Mining Drug Resistance Relational Features with Hierarchical Multitask kFOIL

Elisa Cilia[1], Niels Landwehr[2], and Andrea Passerini[1]

[1] Dipartimento di Ingegneria e Scienza dell'Informazione
Università degli Studi di Trento, Italy
Email: {cilia,passerini}@disi.unitn.it
[2] Department of Computer Science, University of Potsdam, Germany
Email: landwehr@cs.uni-potsdam.de

Abstract. We introduce hierarchical kFOIL as a simple extension of the multitask kFOIL learning algorithm. The algorithm first learns a core logic representation common to all tasks, and then refines it by specialization on a per-task basis. The approach is applied to a HIV resistance mutation dataset in order to learn models of drug resistance for mutants. Experimental results show the advantage of the proposed algorithm over both single and multi task alternatives, and its potential usefulness in providing explanatory features for the domain.

1 Introduction

Multitask learning [1] deals with the problem of exploiting information on related tasks in order to improve predictive performances. Existing approaches rely on some type of parameter sharing among tasks, like learning a common hidden layer representation in multitask feed-forward neural networks [1], employing common priors for task dependent parameters in hierarchical Bayesian models [2], or both [3]. Most existing approaches assume that task models can be represented as parameter vectors over which to define prior distributions. 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. A notable example in this direction is the recent work by Deshpande et al [4] for learning multitask probabilistic planning rules from a common set of rule prototypes. Taking a discriminative viewpoint, kFOIL [5] greedily learns a relational kernel representation with high discriminant power for a certain task. The algorithm was recently [6] 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, and can be largely suboptimal if tasks relatedness is not high [7]. We propose here a simple extension where the common representation is viewed as an initial structure which is further specialized on a per-task basis.

This simple approach can be generalized to a deeper hierarchical process of refinements whenever a hierarchical clustering of the tasks is available or can be learned from data.

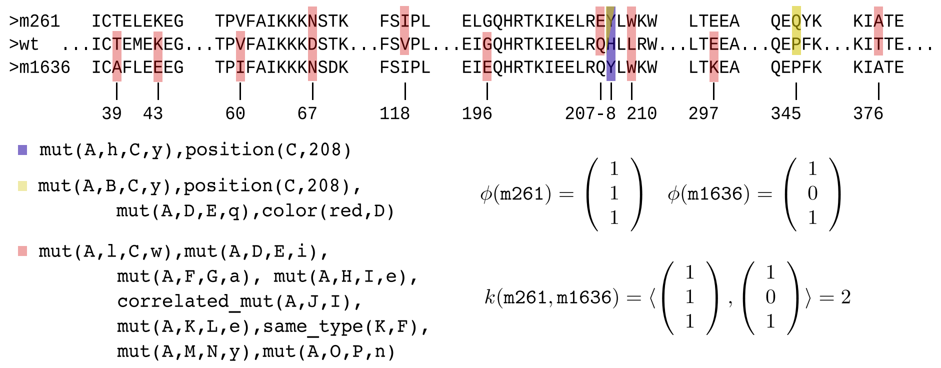
We applied our hierarchical kFOIL algorithm to a dataset of HIV resistance mutations [8]. The dataset reports wild type and mutations of reverse transcriptase, a viral protein which is essential for the success of the viral propagation. A mutation can confer the mutant resistance to one or more drugs, for instance by modifying their 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 to 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 because drugs are already grouped into classes in the processed data [8]. On the other hand, our hierarchical refinement approach succeeds in combining the advantages of the two methods, being always at least as good as the better of them and often significantly better than at least one. Nonetheless, our 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 learned can be inspected in order to distinguish between general features increasing the overall resistance of the mutant and drug-specific ones.

The rest of the paper is organized as follows: Section 2 briefly describes the original kFOIL formulation and its multitask version; section 3 introduces our extension for hierarchical multitask learning; section 4 describes the HIV dataset and the relational representation employed, while section 5 reports our experimental evaluation and discusses the models obtained. Conclusions are drawn in section 6.

2 kFOIL

kFOIL [5] is a statistical relational learner developed by adapting the FOIL [9] learning algorithm. It greedily learns a set of clauses in a general-to-specific fashion, where clauses are constructed one at a time starting with the empty one and refining it guided by a certain scoring measure. Instead of learning a logic program covering positive and not negative instances, kFOIL learns a logic kernel to be used within a kernel machine learning algorithm. The set of learned clauses defines a feature space representation for examples and a statistical learning algorithm is trained with such representation.

Figure 1 shows a sample of learned logic rules together to their feature space representation ϕ and kernel value k for two mutants from the HIV resistance mutation database. Each mutant is mapped into a vector with one entry for each clause, evaluating to one if the clause fires on the example (i.e. the clause



```
>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)
```

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

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

Fig. 1. Example of logic kernel for the HIV resistance mutation database. An alignment between the wildtype and two mutants is reported, with colors 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.

and the background knowledge logically entail it) and zero otherwise. A linear kernel in this representation amounts at counting the number of rules firing on both examples. For details on the logic representation employed see Section 4.1.

Algorithm 1 kFOIL algorithm.

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

The search procedure is sketched in Algorithm 1, where $\mathcal{D}$ is the training set, $\mathcal{B}$ the available background knowledge, p/n the target predicate, $\rho(c)$ a proper refinement operator, and S the scoring function measuring quality of candidate clauses. The input 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 refined version of the algorithm as detailed later on. Several scoring functions have been employed to guide the search, including 0-1 loss and area under the ROC curve. Both measures require that the statistical learner is trained for each candidate clause and its performance on the training set are reported. An efficient alternative consists in using kernel target alignment (KTA) [10], defined as:

$$A(K, y) = \frac{\langle K, yy^T \rangle_F}{\sqrt{\langle K, K \rangle_F \langle yy^T, yy^T \rangle_F}} \tag{1}$$

Algorithm 2 Hierarchical kFOIL algorithm.

```
1: procedure HierarchicalKFOIL($\{\mathcal{D}_1, \ldots, \mathcal{D}_k\}, \mathcal{B}$)
2:     Initialize $H_0 \leftarrow$ kFOIL($\emptyset, \{\mathcal{D}_1, \ldots, \mathcal{D}_k\}, \mathcal{B}$)    ▷ compute initial representation
3:     for all $\mathcal{D}_i \in \{\mathcal{D}_1, \ldots, \mathcal{D}_k\}$ do
4:         $H_i \leftarrow$ kFOILRefine($H_0, \mathcal{D}_i, \mathcal{B}$)    ▷ compute task-dependent refinements
5:     end for
6:     return $\{H_1, \ldots, H_k\}$
7: end procedure
```

where $y \in \{-1, 1\}^m$ 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.

kFOIL was recently adapted [6] to deal with multitask learning in a rather straightforward manner: the algorithm learns a common representation for all tasks, using a scoring measure which accounts for the usefulness of the clause on the different tasks, for instance by simply taking the average of the task-dependent scores.

3 Hierarchical kFOIL

Learning a representation which is common to multiple related tasks is a way to reduce the risk of overfitting the data [2]. 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. In this work we propose a simple extension to the multitask kFOIL learning algorithm dealing with this problem. A hierarchical approach to multitask learning is taken: the algorithm first learns a representation which is common to all tasks by using plain 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 2 shows the resulting hierarchical kFOIL learning system, where $\{\mathcal{D}_1, \ldots, \mathcal{D}_k\}$ are the datasets for the k different tasks. For simplicity we assumed a common background knowledge $\mathcal{B}$, but it is straightforward to replace it with task-specific background knowledges, as well as a task-specific language bias defining the clause refinement operator ρ.

The refinement stage is described in Algorithm 3. It is made of two steps: a clause refinement one where each clause from the initial representation is further refined guided by the task-specific score; a clause addition one where the model from the previous step is enlarged by creating novel clauses using the plain kFOIL procedure.

A detailed analysis of computational complexity for both single task and multitask kFOIL is reported in [6], showing the increase in efficiency of the

Algorithm 3 kFOIL refinement algorithm.

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

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. We postpone a deeper comparison to an extended version of the paper.

The hierarchical kFOIL algorithm can be easily generalized to multiple levels of refinements, whenever tasks can be naturally aggregated into a hierarchical clustering structure. When the existence of a clustering 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 [11].

4 The HIV Resistance Mutation Dataset

We applied the hierarchical kFOIL algorithm to a dataset of mutations from the Los Alamos National Laboratories (LANL) HIV resistance database[3]. The dataset was derived in [8] and is composed of 2,339 mutants of the HIV reverse transcriptase (RT). RT is a 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 thus crucial for virus propagation. Viruses typically have a very high mutation rate, and mutants in the dataset have a number of mutations ranging from 6 to 90 each, with an average of 32. These characteristics make the HIV RT mutant dataset suitable for the prediction of mutant resistance as in [12]. Some of the observed mutations confer resistance to one or more drugs or drug classes. Richter et al. [8] 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 mutation to that drug. We selected the three classes of drugs: (a) NonNucleoside RT Inhibitors (NNRTI); (b) NonCompetitive RT inhibitors (NCRTI); (c)

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

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 other inhibitors like the Nucleoside RT Inhibitors (NRTI) since all mutants had at least one mutation conferring resistence to them.

4.1 Background Knowledge

We built a relational knowledge base for the domain at hand. Table 1 summarizes the predicates we 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 characterizing them (`mut/4`). Target predicates were encoded as resistance of the mutant to a certain drug (`res_against/2`).

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 a residue 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 1. Summary of the background knowledge facts and rules.

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

`color/2` indicates the type of the natural amino acids according to the coloring proposed in [13]. 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 aminoacid.

`same_type_mut/2` indicates that a residue substitution at a certain position does not modify the aminoacid type with respect to the wild type. For example mutation d123e conserves the aminoacid 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.

5 Experimental Evaluation

We evaluated our hierarchical kFOIL algorithm on the multitask RT mutant resistance problem, with three tasks corresponding to the three drug classes (see Section 4). We compared the results 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.

5.1 Experimental Setting

We performed a 3-fold stratified cross-validation to ensure a good balancing between positive and negative examples for each learning task. In the experiments the KTA (see Equation 1) has been used as scoring function for guiding the search of the kFOIL algorithm, as it is more efficient even if less effective [6] in general than measures based on a trained kernel machine. A simple linear kernel was employed in order to maximize the understandability of the learned models. Note that we are not interested in pushing the performance of the learning algorithm by fine tuning its parameters but rather comparing the respective advantages of the different approaches and evaluate their explanatory power. We employed area under the ROC curve (AUC_{ROC}) and area under the recall/precision curve (AUC_{RP}) as measures of performance in all experiments.

5.2 Experimental Results

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,Rold,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 pointwise mutations, as for the association rules discovered in [8]. 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 2 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. We conjecture that a different behaviour could be expected when considering as related tasks specific drugs belonging to the same class. By adding a refinement stage on a per-task basis, we succeed in improving the results with respect to both single task 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. We are currently investigating the usefulness of a deeper hierarchical structure obtained by further splitting drug classes into specific drugs.

	Classes	NNRTI		NCRTI		PARTI	
	CV Exp	AUC_{ROC}	AUC_{RP}	AUC_{ROC}	AUC_{RP}	AUC_{ROC}	AUC_{RP}
	single task	0.95○	0.97	0.95	0.93	0.81	0.38
V1	multitask	0.77	0.82	0.99	0.84	0.95●	0.62
	hierarchical	0.96○	0.97	0.99●	0.92	0.98●	0.70
	single task	0.68	0.72	0.88	0.79	0.65	0.41
V2	multitask	0.66	0.71	0.86	0.57	0.84●	0.64
	hierarchical	0.71○●	0.76	0.88	0.70	0.90●	0.57

Table 2. Summary of the cross-validation experiments (kta scoring). Statistical tests for the significance of the differences in AUC_{ROC} where computed for the methods within each language bias variant using the two-tailed Hanley-McNeil test [14] (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 in AUC_{ROC} are not statistically significant.

Concerning the language bias variants, we can observe from Table 2 that as expected the instantiation of a specific position (V1) gives better results with respect to searching more general rules (V2). The models generated in the V1 case closely reflect the specific mutations and the fact that mutants resistant to the same drug share mutations located in the same positions along the primary sequence. These positions could be located in the protein binding site or its vicinity, but drug resistance could also be conferred more indirectly by other conformational modifications.

For example the clause `mut(A,g,196,B),color(blue,B)` belongs to the model learned for variant V1 in the multitask learning setting and was found in 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 a research direction on an amidase. When inspecting the models resulting from the single task refinement we found that for the NNRTI task the up mentioned clause is not extended, while for the NCRTI and the PARTI tasks the clause is extended with other mutations. This could suggest the fact that in those cases the drug-specific resistance results from the combination with one or two other mutations.

kFOIL successfully identified a known correlation between the mutations at positions 215 and 41, which was previously observed to be related to resistance to NNRTI [15].

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 the fact 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 learned 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 quite large set of mutations, with the latter clause including an explicitly correlated pairs 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 completely changing the type of aminoacid in the mutated position. Note that the need for multiple mutations in order to induce a change in the phenotype has recently found confirmation [16] in experimental studies on molecular phenotypes. A suggested interpretation [16] 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 our algorithm, the obvious limitation of learning techniques focusing on single point mutations alone is an additional stimulus for this research direction.

6 Conclusions

We developed hierarchical kFOIL as a simple 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 and successively refining a common model on a per-task basis. We applied the algorithm to a dataset of HIV resistance mutations, showing its advantage over both single and multi task alternatives. We stress here that a major advantage of the adopted strategy is the ability to provide explanations for the learned 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 approach can be generalized to deeper levels of task structure relying on hierarchical task clustering.

Acknowledgment

We would like to thank Lothar Richter, Regina Augustin and Stefan Kramer for sharing the processed version of the HIV resistance mutation database.

References

1. Caruana, R.: Multitask learning. Machine Learning **28**(1) (1997) 41–75
2. Baxter, J.: A bayesian/information theoretic model of learning to learn via multiple task sampling. Machine Learning (1997) 7–39
3. Bakker, B., Heskes, T.: Task clustering and gating for bayesian multitask learning. Journal of Machine Learning Research **4** (2003) 83–99
4. Deshpande, A., Milch, B., Zettlemoyer, L.S., Kaelbling, L.P.: Learning probabilistic relational dynamics for multiple tasks. In: Proc. 23rd Conference on Uncertainty in Artificial Intelligence (UAI). (2007) 83–92
5. Landwehr, N., Passerini, A., De Raedt, L., Frasconi, P.: kfoil: learning simple relational kernels. In: Proc. of AAAI'06. (2006) 389–394
6. Landwehr, N., Passerini, A., De Raedt, L., Frasconi, P.: Fast learning of relational kernels. conditionally accepted at Machine Learning (2009)
7. Madrid-Sanchez, J., Parrado-Hernandez, E., Figueiras-Vidal, A.: Selective multitask learning by coupling common and private representations. In: NIPS 08 Workshop on Learning from Multiple Sources. (2008)
8. Richter, L., Augustin, R., Kramer, S.: Finding relational associations in hiv resistance mutation data. In: Proc. of ILP'09. (2009)
9. Quinlan, J.: Learning Logical Definitions from Relations. Machine Learning **5** (1990) 239–266
10. Cristianini, N., Shawe-Taylor, J., Elisseef, A., Kandola, J.: On kernel-target alignment. In: NIPS 14. (2001)
11. Thrun, S., O'Sullivan, J.: Discovering structure in multiple learning tasks: The TC algorithm. In: Proc. of IMCL'96. (1996)
12. Rhee, S., Taylor, J., Wadhera, G., Ben-Hur, A.: Genotypic predictors of human immunodeficiency virus type 1 drug resistance. Proceedings of the National Academy of Sciences (Jan 2006)
13. Taylor, W.R.: The classification of amino acid conservation. J Theor Biol **119**(2) (March 1986) 205–218
14. Hanley, J., McNeil, B.: A method of comparing the areas under receiver operating characteristic curves derived from the same cases. Radiology **148**(3) (1983) 839–843
15. Lengauer, T., Sing, T.: Bioinformatics-assisted anti-hiv therapy. Nature Reviews Microbiology **4**(10) (2006) 790–797
16. Wagner, A.: Neutralism and selectionism: a network-based reconciliation. Nature reviews. Genetics **9**(12) (December 2008) 965–974