSST...PLDEDFRKY...ELRDHLLRWGFT...EIQKQGPGQWT...IVGAETF

>m2006 PMSPIET   FAIKKKDST   PLHEDFRKY   ELREHLLKWGLT   EVQKQGPDQWT   IAGAETY
>wt    PISPIET...FAIKKKDST...PLDEDFRKY...ELRQHLLRWGFT...EIQKQGQGQWT...IVGAETF
>m1288 PISPIDT   FAIKKKNSD   PLDESFRKY   ELREHLLKWGFF   EIQKQGPGQWT   IPGAETY
       *:***:*   ******:*    ** *.****   ***:***:**:    *:**** .***   * ****:
                       -       -            -                 -
                       |       |            |                 |
                      67      121          207               334
```

Figure 8.5: Example of application of the approach for mutant engineering.

In Figure 8.5 is shown an example of hypothetical rule learnt from RT (wt) mutants (m542, m2012, m2006, m1288) and 3 examples of novel mutants that can be generated starting from the learnt rule.

# Chapter 9

# Conclusions

In the present thesis the problem of characterising an enzyme function has been tackled from two different perspectives: the identification of functional residues and the sites they belong to, based on the target protein structure information, and the mining of functional properties from the protein mutation data.

Statistical and relational machine learning techniques are investigated for helping in the enzyme function characterisation. Statistical learning techniques are chosen for their sound foundation in learning theory and their efficiency and proved empirical effectiveness when learning from the experience and generalising the prediction to novel cases. Relational learning techniques are appealing for their expressivity and their ability to provide human understandable models of the learnt concept. Statistical relational learning techniques combine these complementary advantages. Therefore, in this thesis hybrid approaches are also investigated.

First explorations of kernel-based algorithms for classification led to the development of a machine learning system for the prediction of functional residues and lately for identifying the active sites they belong to. The system is based on a support vector classifier and was shown to be more accurate than other state-of-the-art approaches.

The improvement over these approaches on a number of benchmark datasets and its statistical significance were systematically assessed. Fruitful results were obtained both by predicting functional residues from the protein primary sequence alone and by incorporating information about the protein 3D structure. In the last case the improvements over previous state-of-the-art approaches were achieved thanks to an accurate modelling and engineering of features of the spherical region surrounding the candidate residue, highlighting the important role played by structural information in this learning task. Information related to bounded heterogen molecules was shown to play a key role in identifying functional residues with low catalytic propensities. Future investigations will be directed to the the joint prediction of binding and catalytic sites in a fully collective approach.

Further improvements of the sequence and structure-based predictors were obtained by collectively learning functional residues belonging to an active site in a a distance-based supervised clustering approach based on the formalisation of the problem as a combinatorial search of maximum-weight cliques in protein weighted graph structures. An efficient optimisation algorithm based on tabu search has been proposed and applied for finding maximum-weight cliques of putative functional residues showing that the prediction accuracy can be further improved. The proposed algorithm naturally handles the lack of knowledge in the number of clusters, partial clusterings with many outliers and overlapping clusters. It is worth to notice that the maximum-weight clique clustering algorithm is however independent of the supervised learning stage, and can be easily integrated in more complex supervised clustering approaches such as the structured-output formulation proposed in [57].

The successful application of the algorithm also to the sequence-based prediction of metal binding sites suggests to jointly address the two related problems of active site and metal binding site detection. A future extension of the algorithm could make it able to return a structured set of solutions, such as metal binding sites as parts of wider functional sites, or as “cocatalytic” sites, which is a quite common situation in enzymes.

A web-server based on the proposed functional residue sequence- and structure-based predictors has been developed and made publicly available as a useful tool at the biologist fingertips.

The sequence-based functional residue predictor proposed in this thesis was shown to be useful also in the context of the analysis of mutants of the same protein and their activity on groups of related substrates. However, the work suggested the need for a more sophisticated approach: (a) a multitask learning setting for predicting the activity of mutants on different but related substrates; (b) a method which was eventually able to provide explanations for an improved or reduced activity and some insights on the properties of the wild type protein functional site.

Those considerations led to the investigation of logic-based learning algorithms for inferring rules from protein mutation data. A hybrid learner, namely the kFOIL algorithm, which offers at the same time the advantages of statistical learning and of induction on logic-based hypotheses, was used for learning from mutation data in a multitask learning setting. A hierarchical extension of the multitask kFOIL algorithm was proposed allowing to incrementally refine models common to groups of related tasks, at each level of the hierarchy, until a refinement on a per-task basis at the hierarchy leaves. The performance of the method was tested on a large dataset of HIV mutants showing resistance to different classes of inhibitors. Accuracy improvements were obtained with respect to the standard single task learning or to the more general multitask learning. If the task hierarchical structure is unknown, it can be inferred by clustering tasks based on the predictions in the multitask setting. Indeed the approach showed to be able to recover a grouping of inhibitors in membership classes that was also useful for highlighting characteristics of the underlying dataset. A major advantage of the adopted strategy is the ability to provide explanations for the learnt models which are themselves hierarchical: a subset of relational features relevant to all tasks can be identified together with more specific task-dependent ones.

The proposed mining technique also has highlighted a potential in guiding mutant engineering. The learnt logical rules are useful for highlighting characteristics of protein functional sites and, as relevant mined features, they can be further exploited for building artificial instances satisfying the learnt concept. Building artificial mutants with an improved activity on a drug has important applications in biotechnology and pharmacology. As an example, engineered mutants showing resistance to a certain inhibitor can be useful for predicting and anticipating a protein response to a certain drug and taking the necessary countermeasures for the prevention or the infection treatment. At this aim, a simple relational learning approach toward protein engineering has been proposed. The first step is the mining of relevant relational features modelling a mutation conferring mutant resistance to drugs, by using an inductive logic programming learner. Then, based on the learnt relational rules, the second step is to generate a set of candidate mutations satisfying them. Albeit preliminary, experimental results suggest that this approach for learning mutations has a potential in guiding mutant engineering, as well as in predicting virus evolution in order to try and devise appropriate countermeasures. A more detailed background knowledge, possibly including 3D information whenever available, is necessary in order to further focus the set of generated mutations, and possibly post-processing stages involving mutant evaluation by statistical machine learning approaches. Future research plans are to generalise the proposed approach to jointly generate sets of related mutations shifting the focus from single point mutations to entire mutants.

It is worth to notice that all of the proposed approaches — applied in this thesis with the goal of functionally characterising enzymes — can be easily generalised to other kind of proteins. Especially the presented relational learning approaches for mining features from mutation data and engineering mutants are by definition generally applicable to mutations and mutants of any protein sequence, independently of its specific function inside the cell. The background knowledge in those cases can be slightly adapted and enriched based on the specific domain. Moreover these approaches can be easily adapted for solving other protein bioinformatics problems. Some example are already given in this thesis. For instance, the supervised clustering approach was also successfully applied to the detection of metal binding sites. The hierarchical multitask kFOIL was shown useful also for the classification of protein structures into folds. But many other examples can be made. For instance, the same approach described in Part I. can be easily adapted for predicting protein-protein interaction sites and their structural interfaces.

# Bibliography

[1] S. F. Altschul, T. L. Madden, A. A. Schäffer, J. Zhang, Z. Zhang, W. Miller, and D. J. Lipman. Gapped BLAST and PSI-BLAST: a new generation of protein database search programs. *Nucleic Acids Research*, 25(17):3389–3402, Sept. 1997.

[2] G. Amitai, A. Shemesh, E. Sitbon, M. Shklar, D. Netanely, I. Venger, and S. Pietrokovski. Network analysis of protein structures identifies functional residues. *Journal of Molecular Biology*, 344(4):1135–1146, 2004.

[3] M. Babor, S. Gerzon, B. Raveh, V. Sobolev, and M. Edelman. Prediction of transition metal-binding sites from apo protein structures. *Proteins*, 70(1):208–217, 2008.

[4] G. H. BakIr, T. Hofmann, B. Schölkopf, A. J. Smola, B. Taskar, and S. V. Vishwanathan. *Predicting Structured Data.* Neural information processing. MIT Press, Cambridge, MA, USA, 09 2007.

[5] B. Bakker and T. Heskes. Task clustering and gating for Bayesian multitask learning. *Journal of Machine Learning Research*, 4:83–99, 2003.

[6] D. Ballard and C. Brown. *Computer vision.* Prentice-Hall, Englewood Cliffs, 1982.

[7] G. J. Bartlett, C. T. Porter, N. Borkakoti, and J. M. Thornton. Analysis of catalytic residues in enzyme active sites. *Journal of Molecular Biology*, 324(1):105–121, 2002.

[8] S. Basu. *Semi-supervised clustering: probabilistic models, algorithms and experiments*. PhD thesis, University of Texas at Austin, 2005.

[9] R. Battiti and F. Mascia. Reactive and dynamic local search for max-clique: engineering effective building blocks. *Computers Operations Research*, 37(3):534–542, Mar. 2010.

[10] R. Battiti and M. Protasi. Reactive Local Search for the Maximum Clique Problem. *Algorithmica*, 29(4):610–637, 2001.

[11] J. Baxter. A Bayesian/information theoretic model of learning to learn via multiple task sampling. *Machine Learning*, pages 7–39, 1997.

[12] D. E. Bennett, R. J. Camacho, D. Otelea, D. R. Kuritzkes, H. Fleury, M. Kiuchi, W. Heneine, R. Kantor, M. R. Jordan, J. M. Schapiro, A.-M. Vandamme, P. Sandstrom, C. a. B. Boucher, D. van de Vijver, S.-Y. Rhee, T. F. Liu, D. Pillay, and R. W. Shafer. Drug resistance mutations for surveillance of transmitted HIV-1 drug-resistance: 2009 update. *PloS one*, 4(3):e4724, 2009.

[13] M. M. Benning, T. Haller, J. A. Gerlt, and H. M. Holden. New reactions in the crotonase superfamily: structure of methylmalonyl CoA decarboxylase from Escherichia coli. *Biochemistry*, 39(16):4630–4639, Apr. 2000.

[14] H. M. Berman, J. Westbrook, Z. Feng, G. Gilliland, T. N. Bhat, H. Weissig, I. N. Shindyalov, and P. E. Bourne. The Protein Data Bank. *Nucleic Acids Research*, 28(1):235–242, 2000.

[15] J. S. Bernardes, J. H. Fernandez, and A. T. R. Vasconcelos. Structural descriptor database: a new tool for sequence-based functional site prediction. *BMC Bioinformatics*, 9:492+, 2008.

[16] I. Bertini, A. Sigel, and H. Sigel, editors. *Handbook on Metalloproteins*. Marcel Dekker, New York, 1 edition, 2001.

[17] T. A. Binkowski, S. Naghibzadeh, and J. Liang. CASTp: computed atlas of surface topography of proteins. *Nucleic Acids Research*, 31(13):3352–3355, 2003.

[18] H. Blockeel, L. De Raedt, and J. Ramon. Top-down induction of clustering trees. In *Proceeding of the 15th International Conference on Machine Learning (ICML)*, Madison, Wisconsin, USA, 1998.

[19] H. Blockeel, S. Dzeroski, B. Kompare, S. Kramer, B. Pfahringer, and W. V. Laer. Experiments in predicting biodegradability. *Applied Artificial Intelligence*, 18(2):157–181, 2004.

[20] J. R. Bock and D. A. Gough. Predicting protein-protein interactions from primary structure. *Bionformatics*, 17(5):455–460, January 2001.

[21] K. Borgwardt. *Graph-based Functional Classification of Proteins using Kernel Methods*. Ludwig Maximilians University of Monaco, Dec. 2004.

[22] K. M. Borgwardt, C. S. Ong, S. Schönauer, S. V. N. Vishwanathan, A. J. Smola, and H.-P. Kriegel. Protein function prediction via graph kernels. *Bioinformatics*, 21:i47–i56, 2005.

[23] B. E. Boser, I. Guyon, and V. Vapnik. A training algorithm for optimal margin classifiers. In *Computational Learning Theory*, pages 144–152, 1992.

[24] S. Brenner. A tour of structural genomics. *Nature Reviews Genetics*, 2(10):801–809, 2001.

[25] S. E. Brenner, C. Chothia, T. J. Hubbard, and a. G. Murzin. Understanding protein structure: using SCOP for fold interpretation. *Methods in Enzymology*, 266(1995):635–43, Jan. 1996.

[26] Y. Bromberg and B. Rost. SNAP: predict effect of non-synonymous polymorphisms on function. *Nucleic Acids Research*, 35(11):3823–3835, Jan. 2007.

[27] S. K. Burley, S. C. Almo, J. B. Bonanno, M. Capel, M. R. Chance, T. Gaasterland, D. Lin, A. Sali, F. W. Studier, and S. Swaminathan. Structural genomics: beyond the human genome project. *Nature Genetics*, 23(2):151–157, Oct. 1999.

[28] C. Camacho, G. Coulouris, V. Avagyan, N. Ma, J. Papadopoulos, K. Bealer, and T. L. Madden. BLAST+: architecture and applications. *BMC Bioinformatics*, 10(1):421, 2009.

[29] E. Camon, D. Barrell, V. Lee, E. Dimmer, and R. Apweiler. The Gene Ontology Annotation (GOA) Database - an integrated resource of GO annotations to the UniProt Knowledgebase. *In Silico Biology*, 4(1):5–6, 2004.

[30] Z. Cao, L. Han, C. Zheng, Z. Ji, X. Chen, H. Lin, and Y. Chen. Computer prediction of drug resistance mutations in proteins. *Drug Discovery Today*, 10(7):521–529, Apr. 2005.

[31] J. A. Capra and M. Singh. Predicting functionally important residues from sequence conservation. *Bioinformatics*, 23(15):1875–1882, May 2007.

[32] E. Capriotti, P. Fariselli, R. Calabrese, and R. Casadio. Predicting protein stability changes from sequences using support vector machines. *Bioinformatics (Oxford, England)*, 21 Suppl 2:ii54–8, 2005.

[33] E. Capriotti, P. Fariselli, I. Rossi, and R. Casadio. A three-state prediction of single point mutations on protein stability changes. *BMC Bioinformatics*, 9 Suppl 2:S6, 2008.

[34] R. Caruana. Multitask learning. *Machine Learning*, 28(1):41–75, 1997.

[35] A. Ceroni, F. Costa, and P. Frasconi. Classification of small molecules by two- and three-dimensional decomposition kernels. *Bioinformatics*, 23(16):2038–2045, Aug. 2007.

[36] M. Chang, D. Goldwasser, D. Roth, and V. Srikumar. Structured output learning with indirect supervision. In *Proceedings of the International Conference on Machine Learning (ICML)*, 2010.

[37] E. Chea and D. R. Livesay. How accurate and statistically robust are catalytic site predictions based on closeness centrality? *BMC Bioinformatics*, 8:153+, May 2007.

[38] J. Chen, L. Kelley, S. Muggleton, and M. Sternberg. Multi-class prediction using stochastic logic programs. *Inductive Logic Programming*, pages 109–124, 2007.

[39] J. Chen, L. Kelley, S. Muggleton, and M. Sternberg. Protein fold discovery using stochastic logic programs. In L. De Raedt, P. Frasconi, K. Kersting, and S. Muggleton, editors, *Probabilistic Inductive Logic Programming*, pages 244–262. Springer-Verlag, Berlin, Heidelberg, 2008.

[40] J. Cheng, A. Randall, and P. Baldi. Prediction of protein stability changes for single-site mutations using support vector machines. *Proteins: Structure, Function, and Bioinformatics*, 62(4):1125–1132, 2006.

[41] M. Collins and N. Duffy. New ranking algorithms for parsing and tagging: Kernels over discrete structures, and the voted perceptron. In *Proceedings of ACL02*, 2002.

[42] C. Cortes and V. Vapnik. Support-vector networks. *Machine Learning*, pages 273–297, 1995.

[43] N. Cristianini and J. Shawe-Taylor. *An Introduction to Support Vector Machines and Other Kernel-based Learning Methods.* Cambridge University Press, Mar. 2000.

[44] N. Cristianini, J. Shawe-Taylor, A. Elisseef, and J. Kandola. On kernel-target alignment. In *Proceedings of Neural Information Processing Systems (NIPS) 14*, 2001.

[45] A. S. D'Abusco, S. Ammendola, R. Scandurra, and L. Politi. Molecular and biochemical characterization of the recombinant amidase from hyperthermophilic archaeon Sulfolobus solfataricus. *Extremophiles*, 5:183–192, June 2001.

[46] P. Datta and D. F. Kibler. Concept sharing: A means to improve multi-concept learning. In *Proceedings of the 10th International Conference on Machine Learning (ICML), Amherst, MA, USA*, 1993.

[47] H. Daumé III. Bayesian multitask learning with latent hierarchies. In *Proceedings of the 25th Annual Conference on Uncertainty in Artificial Intelligence (UAI-09)*, pages 135–142, Corvallis, Oregon, 2009. AUAI Press.

[48] J. Davis and M. Goadrich. The relationship between Precision-Recall and ROC curves. In *Proceedings of the 23rd International Conference on Machine Learning (ICML '06)*, pages 233–240, New York, NY, USA, 2006. ACM.

[49] F. De Bona, C. Ong, A. Zien, and G. Ratsch. RNA secondary structure prediction using large margin methods. ISMB, 2007.

[50] L. De Raedt, N. Lavrac, and S. Dzeroski. Multiple predicate learning. In *Proceedings of the 13th International Joint Conference on Artificial Intelligence, Chambery, France*, pages 1037–1043, 1993.

[51] K. Deforche, R. J. Camacho, Z. Grossman, M. a. Soares, K. Van Laethem, D. a. Katzenstein, P. R. Harrigan, R. Kantor, R. Shafer, and A.-M. Vandamme. Bayesian network analyses of resistance pathways against efavirenz and nevirapine. *AIDS (London, England)*, 22(16):2107–15, Oct. 2008.

[52] A. Deshpande, B. Milch, L. S. Zettlemoyer, and L. P. Kaelbling. Learning probabilistic relational dynamics for multiple tasks. In *Proceedings of the 23rd Conference on Uncertainty in Artificial Intelligence (UAI-07)*, pages 83–92, 2007.

[53] J. Drenth and J. Mesters. *Principles of protein X-ray crystallography.* Springer-Verlag, 2007.

[54] J. C. Ebert and R. B. Altman. Robust recognition of zinc binding sites in proteins. *Protein Science*, 17(1):54–65, Jan. 2008.

[55] T. Evgeniou, C. A. Micchelli, and M. Pontil. Learning multiple tasks with kernel methods. *Journal of Machine Learning Research*, 6:615–637, 2005.

[56] T. S. Ferguson. A Bayesian analysis of some nonparametric problems. *The Annals of Statistics*, 1(2):209–230, 1973.

[57] T. Finley and T. Joachims. Supervised clustering with support vector machines. In *Proceedings of the International Conference on Machine Learning (ICML)*, 2005.

[58] R. D. Finn, J. Mistry, B. Schuster-Böckler, S. Griffiths-Jones, V. Hollich, T. Lassmann, S. Moxon, M. Marshall, A. Khanna, R. Durbin, S. R. Eddy, E. L. L. Sonnhammer, and A. Bateman. Pfam: clans, web tools and services. *Nucleic Acids Research*, 34(Database issue):D247–D251, Jan. 2006.

[59] J. D. Fischer, C. E. Mayer, and J. Söding. Prediction of protein functional residues from sequence by probability density estimation. *Bioinformatics*, 24(5):613–620, 2008.

[60] P. Frasconi and A. Passerini. Predicting the geometry of metal binding sites from protein sequence. In *Proceedings of the 22nd Annual Conference on Neural Information Processing Systems (NIPS'08)*, pages 465–472, 2008.

[61] I. Friedberg. Automated protein function prediction–the genomic challenge. *Briefings in Bioinformatics*, 7(3):225–42, Sept. 2006.

[62] T. Gärtner. *Kernels for structured data*, volume 72 of *Series in Machine Perception and Artificial Intelligence.* World Scientific, 2009.

[63] T. Gärtner, P. Flach, and S. Wrobel. On graph kernels: hardness results and efficient alternatives. In *Proceedings of the 16th Annual Conference on Computational Learning Theory and 7th Kernel Workshop*, pages 129–143. Springer-Verlag, Aug. 2003.

[64] L. Getoor and B. Taskar. *Introduction to statistical relational learning*. The MIT Press, 2007.

[65] Z. Ghahramani. Unsupervised learning. *Advanced Lectures on Machine Learning*, pages 72–112, 2004.

[66] P. F. Gherardini and M. Helmer-Citterich. Structure-based function prediction: approaches and applications. *Briefings in Functional Genomics and Proteomics*, 7(4):291–302, 2008.

[67] K. Goyal, D. Mohanty, and S. C. Mande. PAR-3D: a server to predict protein active site residues. *Nucleic Acids Research*, 35(Web server issue):W503–W505, May 2007.

[68] J. A. Hanley and B. J. McNeil. A method of comparing the areas under receiver operating characteristic curves derived from the same cases. *Radiology*, 148(3):839–843, 1983.

[69] D. Haussler. Convolution kernels on discrete structures. Technical Report Technical Report UCS-CRL-99-10, UC Santa Cruz, 1999.

[70] H. Hoos and T. St
"utzle. *Stochastic local search: Foundations and applications*. Morgan Kaufmann, 2005.

[71] S. J. Hubbard and J. M. Thornton. NACCESS computer program. Technical report, Department of Biochemistry and Molecular Biology, University College London, 1993.

[72] T. Hubbard, A. Murzin, S. Brenner, and C. Chothia. SCOP: a structural classification of proteins database. *Nucleic Acids Research*, 25(1):236–9, Jan. 1997.

[73] A. Humm, E. Fritsche, K. Mann, M. Göhl, and R. Huber. Recombinant expression and isolation of human L-arginine: glycine amidinotransferase and identification of its active-site cysteine residue. *Biochemical Journal*, 322(Pt 3):771–776, 1997.

[74] T. Joachims. Making large-scale SVM learning practical. In C. B. B. Schölkopf and A. Smola, editors, *Advances in Kernel Methods - Support Vector Learning*, chapter 11. MIT Press, Cambridge, MA, 1999.

[75] W. Kabsch and C. Sander. Dictionary of protein secondary structure: pattern recognition of hydrogen-bonded and geometrical features. *Biopolymers*, 22:2577–2637, 1983.

[76] H. Kashima, K. Tsuda, and A. Inokuchi. Kernels for graphs. In B. Schölkopf, K. Tsuda, and J.-P. Vert, editors, *Kernel Methods in Computational Biology*. MIT Press, 2004.

[77] S. Kawashima, H. Ogata, and M. Kanehisa. AAindex: Amino Acid index database. *Nucleic Acids Research*, 27(1):368–369, 1999.

[78] C. Kemp, J. B. Tenenbaum, T. L. Griffiths, T. Yamada, and N. Ueda. Learning systems of concepts with an infinite relational model. In *Proceedings of the 21st national conference on Artificial Intelligence (AAAI'06)*, pages 381–388. AAAI Press, 2006.

[79] K. Kersting and L. De Raedt. Bayesian logic programming: theory and tools. In L. Getoor and B. Taskar, editors, *Introduction to statistical relational learning*. MIT Press, Cambridge, 2007.

[80] K. Khan, S. Muggleton, and R. Parson. Repeat learning using predicate invention. In *Proceedings of the 8th Inductive Logic Programming International Workshop (ILP'08), Madison, Wisconsin, USA*,

volume 1446 of *Lecture Notes in Computer Science*, pages 165–174. Springer, 1998.

[81] S. Kim, H. Tang, and E. R. Mardis. *Genome Sequencing Technology and Algorithms.* Artech House, Inc., Norwood, MA, USA, 2007.

[82] J. Ko, L. F. Murga, Y. Wei, and M. J. Ondrechen. Prediction of active sites for protein structures from computed chemical properties. *Bioinformatics*, 21(1):258–265, 2005.

[83] S. Kok and P. Domingos. Learning the structure of Markov logic networks. In *Proceedings of the 22nd international conference on Machine learning*, pages 441–448. ACM, 2005.

[84] R. Koradi, M. Billeter, and K. Wüthrich. MOLMOL: a program for display and analysis of macromolecular structures. *Journal of Molecular Graphics*, 14(1):51–55, 1996.

[85] N. Landwehr, K. Kersting, and L. De Raedt. nFOIL: Integrating naive bayes and FOIL. In *Proceedings of the 20th national conference on Artificial Intelligence (AAAI'05), Pittsburgh, Pennsylvania, USA*, pages 795–800. AAAI Press, 2005.

[86] N. Landwehr, A. Passerini, L. De Raedt, and P. Frasconi. kFOIL: learning simple relational kernels. In *Proceedings of the 21st national conference on Artificial intelligence (AAAI'06)*, pages 389–394. AAAI Press, 2006.

[87] N. Landwehr, A. Passerini, L. Raedt, and P. Frasconi. Fast learning of relational kernels. *Machine Learning*, 78(3):305–342, Jan. 2010.

[88] P. Larranaga. Machine learning in bioinformatics. *Briefings in Bioinformatics*, 7(1):86–112, Feb. 2006.

[89] A. Laurie and R. Jackson. Q-SiteFinder: an energy-based method for the prediction of protein-ligand binding sites. *Bioinformatics*, 21(9):1908–1916, Jan. 2005.

[90] D. Lee, O. Redfern, and C. Orengo. Predicting protein function from sequence and structure. *Nature Reviews Molecular Cell Biology*, 8(12):995–1005, 2007.

[91] T. Lengauer and T. Sing. Bioinformatics-assisted anti-HIV therapy. *Nature Reviews Microbiology*, 4(10):790–797, 2006.

[92] C. Leslie, E. Eskin, and W. Noble. The spectrum kernel: a string kernel for SVM protein classification. In *Proceedings of the Pacific Symposium on Biocomputing*, volume 7, pages 566–575. Hawaii, USA., 2002.

[93] M. Lippi, A. Passerini, M. Punta, B. Rost, and P. Frasconi. MetalDetector: a web server for predicting metal binding sites and disulfide bridges in proteins from sequence. *Bioinformatics*, 24(18):2094–2095, July 2008.

[94] J. Madrid-Sanchez, E. Parrado-Hernandez, and A. Figueiras-Vidal. Selective multitask learning by coupling common and private representations. In *Proceedings of the Workshop on Learning from Multiple Sources @ Neural Information Processing Systems (NIPS'08)*, 2008.

[95] E. C. Meng, B. J. Polacco, and P. C. Babbitt. Superfamily active site templates. *PROTEINS: Structure, Function, and Bioinformatics*, 55:962–976, 2004.

[96] J. Mistry, A. Bateman, and R. Finn. Predicting active site residue annotations in the Pfam database. *BMC Bioinformatics*, 8(1):298, 2007.

[97] A. Moschitti. Efficient convolution kernels for dependency and constituent syntactic trees. In *Proceedings of the 17th European Conference on Machine Learning (ECML), Berlin, Germany, 2006*, Berlin, Germany, 2006.

[98] S. Muggleton. Learning from positive data. In *Proceedings of the Inductive Logic Programming (ILP) Workshop*, pages 358–376, 1996.

[99] S. Muggleton. Learning stochastic logic programs. *Electronic Transactions on Artificial Intelligence*, 4:141–153, 2000.

[100] S. Muggleton and L. De Raedt. Inductive logic programming: Theory and methods. *University Computing*, pages 629–682, 1994.

[101] N. Nagano, C. A. Orengo, and J. M. Thornton. One fold with many functions: The evolutionary relationships between tim barrel families based on their sequences, structures and functions. *Journal of Molecular Biology*, 321(5):741–765, 2002.

[102] R. M. Neal. Density modeling and clustering using dirichlet diffusion trees. In *Bayesian Statistics 7*, pages 619–629, 2003.

[103] J.-C. Nebel. Generation of 3D templates of active sites of proteins with rigid prosthetic groups. *Bioinformatics*, 22(10):1183–1189, May 2006.

[104] C. J. Needham, J. R. Bradford, A. J. Bulpitt, M. a. Care, and D. R. Westhead. Predicting the effect of missense mutations on protein function: analysis with Bayesian networks. *BMC Bioinformatics*, 7:405, 2006.

[105] M. Ondrechen, J. Clifton, and D. Ringe. THEMATICS: a simple computational predictor of enzyme function from structure. *Proceed-*

*ings of the National Academy of Sciences*, 98(22):12473–12478, Jan. 2001.

[106] S. Osawa, T. H. Jukes, K. Watanabe, and A. Muto. Recent evidence for evolution of the genetic code. *Microbiology and Molecular Biology Reviews*, 56(1):229, 1992.

[107] A. Passerini, M. Punta, A. Ceroni, B. Rost, and P. Frasconi. Identifying cysteines and histidines in transition-metal-binding sites using support vector machines and neural networks. *Proteins*, 65(2):305–316, 2006.

[108] P. Pavlidis, T. Furey, M. Liberto, and W. N. Grundy. Learning gene functional classification from multiple data types. *Journal of Computational Biology*, 2002.

[109] P. Pavlidis, T. S. Furey, M. Liberto, and W. N. Grundy. Promoter region-based classification of genes. In *Proceedings of the Pacific Symposium on Biocomputing*, pages 151–163, January 2001.

[110] E. Pennisi. Human genome. A low number wins the GeneSweep Pool. *Science*, 300(5625):1484, June 2003.

[111] N. V. Petrova and C. H. Wu. Prediction of catalytic residues using Support Vector Machine with selected protein sequence and structural properties. *BMC Bioinformatics*, 7:312, 2006.

[112] E. F. Pettersen, T. D. Goddard, C. C. Huang, G. S. Couch, D. M. Greenblatt, E. C. Meng, and T. E. Ferrin. UCSF Chimera–a visualization system for exploratory research and analysis. *Journal of Computational Chemistry*, 25(13):1605–1612, Oct. 2004.

[113] C. T. Porter, G. J. Bartlett, and J. M. Thornton. The Catalytic Site Atlas: a resource of catalytic sites and residues identified in

enzymes using structural data. *Nucleic Acids Research*, 32(Database issue):D129–D133, Jan. 2004.

[114] W. Pullan. Approximating the maximum vertex/edge weighted clique using local search. *Journal of Heuristics*, 14:117–134, 2008.

[115] J. R. Quinlan. Learning logical definitions from relations. *Machine Learning*, 5:239–266, 1990.

[116] M. D. Reid. Improving rule evaluation using multitask learning. In *Proceeding of the 14th International Conference on Inductive Logic Programming (ILP), Porto, Portugal, Proceedings*, volume 3194 of *Lecture Notes in Artificial Intelligence.* Springer, 2004.

[117] S.-Y. Rhee, M. J. Gonzales, R. Kantor, B. J. Betts, J. Ravela, and R. W. Shafer. Human Immunodeficiency Virus reverse transcriptase and protease sequence database. *Nucleic Acids Research*, 31(1):298–303, Jan. 2003.

[118] S.-Y. Rhee, J. Taylor, G. Wadhera, A. Ben-Hur, D. L. Brutlag, and R. W. Shafer. Genotypic predictors of Human Immunodeficiency Virus type 1 drug resistance. *Proceedings of the National Academy of Sciences*, Jan. 2006.

[119] E. Ricci, T. De Bie, and N. Cristianini. Magic moments for structured output prediction. *Journal of Machine Learning Research*, 9:2803–2846, 2008.

[120] L. Richter, R. Augustin, and S. Kramer. Finding Relational Associations in HIV Resistance Mutation Data. In *Proceeding of 19th International Conference on Inductive Logic Programming (ILP'09)*, pages 1–6, June 2009.

[121] D. M. Roy, C. Kemp, V. K. Mansinghka, and J. B. Tenenbaum. Learning annotated hierarchies from relational data. In B. Schölkopf, J. Platt, and T. Hoffman, editors, *Advances in Neural Information Processing Systems 19*, pages 1185–1192. MIT Press, Cambridge, MA, 2007.

[122] S. Sankararaman, B. Kolaczkowski, and K. Sjölander. INTREPID: a web server for prediction of functionally important residues by evolutionary analysis. *Nucleic Acids Research*, 37(Web server issue):W390–W395, Jan. 2009.

[123] B. Schölkopf and A. J. Smola. *Learning with kernels: support vector machines, regularization, optimization, and beyond.* MIT Press, Cambridge, MA, USA, 2001.

[124] B. Schölkopf, K. Tsuda, and Jean-Phi. *Kernel Methods in Computational Biology.* The MIT Press, 2004.

[125] J. Shawe-Taylor and N. Cristianini. *Kernel methods for pattern analysis.* Cambridge University Press, 2004.

[126] N. Shu, T. Zhou, and S. Hovmöller. Prediction of zinc-binding sites in proteins from sequence. *Bioinformatics*, 24(6):775–782, 2008.

[127] T. I. H. C. Study. The Major Genetic Determinants of HIV-1 Control Affect HLA Class I Peptide Presentation. *Science*, pages science.1195271+, Nov. 2010.

[128] Y.-R. Tang, Z.-Y. Sheng, Y.-Z. Chen, and Z. Zhang. An improved prediction of catalytic residues in enzyme structures. *Protein engineering, design & selection : PEDS*, 21(5):295–302, 2008.

[129] B. Taskar, V. Chatalbashev, D. Koller, and C. Guestrin. Learning structured prediction models: a large margin approach. In *Pro-*

*ceedings of the 22nd International Conference on Machine Learning (ICML)*, pages 896–903. ACM, 2005.

[130] W. Taylor. The classification of amino acid conservation*. *Journal of Theoretical Biology*, 119(2):205–218, Mar. 1986.

[131] The UniProt Consortium. The Universal Protein Resource (UniProt) in 2010. *Nucleic Acids Research*, 38:D142–D148, 2010.

[132] S. Thrun and J. O'Sullivan. Discovering structure in multiple learning tasks: the TC algorithm. In *Proceedings of IMCL'96*, 1996.

[133] A. E. Todd, C. A. Orengo, and J. M. Thornton. Evolution of function in protein superfamilies, from a structural perspective. *Journal of Molecular Biology*, 307(4):1113–1143, Apr. 2001.

[134] W. Tong, Y. Wei, L. F. Murga, M. J. Ondrechen, and R. J. Williams. Partial order optimum likelihood (POOL): maximum likelihood prediction of protein active site residues using 3D Structure and sequence properties. *PLoS Computational Biology*, 5(1):e1000266+, 2009.

[135] A. Tramontano. *The ten most wanted solutions in protein bioinformatics.* Chapman & Hall/CRC Mathematical Biology and Medicine Series, 2005.

[136] I. Tsochantaridis, T. Hofmann, T. Joachims, and Y. Altun. Support vector machine learning for interdependent and structured output spaces. In *Proceedings of the 21st International Conference on Machine Learning (ICML)*, pages 823–830. ACM, 2004.

[137] M. Turcotte, S. H. Muggleton, and M. J. Sternberg. Automated discovery of structural signatures of protein fold and function. *Journal of Molecular Biology*, 306(3):591 – 605, 2001.

[138] W. S. Valdar. Scoring residue conservation. *Proteins*, 48(2):227–241, Aug. 2002.

[139] V. Vapnik. *The nature of statistical learning theory.* Springer, 1995.

[140] A. Wagner. Neutralism and selectionism: a network-based reconciliation. *Nature reviews. Genetics*, 9(12):965–974, Dec. 2008.

[141] H. Walter, B. Schmidt, M. Werwein, E. Schwingel, and K. Korn. Prediction of abacavir resistance from genotypic data: impact of zidovudine and lamivudine resistance in vitro and in vivo. *Antimicrobial Agents and Chemotherapy*, 46(1):89, 2002.

[142] Y. Wei, J. Ko, L. F. Murga, and M. J. Ondrechen. Selective prediction of interaction sites in protein structures with THEMATICS. *BMC Bioinformatics*, 8:119, 2007.

[143] D. Whitford. *Proteins: structure and function.* Wiley, 2005.

[144] K. Wüthrich. *NMR of proteins and nucleic acids.* Wiley, 1986.

[145] L. Xie and P. Bourne. A robust and efficient algorithm for the shape description of protein structures and its application in predicting ligand binding sites. *BMC Bioinformatics*, 8:S9, Jan. 2007.

[146] Z. Xu, V. Tresp, K. Yu, and H.-P. Kriegel. Infinite hidden relational models. In *Proceedings of the 22nd Conference on Uncertainty in Artificial Intelligence (UAI 2006)*, Cambridge, MA, USA, July 2006.

[147] Y. Xue, X. Liao, L. Carin, and B. Krishnapuram. Multi-Task Learning for Classification with Dirichlet Process Priors. *Journal of Machine Learning Research*, 8:35–63, 2007.

[148] A. Yamaguchi, K. Iida, N. Matsui, S. Tomoda, and K. Yura. Het-PDB Navi: a database for protein-small molecule interactions. *Journal of Biochemistry*, 135(1):79–84, Jan. 2004.

[149] E. Youn, B. Peters, P. Radivojac, and S. D. Mooney. Evaluation of features for catalytic residue prediction in novel folds. *Protein science : a publication of the Protein Society*, 16(2):216–26, 2007.

[150] C.-N. Yu, T. Joachims, R. Elber, and J. Pillardy. Support vector training of protein alignment models. In *Proceeding of the International Conference on Research in Computational Molecular Biology (RECOMB)*, pages 253–267, 2007.

[151] T. Zhang, H. Zhang, K. Chen, S. Shen, J. Ruan, and L. Kurgan. Accurate sequence-based prediction of catalytic residues. *Bioinformatics*, 24(20):2329–38, 2008.

[152] A. Zien, G. Rätsch, B. S. S. Mika, T. Lengauer, and K. Muller. Engineering support vector machine kernel that recognize translation initiation sites. *Bioinformatics*, 16(9):799–807, 2000.

# List of Publications

[153] E. Cilia and S. Ammendola. Identification of the Amino Acid Residues Affecting the Catalytic Pocket of the Sulfolobus solfataricus Signature Amidase. *Protein & Peptide Letters*, 17(2):146–150(5), Feb. 2010.

[154] E. Cilia, A. Fabbri, M. Uriani, G. G. Scialdone, and S. Ammendola. The signature amidase from Sulfolobus solfataricus belongs to the CX3C subgroup of enzymes cleaving both amides and nitriles: Ser195 and Cys145 are predicted to be the active site nucleophiles. *FEBS Journal*, 272(18):4716–4724, 2005.

[155] E. Cilia, N. Landwehr, and A. Passerini. Mining Drug Resistance Relational Features with Hierarchical Multitask kFOIL. *Proceedings of BioLogical@AI*IA2009*, 2009.

[156] E. Cilia, N. Landwehr, and A. Passerini. Mining Drug Resistance Relational Features with Hierarchical Multitask kFOIL. *Submitted at Fundamenta Informaticae*, 2010.

[157] E. Cilia and A. Moschitti. Advanced tree-based kernels for protein classification. In *Proceeding of AI*IA 2007*, pages 218–229, 2007.

[158] E. Cilia and A. Passerini. Automatic Prediction of Catalytic Residues by Modeling Residue Structural Neighborhood. *BMC Bioinformatics*, 11(1):115, 2010.

[159] E. Cilia and A. Passerini. CatANalyst: a web server for predicting catalytic residues. In *Poster Proceedings of the 9th European Conference of Computational Biology (ECCB2010)*, Ghent, Belgium, Sept. 2010.

[160] E. Cilia and A. Passerini. Frankenstein Junior: a relational learning approach toward protein engineering. In *Proceedings of Annotation, Interpretation and Management of Mutations Workshop (AIMM@ECCB2010)*, Sept. 2010.

[161] E. Cilia, A. Passerini, and M. Brunato. Automatic Prediction of Functional Residues from Sequence and Structural Information. Technical Report DISI-08-036, Department of Engineering and Computer Science (DISI), University of Trento, Trento, July 2008.

[162] F. Mascia, E. Cilia, M. Brunato, and A. Passerini. Predicting structural and functional sites in proteins by searching for maximum-weight cliques. In *Proceedings of the 24th AAAI Conference on Artificial Intelligence (AAAI-10)*, pages 1274–1279. AAAI Press, July 2010.