Automated identification of protein-ligand interaction features using Inductive Logic Programming: A hexose binding case study

Jose C. A. Santos*[1] , Houssam Nassif[2], David Page[2] , Stephen H. Muggleton[1] and Michael J. E. Sternberg[3]

[1]Computational Bioinformatics Laboratory, Department of Computer Science, Imperial College London, London SW7 2BZ
[2]Department of Computer Sciences, Department of Biostatistics and Medical Informatics, University of Wisconsin-Madison, WI-53706, USA
[3]Center for Bioinformatics, Department of Life Sciences, Imperial College London, London, SW7 2AZ

Email: Jose C. A. Santos*- jcs06@doc.ic.ac.uk;

*Corresponding author

Abstract

There is a need for automated methods to learn general features of the interactions of a ligand class with its diverse set of protein receptors. An appropriate machine learning approach is Inductive Logic Programming (ILP) which automatically generates comprehensible rules in addition to prediction. The development of ILP systems which can learn rules of the complexity required for studies on protein structure remains a challenge. In this work we use a new ILP system, ProGolem, and demonstrate its performance on learning features of hexose-protein interactions.

The rules induced by ProGolem detect aromatic and planar-polar residues interactions in addition to less common features like the aromatic sandwich. The rules also reveal a previously unreported dependency for residues CYS and LEU. They also specify stereo configurations involving aromatic and hydrogen bonding residues. Further to confirming literature results, ProGolem's model has a 10-fold cross-validated predictive accuracy that is superior, at the 95% confidence level, to other ILP systems and comparable with state-of-the-art statistical learners.

Introduction

Elucidating unifying features of protein-ligand interactions in systems showing a diversity of interaction modes remains a challenging problem, often requiring extensive human intervention. In this work we present an automated general approach to identify these features using Inductive Logic Programming (ILP). We apply ILP to study the factors relevant to protein-hexose binding.

Hexoses are 6-carbon monosaccharides involved in numerous biochemical processes, including energy release and carbohydrate synthesis [1]. Several non-homologous protein families bind hexoses using a diverse set of protein-ligand interactions. Many research groups have used computational techniques to model and analyze hexose- and sugar-protein interactions, often employing extensive visualization and empirical methods [2–5]. Some techniques use surface and binding site similarities to search for matching functional sites in other proteins [6,7]. Others apply machine learning algorithms to construct sugar-specific classifiers [8, 9]. Such classifiers can be combined with programs that detect protein surface-pockets of a given size [10,11] to discriminate potential binding-sites.

Recently [12] used the predominant ILP system, Aleph [13], to study hexose binding. A powerful feature of ILP is that, in addition to prediction, it automatically learns rules which can be readily understood. It has been successfully applied to predict and model various medical [14, 15] and biological datasets [16, 17]. However, the complexity and size of the hypothesis space often presents computational challenges in search time which limit both the insight and predictive power of the rules found.

Recognizing the limitations of Aleph and other current ILP systems, [18] developed ProGolem to facilitate the learning of long, complex rules. Long, complex rules are common in the molecular biology domain and we argue that a sophisticated ILP system such as ProGolem is a promising approach to automatically learn these rules from molecular data.

The present work extends previous hexose prediction work in multiple ways. First we supplement the background knowledge with both atomic and amino-acid information. Second, we bias the hypothesis space to reduce the search space and increase the likelihood of generating meaningful rules. Third, we employ the newly-developed ProGolem, which has been shown to learn better than Aleph in highly non-determinate domains such as this hexose-binding application. Finally, we explore several approaches to curb the limitations of the *recall* bound in ILP systems.

The combined usage of an extended background knowledge, a better biased search, and the ILP system ProGolem allowed the discovery of more accurate and insightful rules explaining the stereochemistry of hexose binding.

Automatically finding these stereochemical rules and providing their explanation is the main contribution of this paper. While some of the rules ProGolem found were already known from the literature, other rules, namely one that specifies a dependency over residues CYS and LEU, have never been reported but are plausible and require further investigation.

Predicting whether an actual protein binds an hexose is of secondary importance to us. Nevertheless, the predictive accuracy of our approach is competitive to statistical learners such as Support Vector Machines and superior to other logic-based approaches.

Problem Representation

Dataset

For ease of comparison, we use the same dataset and cross-validation folds described in [12]. To retrieve the positive examples, the Protein Data Bank (PDB) [19] was mined for proteins crystallized with the most common hexoses: galactose, glucose and mannose [20]. Theoretical structures and files older than PDB format 2.1 were ignored. Glycosylated sites and redundant structures (at most 30% overall sequence identity using PISCES [21]) were also ignored. The positive subset consisting of 80 protein-hexose binding sites (33 galactose, 35 glucose and 12 mannose) is presented in Table 1.

The Protein Data Bank was mined in a similar way for the 80 negative examples. The negative dataset consists of 22 binding sites that bind hexose-like ligands (e.g. hexose or fructose derivatives, 6-carbon molecules, and molecules similar in shape to hexoses), 27 other-ligand binding sites and 31 non-binding sites. The non-binding sites are surface pockets that look like binding sites but are not known to bind any ligand. The negative dataset is presented in Table 2 (non-hexose binding sites) and Table 3 (non-binding sites).

The data also specifies the center for each of the resulting 160 examples. The binding-site center is computed as the hexose pyranose ring centroid for the positive examples and as the ligand or empty pocket centroid for the negative examples. The binding-site consists of all protein atoms present within a 10 Å radius sphere around this center. All other atoms are discarded.

Inductive Logic Programming

ILP is a machine learning approach that generates a hypothesis composed of a set of logical if-then rules that explains a given dataset [22]. ILP has three major advantages over other machine learning and data mining techniques. First, it allows an easy interaction between humans and computers by using background

knowledge to guide the search. Second, it returns results in an easy-to-understand if-then format. Finally, ILP can easily operate on relational databases, as relational databases are naturally expressed as relations in first-order logic.

Most leading ILP approaches start by a saturation step, randomly selecting a positive example for which they construct the *bottom clause*: the most specific hypothesis that explains the example. This most specific clause is the rule formed by the conjunction of all features (called predicates or literals) pertaining to the chosen example. In the reduction step, ILP generalizes this rule (called clause or hypothesis) to include other positive examples using one of two basic induction methods, generalization or specialization.

Aleph, using a general-to-specific approach, starts with the most general hypothesis, "all sites are hexose-binding sites", calling all examples positives. It then refines this hypothesis by repeatedly adding the literal from the bottom-clause that best improves the hypothesis score. The new rule will be more specific, covering only a subset of the examples previously covered.

ProGolem, in contrast, uses a specific-to-general search. Starting with the bottom clause, it successively drops a minimal set of literals to allow coverage of one additional positive example. By dropping this set of literals the clause becomes more general, and will cover a superset of the examples previously covered.

Both ProGolem and Aleph stop hypothesis refinement when the hypothesis score stops improving. A rule scores well if it covers many positive and few negative examples. If the rule passes a certain performance threshold, it is added to the *theory*, and all the positive examples it covers are removed. The cycle of saturation and reduction continues on the remaining examples. When all positive examples are covered or no new rules can be found, the ILP system outputs its theory, the set of all compressive rules found thus far. The compression score of a rule is defined as: *positive examples covered - negative examples covered - clause length*.

The compressive rules found in the training phase are later used to predict the class of new, test, examples. A new example (i.e. a new protein) is classified as positive (i.e. binds an hexose) if it is covered by any of the discovered rules, otherwise it is labeled as negative.

The newly developed ProGolem is more than a specific-to-general version of Aleph. Two additional features set it apart. Aleph adopts a local theory construction method, incrementally adding a new rule to its theory after each reduction cycle. This method depends on the ordering of the positive examples, and it is possible that the best rules are not generated. This situation may occur if these better rules would be generated by examples that were removed by previous sub-optimal rules. By contrast, ProGolem implements a global theory construction approach, which ensures that the theory is only constructed after all rules have

been generated. ProGolem repeatedly adds to the theory the rule that best improves the global theory score. The global-theory-construction feature of ProGolem is especially useful in this application.

In ILP, an example can have multiple instances from the same predicate. For example, a binding site has multiple atoms. If a predicate has more than one possible solution, it is called *non-determinate*. Hence the site *has_atom* predicate is non-determinate. In fact our hexose dataset is highly non-determinate.

When evaluating a clause, Aleph will proceed literal by literal from left to right. This is the standard Selective Linear Definite (SLD) resolution [23], which is the only option in most ILP systems. However, ProGolem has to evaluate longer clauses than Aleph due to its specific-to-general hypothesis search. SLD-resolution is too slow to compute the coverage of such long clauses. To cope with this problem ProGolem supports the usage of different resolution engines, including the smallest variable domain resolution, which enumerates the possible values a variable in a literal may take and, during clause evaluation, chooses at each moment the variable with the smallest domain [24]. This clause evaluation engine is better suited to our problem than SLD-resolution, drastically reducing the runtime per evaluated clause.

Background knowledge

The background knowledge is the set of features, facts and rules known a priori. This is given to the ILP system as a basis for learning and constructing the classification rules. The piece of background knowledge central to our task is the binding site representation. Figure 1 is an excerpt of the background knowledge for protein 1BDG. The *center_coords* predicate specifies the binding-site center coordinates, which is the pyranose ring centroid of the bound glucose in this case. The *has_aminoacid* predicate specifies each amino acid present within the protein binding site, listing its unique identifier and name. The *has_atom* predicate details the residue atoms, specifying the PDB atom name and its coordinates.

By extracting the coordinates of the center and the various atoms, we compute their respective distances. We set a tolerance of 0.5 Å on distances between atoms, a sensible error margin in a hexose binding site [25].

In addition to these facts, ILP allows for a higher level of expressiveness within its background knowledge: human coded rules. Using the facts of Figure 1, and the Euclidean distances between atoms that are derived from this data, we can now define the predicates *atom_to_center_dist* and *atom_to_atom_dist*.

These predicates respectively compute the distance between an atom and the binding-site center, and between two atoms. We also define a *diff_aminoacid* predicate which allows expressing that two amino acids are different. This may be relevant when there are multiple amino acids of the same type and each amino acid needs to be individually identified.

Hypothesis space

We experiment with multiple hypothesis spaces. Similarly to [12], we first exclude residue information and limit the background to the atoms and their 3D coordinates. In this *atom-only* representation, the binding site is a sphere of radius 10 Å containing atoms in space, for which distances can be computed (Table 4).

The distances between atoms are computed by having a *dist* literal in the ILP background knowledge, allowing ILP systems to express the 3D conformation of the binding site. However, the number of possible distances grows quadratically with the number of atoms considered, resulting in an exponential growth of the bottom clause. Starting its generalization search from the bottom clause, ProGolem learning time is highly sensitive to its length. To keep learning tractable, both ProGolem and to a lesser extent Aleph, require a bound on the maximum number of solutions a given predicate may return, called the *recall* (not to be confused with the statistical measure of the same name, also called sensitivity). In practice this *recall* bound limits the hypothesis search space by forcing that only the **first** *recall* solutions of a literal be considered in the bottom clause.

By relying on the arbitrary ordering of the atoms and residues in the background knowledge, having a bound on the *recall* of a predicate is subject to data idiosyncrasies. In this work we explored two alternative approaches of organizing the background knowledge to curb the limitations of having a recall bound. The first approach, *randomized recall*, considers all solutions first, out of which it randomly picks a number of solutions equal to *recall* rather than the always the first N. This is achieved by either altering the internal recall routine, as we did, or equivalently, by randomizing the order of the atoms and residues in the background knowledge.

The second approach is *domain-dependent*. Using Random Forests to measure feature importance [26,27], [9] show that atoms closest to the binding center have higher discriminative power. Closest atoms are more likely to determine whether or not the binding site binds hexose, as compared to more distant atoms. The *domain-dependent* approach orders the atoms and residues in the background knowledge by their distance from the binding site center. For instance, in this approach the distance literal will attempt to match the *recall* atoms closest to the binding center.

Another contribution of this work is the re-modeling of the problem representation and a better bias to the hypothesis space. We propose two major improvements to the atomic representation. First is the inclusion of residues using the *has_aminoacid* predicate. The second is imposing that atoms cannot appear dangling in a hypothesis. A residue has to be introduced into a rule first, before *atom_to_atom_dist* and *atom_to_center_dist* predicates compute its atomic distances. We thus only compute distances between atoms of residues already in a rule. In this *amino acid* representation, the binding site sphere is composed of amino acids, who in turn

contain atoms (Table 4).

By first dealing with residues instead of atoms, the binding site sphere now contains a smaller number of elements. In addition, in the *amino acid* representation we can express the distance between two atoms using only one literal, *atom_to_atom_dist*.A rule can contain up to *recall* residues, and for each atom of a given residue we measure its distance to *recall* atoms of each one of the included residues. In contrast, in the *atom-only* representation we need three literals to express a distance, two *has_atom* and one *dist*. A rule can contain only *recall* atoms, and each atom can only detect *recall* other atoms in the feature space. Thus, for the same recall bound, the *amino acid* representation considers both more features and generates more informative clauses than the *atom-only* representation.

Figure 2 is an example of a hypothesis from the *amino acid* representation hypothesis space, in raw Prolog format as induced by ProGolem.

When interpreting Prolog clauses, it is important to note that these have a structure *Head:-Body*, which reads as, the head is verified (i.e. is true) if the body (a conjunction of literals) holds true. The uppercase letters in the clause, in this case A, B, C and D, are logical variables and represent a certain entity. Lowercase strings, string within quotes (e.g. leu, cys, 'N', 'OD2', and 'C') and numbers are constants representing themselves.

The variable A in this clause represents a protein, variables B, C and D represent amino acids. The types of variables and constants, is specified in ILP via mode declarations. Figure 3 has the translation to English of the rule in Figure 2.

Experiments

Methods

All materials (i.e. dataset, ILP systems and scripts) to reproduce these experiments are available at http://www.doc.ic.ac.uk/∼jcs06/Hexose.

ILP settings

We apply two ILP systems, Aleph and ProGolem, with both *atom-only* and *amino acid* representations, and using YAP 6.0.6 as the Prolog compiler [28]. To ensure a fair comparison, we use the same settings for both ILP algorithms whenever possible. We set the recall bound to 7, and the maximum number of negatives a hypothesis may cover to 5. We evaluate clauses according to their compression score.

We use ProGolem with its global theory construction and smallest variable domain resolution. In Aleph,

we set the number of nodes to be explored when searching for an acceptable clause to 5000. The clause length in Aleph, i.e. the maximum number of literals allowed in a hypothesis, was set to match the clause length that ProGolem generates (5 for *atom-only*, 6 for *amino acid*). Notice that if the same clause length was used in both representations, the predictive accuracies of Aleph would be lower. In ProGolem the user does not need to specify the maximum clause length of a rule, as the hypothesis search is from specific to general.

Homology and cross-validation

Our dataset consists of 160 binding sites, belonging to 152 unique proteins (8 of the hexose-binding proteins have two distinct binding sites). These 152 proteins belong to a total of 122 CATH [29] superfamilies. In order to guarantee that rules are not being learned from homologous proteins, each cross-validation fold should not contain proteins whose superfamilies also occur in other folds.

Unfortunately, with this particular dataset, it is impossible to construct cross-validation folds that verify this non-sharing superfamily constraint. This is because the binding site may span over multiple chains, each belonging to a different superfamily. Moreover, a single chain may be subdivided into domains, each belonging to different CATH superfamilies. Thus, if binding site A belongs to superfamilies $sf1$ and $sf2$, B to $sf2$ and $sf3$, and C to $sf3$ and $sf4$, the binding sites A, B and C must be in the same cross-validation fold. With the current dataset this constraint would result in a single cross-validation fold containing 48 binding sites (34 positives, 14 negatives) out of 160, creating a significative imbalance between cross-validation folds.

Given this impossibility, and in order for our results to be comparable with previous work, we performed a 10-fold cross-validation using the same folds as [12]. Each fold consists of 8 positive and 8 negative examples. Since the number of hexose binding proteins is limited, the dataset proteins share a low sequence identity ($\leq 30\%$), and the main goal of this paper is providing insight into the hexose-binding discriminating process rather than the predictive accuracy of the classifiers, we consider our methodology acceptable. When comparing two approaches or algorithms on the 10 folds, we consistently use a two-tailed paired t-test at the 95% confidence level.

Results

It is important to note that the main aim of this work is to discover rules describing the bio- and stereo-chemistry of protein-hexose binding. Although there is empirical evidence suggesting that many hexose dockings are not accompanied by substantial protein conformational changes [25], we do not aim to predict

the binding sites of new hexose-binding proteins, as we would not know in advance the coordinates of the hexose ligand. Nevertheless, we use 10-folds cross-validated predictive accuracies as a measure to demonstrate the quality of the rules.

As explained previously, an important parameter when running ILP systems, is the *recall* bound, which imposes a bound on the maximum number of solutions a given Prolog predicate may return.

Since, for performance reasons, the *recall* setting has to be limited to a relatively low value, we started by performing experiments to determine how to best set the background knowledge to get the most out of a limited recall.

We considered three schemes. The first considers the atoms of the protein according to the order of their occurrence in the PDB file, which follows the order of the primary sequence. The second scheme randomizes the order of the atoms in the background knowledge. The third scheme, *domain-dependent*, orders the atoms by their distance to the binding-site center.

Using the *atom-only* representation each of the three approaches yielded, in ProGolem, a 10-fold CV accuracy of, respectively, 59.4%, 68.8% and 74.4%. Therefore, after this initial set of experiments, we adopted the *domain-dependent* approach to organize the background knowledge in all our subsequent runs, involving both ILP algorithms (ProGolem and Aleph) and both data representations (*atom-only* and *amino acid*).

We then compared our results to a state-of-the-art approach, where we first use Random Forests for feature selection, and then apply Support Vector Machines (SVM) [30] to the selected attributes. We used internal validation to select the best Random Forests and SVM parameters for each training fold, before predicting the testing fold. Note that SVM is a statistical classifier requiring a constant-length feature vector as input. This requires a different problem representation than the one used with ILP. Essentially we divide the binding site in concentric spherical layers, and for each we compute atomic and residue properties. We also add various atomic features namely hydrophobicity, charge and hydrogen-bonding. Refer to [9] for method and representation details.

Table 6 shows the 10-fold cross-validation predictive accuracies of Aleph and ProGolem with the *atom-only* and *amino acid* representations. We used the same folds for SVM.

Discussion

ProGolem performance

Simply relying on the given order of the background knowledge introduces placement bias. Both randomizing recall selection, and incorporating domain knowledge by ordering the atoms according to their distance to the binding site center, significantly improves accuracy when compared to the given PDB ordering (p-values of 0.026 and 0.021, respectively). This showcases the importance of domain knowledge, whereas clever manipulations based on prior knowledge will have better results compared to default settings. We also argue that randomizing recall selection should be used as default since it avoids data idiosyncrasies.

From Table 6 we notice that ProGolem performs better using the enhanced *amino acid* representation rather than the *atom-only* one (p-value = 0.029). However, the *amino acid* representation yields no statistically significant improvement in Aleph (p-value = 0.390). A possible explanation as to why ProGolem takes advantage of the amino acid representation more than Aleph is the myopia effect [31]. The myopia effect occurs because general-to-specific ILP systems, like Aleph, indirectly assume literals are conditionally independent given the target class. They refine the working hypothesis by adding one literal at a time, the one that maximizes a fitness function. If literals have a strong conditional dependency, any selected literal will roughly have the same score. Thus multiple literals need to be added before Aleph can determine which set is optimal. If the literals are highly non-determinate, as is our case, a significant portion of the search resources is wasted searching very similar hypotheses, which results in a poorer chance of finding good theories.

We also notice that ProGolem outperforms Aleph for both representations (Table 6). The differences in their predictive accuracies are statistically significant for both *atom-only* (p-value = 0.043) and *amino acid* (p-value = 0.004) representations, the latter being significant even at the 99% confidence level. This discrepancy is in part explained by ProGolem's global theory construction, which only constructs the final theory after all hypotheses have been generated rather than incrementally, on a per-example basis, as Aleph does.

Finally, we compare ILP to SVM. Using the *amino acid* representation, despite ProGolem having a higher average accuracy and a lower standard deviation than SVM, the difference is not statistically significant (p-value = 0.52). More surprisingly, SVM do not outperform *amino acid* Aleph (p-value = 0.057). SVM significantly outperforms both Aleph (p-value = 0.005) and ProGolem (p-value = 0.025) in the *atom-only* representation.

Insight from rules

In this section we present the English translation and the biological explanation for some of the most relevant rules found by ProGolem using the *amino acid* representation. ProGolem rules were judged by a structural bioinformatician to be more interesting than those found by Aleph. According to ProGolem a site is hexose-binding if:

1. It contains two different ASN residues and an ASP residue whose CG atom is 5.4 ± 0.5 Å away from the binding center.
   [Positives covered = 37, Negatives covered = 4]

2. It contains an ASN whose N and C atoms are 2.4 ± 0.5 Å apart, and a GLU whose CB and CG atoms are 8.0 ± 0.5 Å and 6.9 ± 0.5 Å away from the binding center, respectively.
   [Positives covered = 24, Negatives covered = 0]

3. It contains an ASN residue whose N atom is 8.2 ± 0.5 Å away from the binding center, and an ASN residue whose N and ND2 atoms are 4.1 ± 0.5 Å apart and whose N and O atoms are 3.6 ± 0.5 Å apart.
   [Positives covered = 30, Negatives covered = 0]

4. It contains a TRP residue whose CB atom is 7.1 ± 0.5 Å away from the binding center, and whose N and CD1 atoms are 4.0 ± 0.5 Å apart.
   [Positives covered = 14, Negatives covered = 0]

5. It contains a TYR residue whose CB and OH atoms are 5.6 ± 0.5 Å apart, a HIS residue whose ND1 atom is 8.9 ± 0.5 Å away from the binding center, and a TYR residue whose O atom is 9.8 ± 0.5 Å away from the binding center.
   [Positives covered = 6, Negatives covered = 0]

6. It contains CYS and LEU residues, and an ASP residue whose N and OD2 atoms are 4.6 ± 0.5 Å apart, and whose C atom is 7.6 ± 0.5 Å away from the binding center.
   [Positives covered = 18, Negatives covered = 0]

The first rule requires the presence of an ASP and two ASNs. Early on, [32] highlighted the importance of both residues in hexose binding. Studying the lectin protein family, they report that the 3D position of binding-site ASP and ASN residues is conserved across the protein family; despite the fact that lectins bind various types of hexoses and exhibit different sugar-binding specificities.

That same rule requires the ASP CG atom to be 5.4 Å away from the centroid of the hexose pyranose ring. The pyranose radius itself being 3 Å, the ASP actually interfaces the docked hexose. Binding-site interface residues are key for hexose recognition and binding [9], especially planar polar residues that establish a network of hydrogen bonds with the various hydroxyl groups of the docked hexose [33]. [34] report that the most common planar polar amino acids involved in hexose binding are mainly ASP and ASN, followed by GLU. ProGolem detects the role of GLU in the second rule.

The second rule also implies a triangular distance relationship between GLU's CB and CG atoms, and the binding center. [25] report that spatial disposition of protein-galactose interacting atoms is not conserved *per se*, but is conserved with respect to the docking position of the ligand. Similarly, ProGolem often specifies the distance of an atom with respect to the centroid of the hexose.

An additional advantage of inducing rules using ILP is the straightforward reverse-engineering to find the particular proteins, residues and atoms covered by a given rule. This is achieved by executing the ILP rule in a Prolog interpreter. As an example, Figure 4 visualizes the second rule with protein 1HIZ, a xylanase. The hexose ligand is depicted with its backbone in light pink. The two amino acids involved in the rule, a glutamic acid and an asparagine, have a white backbone. The relevant distances are shown.

In addition to specifying the distance from the binding center, ProGolem can detect specific amino acid stereochemical dispositions. The third rule determines a particular ASN conformation, specifying the distances between backbone N and O atoms, and the side chain ND2 atom. The various spatial dispositions of the different rules need further investigation to compare them with known 3D hexose binding-site conformations.

The aromatic residues (TRP most frequently, TYR, PHE, and to a lesser extent HIS) provide a stacking platform for the hexose to dock on [34]. The hexose pyranose ring forms a planar apolar hydrophobic side that stacks, through hydrophobic and van der Waals interactions, over the aromatic residues planar apolar hydrophobic side chain ring [35]. Similarly, the ProGolem fourth rule requires the presence of TRP in a particular stereochemical conformation.

The fifth rule requires the presence of one or two TYR, and a HIS. This rule is thus describing a conformational representation of two or three aromatic residues around the binding-site center. It is interesting that this low-coverage rule may indeed be capturing the infrequent sandwich interaction, whereby two or more aromatic residues engage both faces of a hexose pyranose ring [36].

The last rule specifies CYS and LEU residues. Both have negative interface propensity measures and do not form hydrogen bonds with hexoses [37]. To quantify the disposition of each amino acid to be in contact with the docked sugar, [37] devised an interface propensity measure, defined as the logarithm of the

ratio between a residue frequency at the sugar binding site, and the average frequency of any residue at the binding site. They compute and report the sugar-interface propensity measure for the 20 common amino acids. A residue with a negative propensity measure does not favor the sugar binding-site region since it is present there less frequently than average.

This rule covers 18 positive examples and no negative examples, and clearly specifies the presence of CYS and LEU as a discriminative factor for hexose-binding site recognition. This dependency over LEU and CYS is not previously identified in literature and merits further attention.

Conclusion

Inductive Logic Programming (ILP) is a leading technique to mine accurate and comprehensible rules. The newly developed ILP system ProGolem is well suited for complex non-determinate problems, like most biochemical datasets. In our hexose-binding application, its predictive accuracy is significantly better than previous approaches, while showing a clear insight of the underlying discrimination process.

ProGolem was able to infer different aspects of the established biochemical information about hexose-binding, namely the presence of a docking aromatic residue, the importance of interface atoms, and the hydrogen-bonding activity of planar-polar residues (ASN, ASP, GLU). ProGolem also detected the less common aromatic sandwich interaction.

In addition, ProGolem reveals an important previously unreported finding: a dependency over residues CYS and LEU. It also specifies stereo configurations involving aromatic and hydrogen bonding residues. The newly reported relationship and 3D conformations require further investigation.

Author's contributions

Jose Santos developed the ProGolem ILP system under the supervision of Stephen Muggleton. ProGolem's theoretical foundations were laid out by Stephen Muggleton. Houssam Nassif and David Page created the Hexose dataset and did the initial experiments with Aleph. Jose did the current experimental evaluation between ProGolem, Aleph and Support Vector Machines.

Jose drafted the computational sections of the manuscript while Houssam drafted the biological sections. In particular the biological interpretation of the ProGolem rules was written by Houssam. Michael Sternberg reviewed and edited the biological interpretation of the rules, wrote the introductory section and provided critical input on several iterations of the manuscript. All authors read and approved the final manuscript.

Acknowledgment

This work was partially supported by the US National Institute of Health grant R01CA127379-01. Jose Santos thanks the Wellcome Trust for funding his Ph.D. scholarship. Stephen Muggleton thanks the Royal Academy of Engineering and Microsoft for funding his Research Chair. We thank Dr Suhail Islam for his help with Figure 4 and Professor Kurt Drickamer for comments on the rules.

References

1. Solomon E, Berg L, Martin DW: *Biology*. Belmont, CA: Brooks Cole, 8th edition 2007.
2. Shionyu-Mitsuyama C, Shirai T, Ishida H, Yamane T: **An Empirical Approach for Structure-Based Prediction of Carbohydrate-Binding Sites on Proteins**. *Protein Engineering* 2003, **16**(7):467–478.
3. Sujatha MS, Sasidhar YU, Balaji PV: **Energetics of Galactose- and Glucose-Aromatic Amino Acid Interactions: Implications for Binding in Galactose-Specific Proteins**. *Protein Science* 2004, **13**(9):2502–2514.
4. Chakrabarti R, Klibanov AM, Friesner RA: **Computational prediction of native protein ligand-binding and enzyme active site sequences**. *Proceedings of the National Academy of Sciences of the United States of America* 2005, **102**(29):10153–10158.
5. Doxey AC, Cheng Z, Moffatt BA, McConkey BJ: **Structural motif screening reveals a novel, conserved carbohydrate-binding surface in the pathogenesis-related protein PR-5d**. *BMC Structural Biology* 2010, **10**:23.
6. Gold ND, Jackson RM: **Fold independent structural comparisons of protein-ligand binding sites for exploring functional relationships**. *Journal of Molecular Biology* 2006, **355**(5):1112–1124.
7. Cipriano G, Wesenberg G, Grim T, Jr GNP, Gleicher M: **GRAPE: GRaphical Abstracted Protein Explorer**. *Nucleic Acids Research* 2010, **38**:W595–W601.
8. Malik A, Ahmad S: **Sequence and Structural Features of Carbohydrate Binding in Proteins and Assessment of Predictability Using a Neural Network**. *BMC Structural Biology* 2007, **7**:1.
9. Nassif H, Al-Ali H, Khuri S, Keirouz W: **Prediction of Protein-Glucose Binding Sites Using Support Vector Machines**. *Proteins* 2009, **77**:121–132.
10. Kawabata T: **Detection of multi-scale pockets on protein surfaces using mathematical morphology**. *Proteins* 2010, **78**(5):1195–1211.
11. Wong GY, Leung FH: **Predicting Protein-Ligand Binding Site with Support Vector Machine**. In *Proceedings of the IEEE Congress on Evolutionary Computation* 2010:1–5.
12. Nassif H, Al-Ali H, Khuri S, Keirouz W, Page D: **An Inductive Logic Programming Approach to Validate Hexose Biochemical Knowledge**. In *Proceedings of the 19th International Conference on ILP*, Leuven, Belgium 2009:149–165.
13. Srinivasan A: *The Aleph Manual*. 4th 2007, [http://www.comlab.ox.ac.uk/activities/machinelearning/Aleph/%aleph.html].
14. Srinivasan A, King RD, Muggleton SH, Sternberg MJE: **Carcinogenesis Predictions Using ILP**. In *Proceedings of the 7th International Workshop on Inductive Logic Programming*, Prague, Czech Republic 1997:273–287.
15. Burnside ES, Davis J, Santos Costa V, de Castro Dutra I, Kahn CE, Fine J, Page D: **Knowledge Discovery from Structured Mammography Reports Using Inductive Logic Programming**. In *AMIA Symposium Proceedings*, Washington, DC 2005:96–100.
16. Finn P, Muggleton S, Page D, Srinivasan A: **Pharmacophore Discovery using the Inductive Logic Programming System PROGOL**. *Machine Learning* 1998, **30**(2-3):241–270.
17. Szaboova A, Kuzelka O, Zelezny F, Tolar J: **Prediction of DNA-Binding Proteins from Structural Features**. In *Proceedings of the 4th International Workshop on Machine Learning in Systems Biology*, Edinburgh 2010:273–287.
18. Muggleton S, Santos J, Tamaddoni-Nezhad A: **ProGolem: a system based on relative minimal generalisation**. In *Proceedings of the 19th International Conference on ILP, Springer*, Leuven, Belgium 2009:131–148.
19. Berman HM, Westbrook J, Feng Z, Gilliland G, Bhat TN, Weissig H, Shindyalov IN, Bourne PE: **The Protein Data Bank**. *Nucleic Acids Research* 2000, **28**:235–242.
20. Fox MA, Whitesell JK: *Organic Chemistry*. Boston, MA: Jones & Bartlett Publishers, 3rd edition 2004.
21. Wang G, Dunbrack RL: **PISCES: A Protein Sequence Culling Server**. *Bioinformatics* 2003, **19**(12):1589–1591.
22. Mitchell TM: *Machine Learning*. Singapore: McGraw-Hill International Editions 1997.

23. Kowalski RA, Kuehner D: **Linear Resolution with Selection Function**. *Artificial Intelligence* 1971, **2**(3/4):227–260.

24. Santos J, Muggleton S: **Subsumer: A Prolog theta-subsumption engine**. In *Technical communications of the International Conference on Logic Programming*, Edinburgh, Scotland 2010:172–181.

25. Sujatha MS, Balaji PV: **Identification of Common Structural Features of Binding Sites in Galactose-Specific Proteins**. *Proteins* 2004, **55**:44–65.

26. Breiman L: **Random Forests**. *Machine Learning* 2001, **45**:5–32.

27. Díaz-Uriarte R, de Andrés SA: **Gene Selection and Classification of Microarray Data Using Random Forest**. *BMC Bioinformatics* 2006, **7**:3.

28. Santos Costa V: **The Life of a Logic Programming System**. In *Proceedings of the 24th International Conference on Logic Programming*. Edited by de la Banda MG, Pontelli E, Udine, Italy 2008:1–6.

29. Orengo C, Michie A, Jones S, Jones D, Swindells M: **CATH—a hierarchic classification of protein domain structures**. *Structure* 1997, **5**:1093–1108.

30. Vapnik VN: *Statistical Learning Theory*. New York: John Wiley & Sons 1998.

31. Kononenko I, Simec E, Robnik-Sikonja M: **Overcoming the Myopia of Inductive Learning Algorithms with RELIEFF**. *Appl. Intell.* 1997, **7**:39–55.

32. Rao VSR, Lam K, Qasba PK: **Architecture of the Sugar Binding Sites in Carbohydrate Binding Proteins—a Computer Modeling Study**. *International Journal of Biological Macromolecules* 1998, **23**(4):295–307.

33. Zhang Y, Swaminathan GJ, Deshpande A, Boix E, Natesh R, Xie Z, Acharya KR, Brew K: **Roles of individual enzyme-substrate interactions by alpha-1,3-galactosyltransferase in catalysis and specificity**. *Biochemistry* 2003, **42**(46):13512–13521.

34. Quiocho FA, Vyas NK: **Atomic Interactions Between Proteins/Enzymes and Carbohydrates**. In *Bioorganic Chemistry: Carbohydrates*. Edited by Hecht SM, New York: Oxford University Press 1999:441–457.

35. Screen J, Stanca-Kaposta EC, Gamblin DP, Liu B, Macleod NA, Snoek LC, Davis BG, Simons JP: **IR-Spectral Signatures of Aromatic–Sugar Complexes: Probing Carbohydrate–Protein Interactions**. *Angew. Chem. Int. Ed.* 2007, **46**:3644–3648.

36. Boraston AB, Bolam DN, Gilbert HJ, Davies GJ: **Carbohydrate-binding modules: fine-tuning polysaccharide recognition**. *Biochem. J.* 2004, **382**:769–781.

37. Taroni C, Jones S, Thornton JM: **Analysis and Prediction of Carbohydrate Binding Sites**. *Protein Eng.* 2000, **13**(2):89–98.

Figures

Figure 1: **Excerpt of the background knowledge for protein 1BDG in Prolog.** Since 1BDG is a hexose-binding protein, center_coords/2 predicate states the coordinates of the hexose binding center. The has_aminoacid and has_atom predicates state the coordinates of the amino acids and atoms in a neighborhood of 10 Å of the binding site center.

```
center_coords(p1BDG, p(27.0,22.1,64.9)).
has_aminoacid(p1BDG, a64, phe).
has_aminoacid(p1BDG, a85, leu).
has_aminoacid(p1BDG, a86, gly).
has_aminoacid(p1BDG, a87, gly).
has_atom(p1BDG, a64, 'CD2', p(22.4,13.3,65.5)).
has_atom(p1BDG, a64, 'CE2', p(21.6,14.0,66.4)).
has_atom(p1BDG, a85, 'C', p(24.6,25.9,57.4)).
has_atom(p1BDG, a85, 'O', p(24.6,24.8,57.8)).
has_atom(p1BDG, a86, 'N', p(24.8,27.0,58.3)).
has_atom(p1BDG, a86, 'CA', p(24.9,26.8,59.7)).
```

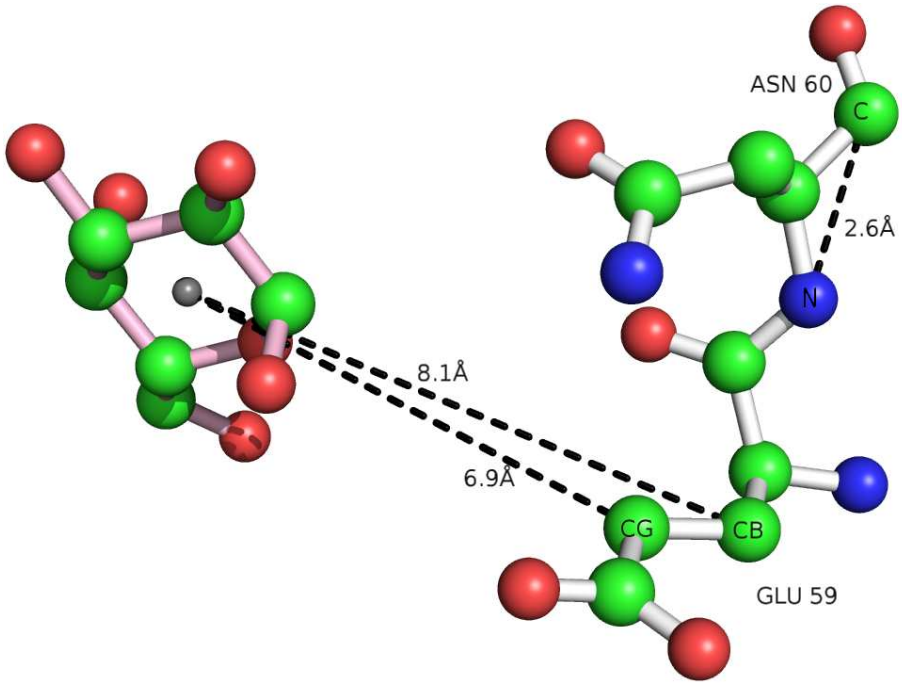
Figure 2: **Example of a hypothesis from the hypothesis space considering *amino acid* mode declarations**

```
bind(A):-
  has_aminoacid(A,B,asp),
  atom_to_atom_dist(B,B,'N','OD2',4.6,0.5),
  has_aminoacid(A,C,leu),
  has_aminoacid(A,D,cys),
  atom_to_center_dist(B,'C',7.6,0.5).
```

Figure 3: **English translation of the rule from Figure 2**

```
A protein is hexose-binding if:
 the N and OD2 atoms of an aspartic acid are 4.6+/-0.5 Angstroms away from each other and
 the C atom of this aspartic acid is 7.6+/-0.5 Angstroms away from the binding center.
 A leucine and a cysteine are also present.
```

Figure 4: **A visualization of the second ProGolem rule instantiated with protein 1HIZ (covered by the rule).** In the left, with a pink backbone, there is the Hexose. On the right, with a white backbone, there are the two aminoacids, an ASN and a GLU, in closer contact with the Hexose. The dotted black lines highlight the distances between the atoms in the aminoacids and the center of the Hexose.



Tables

Table 1: **The positive dataset, composed of** 80 **non-redundant protein-hexose binding sites.**

Hexose	PDB ID	Ligand	PDB ID	Ligand
Glucose	1BDG	GLC-501	1ISY	GLC-1471
	1EX1	GLC-617	1J0Y	GLC-1601
	1GJW	GLC-701	1JG9	GLC-2000
	1GWW	GLC-1371	1K1W	GLC-653
	1H5U	GLC-998	1KME	GLC-501
	1HIZ	GLC-1381	1MMU	GLC-1
	1HIZ	GLC-1382	1NF5	GLC-125
	1HKC	GLC-915	1NSZ	GLC-1400
	1HSJ	GLC-671	1PWB	GLC-405
	1HSJ	GLC-672	1Q33	GLC-400
	1I8A	GLC-189	1RYD	GLC-601
	1ISY	GLC-1461	1S5M	AGC-1001
	1SZ2	BGC-1001	1SZ2	BGC-2001
	1U2S	GLC-1	1UA4	GLC-1457
	1V2B	AGC-1203	1WOQ	GLC-290
	1Z8D	GLC-901	2BQP	GLC-337
	2BVW	GLC-602	2BVW	GLC-603
	2F2E	AGC-401		
Galactose	1AXZ	GLA-401	1MUQ	GAL-301
	1DIW	GAL-1400	1NS0	GAL-1400
	1DJR	GAL-1104	1NS2	GAL-1400
	1DZQ	GAL-502	1NS8	GAL-1400
	1EUU	GAL-2	1NSM	GAL-1400
	1ISZ	GAL-461	1NSU	GAL-1400
	1ISZ	GAL-471	1NSX	GAL-1400
	1JZ7	GAL-2001	1OKO	GLB-901
	1KWK	GAL-701	1OQL	GAL-265
	1L7K	GAL-500	1OQL	GAL-267
	1LTI	GAL-104	1PIE	GAL-1
	1R47	GAL-1101	1S5D	GAL-704
	1S5E	GAL-751	1S5F	GAL-104
	1SO0	GAL-500	1TLG	GAL-1
	1UAS	GAL-1501	1UGW	GAL-200
	1XC6	GAL-9011	1ZHJ	GAL-1
	2GAL	GAL-998		
Mannose	1BQP	MAN-402	1KZB	MAN-1501
	1KLF	MAN-1500	1KZC	MAN-1001
	1KX1	MAN-20	1KZE	MAN-1001
	1KZA	MAN-1001	1OP3	MAN-503
	1OUR	MAN-301	1QMO	MAN-302
	1U4J	MAN-1008	1U4J	MAN-1009

The positive dataset, composed of 80 non-redundant protein-hexose binding sites. The table lists the protein's PDB ID and the hexose ligand considered.

Table 2: **Protein binding-sites that bind non-hexose ligands.**

PDB ID	Cavity Center	Ligand	PDB ID	Cavity Center	Ligand
Hexose-like ligands					
1A8U	4320, 4323	BEZ-1	1AI7	6074, 6077	IPH-1
1AWB	4175, 4178	IPD-2	1DBN	pyranose ring	GAL-102
1EOB	3532, 3536	DHB-999	1F9G	5792, 5785, 5786	ASC-950
1G0H	4045, 4048	IPD-292	1JU4	4356, 4359	BEZ-1
1LBX	3941, 3944	IPD-295	1LBY	3944, 3939, 3941	F6P-295
1LIU	15441, 15436, 15438	FBP-580	1MOR	pyranose ring	G6P-609
1NCW	3406, 3409	BEZ-601	1P5D	pyranose ring	G1P-658
1T10	4366, 4361, 4363	F6P-1001	1U0F	pyranose ring	G6P-900
1UKB	2144, 2147	BEZ-1300	1X9I	pyranose ring	G6Q-600
1Y9G	4124, 4116, 4117	FRU-801	2B0C	pyranose ring	G1P-496
2B32	3941, 3944	IPH-401	4PBG	pyranose ring	BGP-469
Other ligands					
11AS	5132	ASN-1	11GS	1672, 1675	MES-3
1A0J	6985	BEN-246	1A42	2054, 2055	BZO-555
1A50	4939, 4940	FIP-270	1A53	2016, 2017	IGP-300
1AA1	4472, 4474	3PG-477	1AJN	6074, 6079	AAN-1
1AJS	3276, 3281	PLA-415	1AL8	2652	FMN-360
1B8A	7224	ATP-500	1BO5	7811	GOL-601
1BOB	2566	ACO-400	1D09	7246	PAL-1311
1EQY	3831	ATP-380	1IOL	2674, 2675	EST-400
1JTV	2136, 2137	TES-500	1KF6	16674, 16675	OAA-702
1RTK	3787, 3784	GBS-300	1TJ4	1947	SUC-1
1TVO	2857	FRZ-1001	1UK6	2142	PPI-1300
1W8N	4573, 4585	DAN-1649	1ZYU	1284, 1286	SKM-401
2D7S	3787	GLU-1008	2GAM	11955	NGA-502
3PCB	3421, 3424	3HB-550			

Non-hexose binding-sites negative dataset, composed of protein binding-sites that bind non-hexose ligands. The table lists the protein's PDB ID, the ligand considered and the specified cavity center. 22 ligands are similar to hexoses in shape and/or size. The cavity center is the centroid of the reported PDB atom numbers.

Table 3: **Non-binding sites negative dataset, composed of random surface pockets that do not bind any ligand.**

PDB ID	Cavity Center	PDB ID	Cavity Center	PDB ID	Cavity Center
1A04	1424, 2671	1A0I	1689, 799	1A22	2927
1AA7	579	1AF7	631, 1492	1AM2	1277
1ARO	154, 1663	1ATG	1751	1C3G	630, 888
1C3P	1089, 1576	1DXJ	867, 1498	1EVT	2149, 2229
1FI2	1493	1KLM	4373, 4113	1KWP	1212
1QZ7	3592, 2509	1YQZ	4458, 4269	1YVB	1546, 1814
1ZT9	1056, 1188	2A1K	2758, 3345	2AUP	2246
2BG9	14076, 8076	2C9Q	777	2CL3	123, 948
2DN2	749, 1006	2F1K	316, 642	2G50	26265, 31672
2G69	248, 378	2GRK	369, 380	2GSE	337, 10618
2GSH	6260				

Non-binding sites negative dataset, composed of random surface pockets that do not bind any ligand. The table lists the protein's PDB ID and the specified cavity center, computed as the centroid of the reported PDB atom numbers.

Table 4: **Background knowledge predicates for the two binding site representations**

Representation	Background Knowledge Predicates
atom-only	center_coords, has_atom, dist
amino acid	has_aminoacid, atom_to_center_dist, atom_to_atom_dist, diff_aminoacid

Table 5: 10**-folds cross-validation predictive accuracies for ProGolem using different recall selection methods**

	Recall selection method		
Fold	Primary sequence	Randomized	Domain-dependent
1	43.8%	56.3%	87.5%
2	62.5%	93.8%	78.5%
3	81.3%	87.5%	87.5%
4	56.3%	50.0%	43.8%
5	68.8%	68.8%	81.3%
6	37.5%	56.3%	81.3%
7	56.3%	62.5%	75.0%
8	68.8%	68.8%	81.3%
9	62.5%	81.3%	62.5%
10	56.3%	62.5%	68.8%
Mean	59.4%	68.8%	74.8%
Std Dev	12.6%	14.4%	13.4%

Table 6: 10-**folds cross-validation predictive accuracies for Aleph, ProGolem and SVM. The *1* besides Aleph and ProGolem stands for the *atom-only* representation and the *2* for the *amino acid* representation. SVM uses a different representation (see text).**

	Learning algorithm				
Fold	Aleph 1	ProG. 1	Aleph 2	ProG. 2	SVM
1	50.0%	75.0%	56.3%	75.0%	81.3%
2	68.8%	81.3%	68.8%	81.3%	87.5%
3	62.5%	68.8%	68.8%	93.8%	87.5%
4	50.0%	56.3%	68.8%	75.0%	75.0%
5	75.0%	81.3%	56.3%	81.3%	75.0%
6	68.8%	87.5%	81.3%	87.5%	87.5%
7	75.0%	81.3%	75.0%	81.3%	93.8%
8	93.8%	81.3%	75.0%	93.8%	87.5%
9	68.8%	75.0%	75.0%	81.3%	75.0%
10	56.3%	56.3%	87.5%	81.3%	62.5%
Mean	66.9%	74.4%	71.3%	83.2%	81.3%
Std Dev	13.2%	10.8%	9.8%	6.6%	9.3%