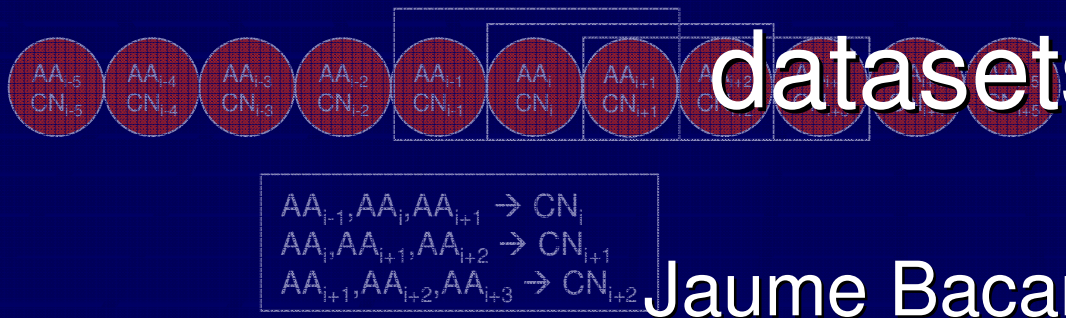
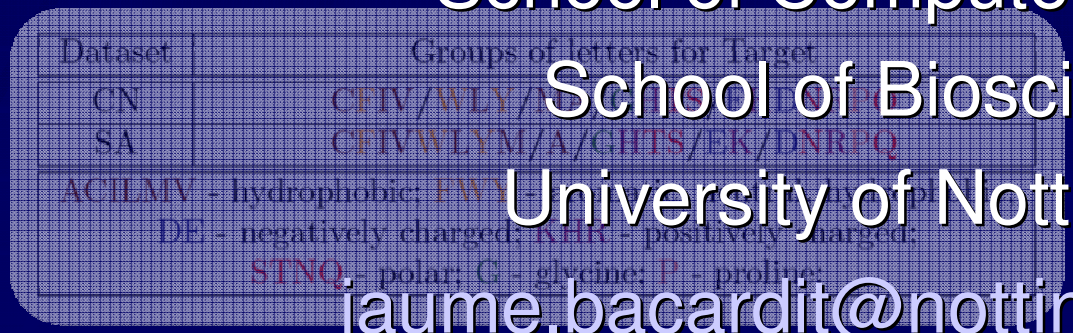
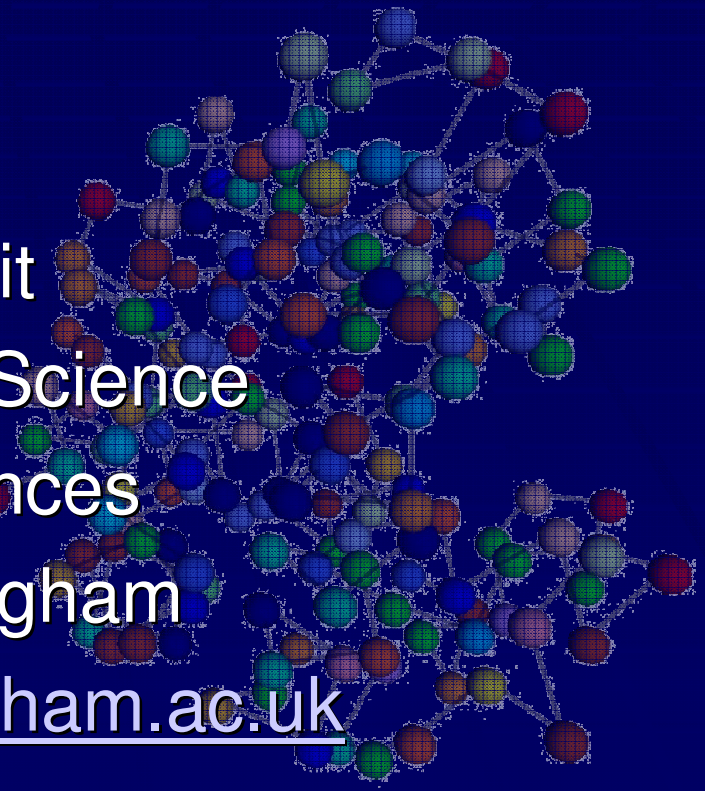
GBML methods for large-scale datasets





Jaume Bacardit
School of Computer Science
School of Biosciences
University of Nottingham
jaume.bacardit@nottingham.ac.uk



asap automated scheduling optimisation & planning research

Multi-disciplinary Centre for Integrative Biology

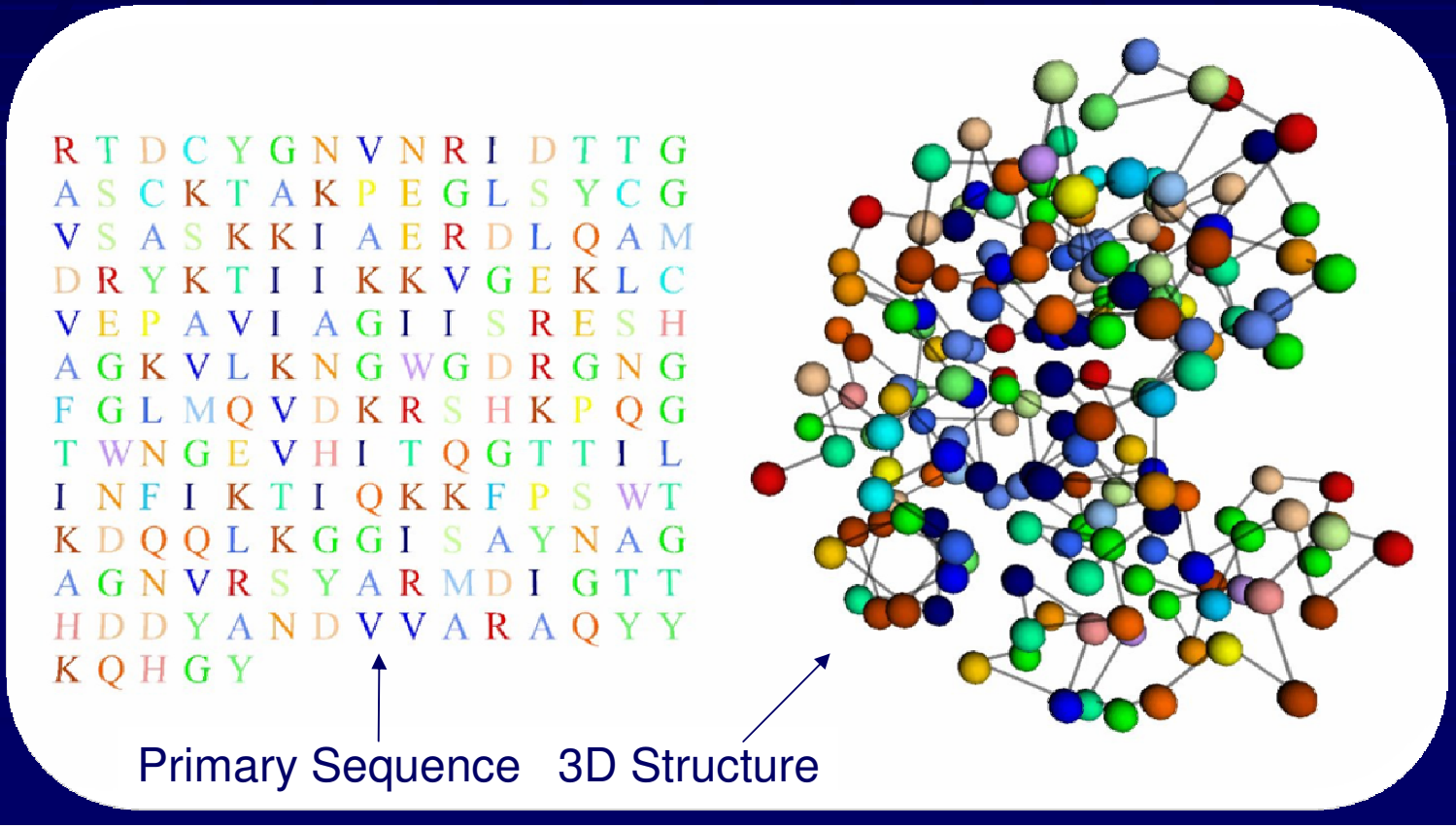
The University of Nottingham

Outline

- Brief introduction to the application domain
- Initial results
- Scalability and performance improvements
  - The BioHEL GBML system
  - Ensemble learning
  - The attribute list knowledge representation
- Interpretability of the solutions
- What lies in the future

Protein Structure Prediction

- Protein Structure Prediction (PSP) aims to predict the 3D structure of a protein based on its primary sequence

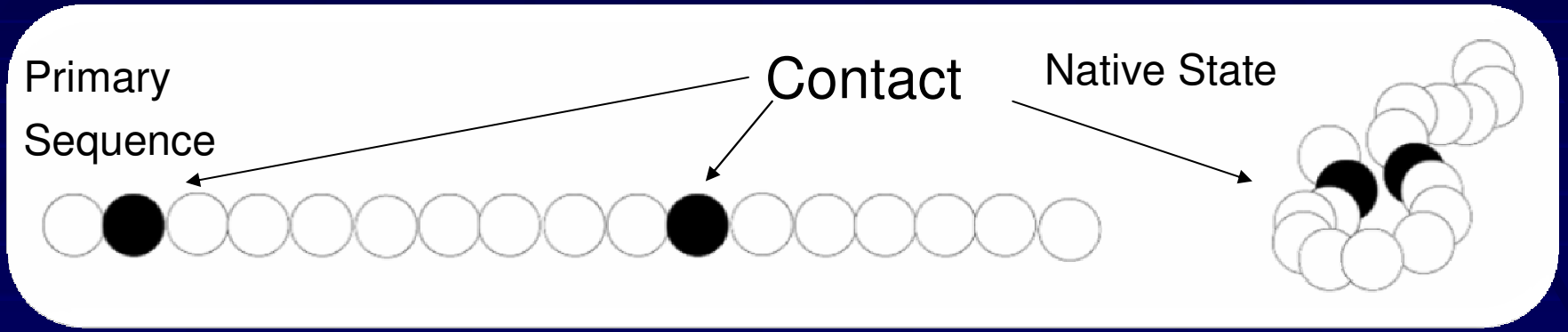


Protein Structure Prediction

- Beside the overall 3D PSP, we can predict several structural aspects of protein residues
  - Coordination number
  - Solvent accessibility
  - Secondary structure
  - Disulfide bonding
- Accurate prediction of these features can help PSP in many ways
  - Constraining the conformation space
  - Identifying better homolog proteins
- These predictions can help research in other areas, beside the main PSP problem
  - Surface prediction
  - Functional prediction

Coordination Number

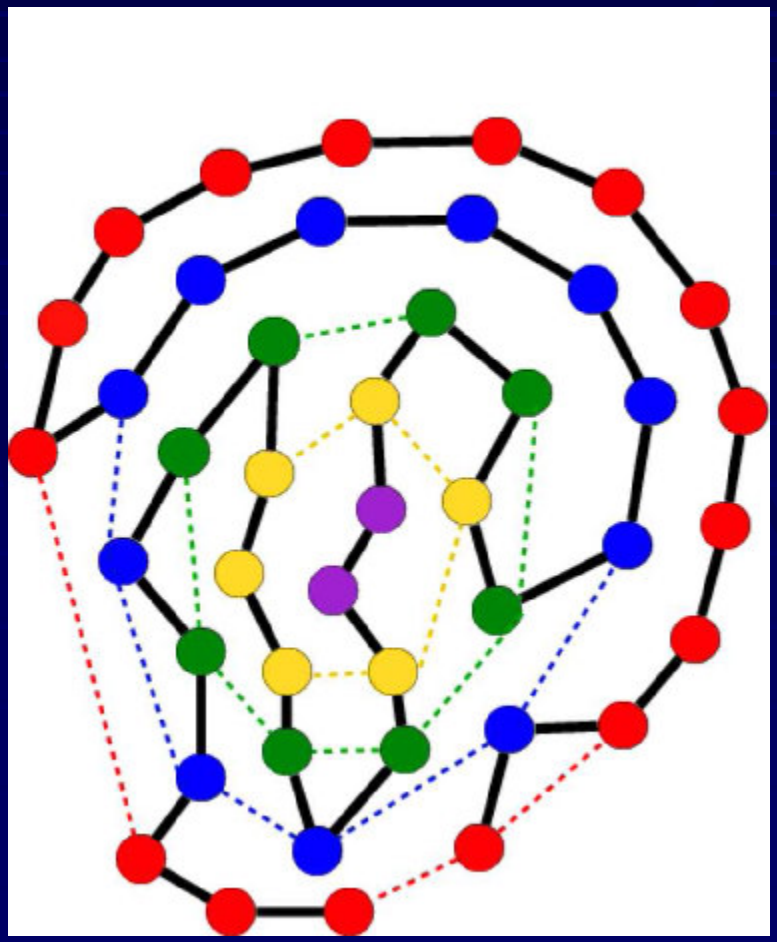
- Two residues of a chain are said to be in contact if their distance is less than a certain threshold



- **CN of a residue :** count of contacts of a residue
- CN gives us a simplified profile of the density of packing of the protein

Recursive Convex Hull

- Structural feature that we have proposed recently [Stout, Bacardit, Hirst & Krasnogor, Bioinformatics 2008 24(7):916-923;]
- We model a protein as an onion, assigning each residue to a different layer of the onion
- Strictly speaking each layer is a convex hull of points
- The convex hull of a point set is a metric easy and fast to compute
- Recursive Convex Hull is computed by iteratively identifying the layers (hulls) of a protein

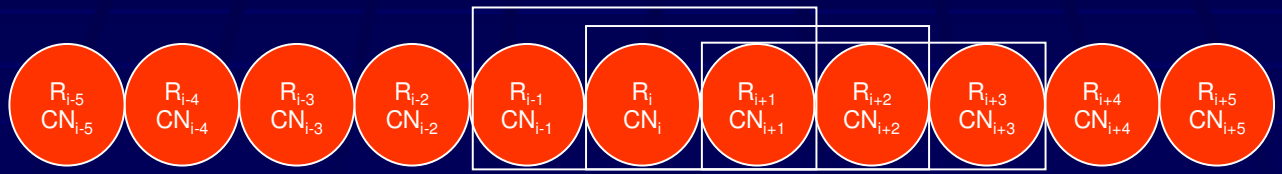


How to predict these features?

- Two dimensions to decide
  - Inputs: What input information (derived from the protein primary sequence) is used?
  - Outputs: How are we modelling the feature that we are predicting?
    - Predicting the actual (continuous) feature
    - Predicting, for instance, buried or exposed
      - Discretization is applied to the original feature, dividing it into 2, 3 or 5 states

Input information

- Two types of input information
  - Local information: From the target residue and its closest neighbours in the chain



$R_{i-1}, R_i, R_{i+1} \rightarrow CN_i$
$R_i, R_{i+1}, R_{i+2} \rightarrow CN_{i+1}$
$R_{i+1}, R_{i+2}, R_{i+3} \rightarrow CN_{i+2}$

  - Global information: From the whole chain we are predicting

Size of the problem

- Dataset that we have used for the last three years
  - 1050 protein chains
  - ~260000 instances
  - Depending on the representation, hundreds of continuous attributes

Initial results

- GAssist parameters
  - 150 strata for the ILAS windowing system
  - 20000 iterations
  - Majority class will be used by the default rule

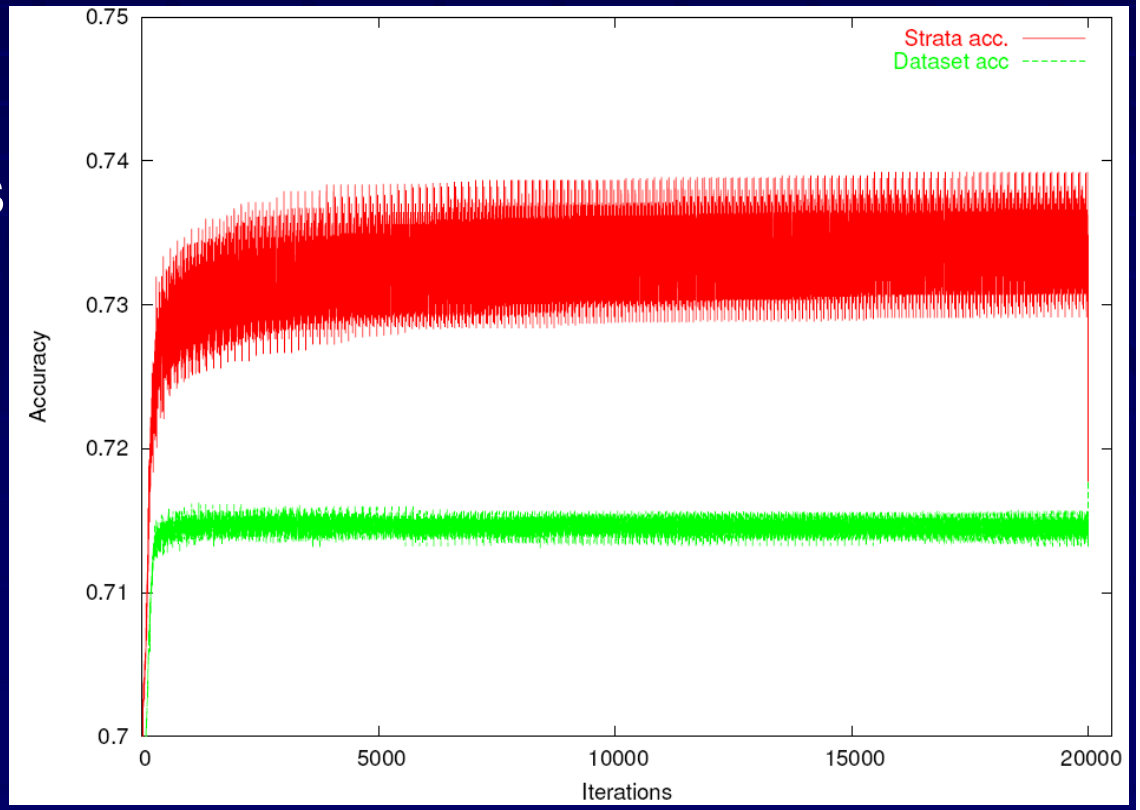
Initial results

- Summary of results of CN prediction from [Bacardit et al., 06]
- Global ranking of performance:

  LIBSVM → GAssist → Naive Bayes → C4.5
- CN prediction accuracy
  - 2 states: ~80%
  - 3 states: ~67%
  - 5 states: ~47%
- SVM obtained better results, but it was slower than GAssist, and test could take hours, as 70-90% of training instances were used as support vectors

Initial results

- Is GAssist learning?
  - Evolution of the training accuracy of the best solution
    - The knowledge learnt by the different strata does not integrate successfully into a single solution



Scalability and performance improvements

- Improvement of the learning system:
  - The BioHEL GBML system
- Improvement wrapped over the learning process
  - Ensemble learning
- Improvement within the learning system
  - The attribute list knowledge representation

THE BIOHEL GMBL SYSTEM

The BioHEL GMBL system

- BIOinformatics-oriented Hiearchical Evolutionary Learning – BioHEL [Bacardit et al., 07]
- GAssist had troubles evolving rule sets of more than ~20 rules for real datasets
- To overcome this kind of scalability limitations, let's learn one rule at a time
- Iterative Rule Learning
  - First used in EC in Venturini's SIA system
  - Widely used for both Fuzzy and non-fuzzy evolutionary learning

Iterative Rule Learning

- IRL has been used for many years in the ML community, with the name of separate-and-conquer

Separate-and-conquer algorithm
Input : $Examples$
$Theory = \emptyset$
While $Examples \neq \emptyset$
    $Rule = \text{FindBestRule}(Examples)$
    $Covered = \text{Cover}(Rule, Examples)$
    If $\text{RuleStoppingCriterion}(Rule, Theory, Examples)$
        Exit while
    EndIf
    $Examples = Examples \setminus Covered$
    $Theory = Theory \cup Rule$
EndWhile
Output : $Theory$

BioHEL characteristics

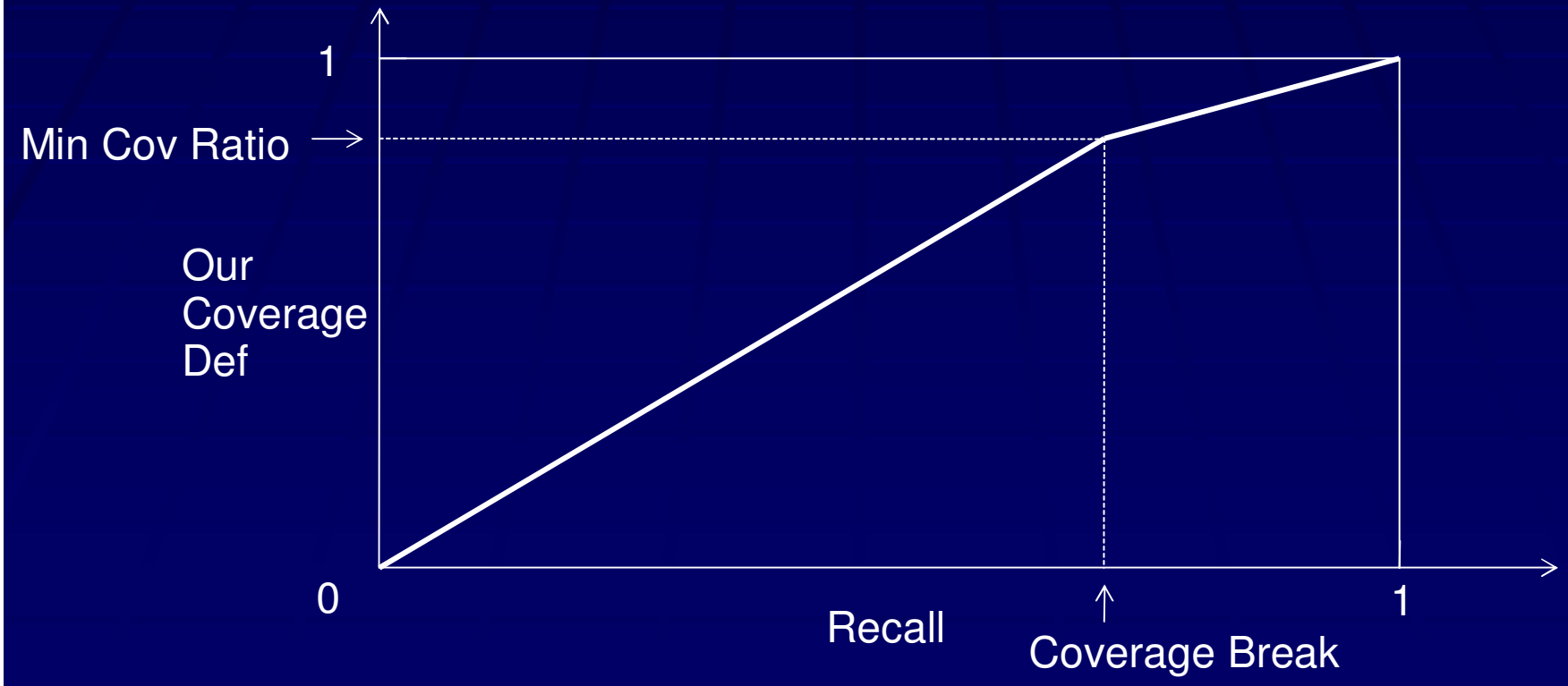
- The MDL-based fitness function, the ILAS windowing scheme and the explicit default rule mechanism of GAssist are used
- Iterative process terminates when there it is impossible to evolve a rule where the rule class is the majority class among the matched examples
- At that point, all remaining training instances are assigned to the default class

BioHEL fitness function

- The fitness function of an IRL system has two objectives to maximize
  - Evolving accurate rules
  - Evolving high coverage rules
- In very noisy domains (like the PSP ones), the equilibrium between these two objectives is difficult
- It is impossible to obtain accurate, high coverage rules, so easier path for evolutionary search is to maximize accuracy at the expense of coverage

BioHEL fitness function

- Fitness = TP/(TP+FP) + Rule coverage
- Coverage term penalizes rules that do not cover a minimum percentage of examples



BioHEL fitness function

- Coverage term
  - 2 parameters: Coverage Break and Minimum Coverage Ratio
  - As a raw coverage function we will use recall TP/(TP+FN)
  - In this way function is not affected by problems with unequal class distribution
  - Coverage breaks for different problem classes are adjusted automatically from a general coverage break given the class distribution

ENSEMBLE LEARNING

Ensembles

- Ensemble learning is a quite well established family of techniques that provides performance boost and robustness to the learning process
- In general these techniques integrate the collective predictions of a set of models in some principled fashion
- We have integrated some simple ensemble methods with our LCS methods [Bacardit & Krasnogor, 06]

Ensemble for consensus prediction

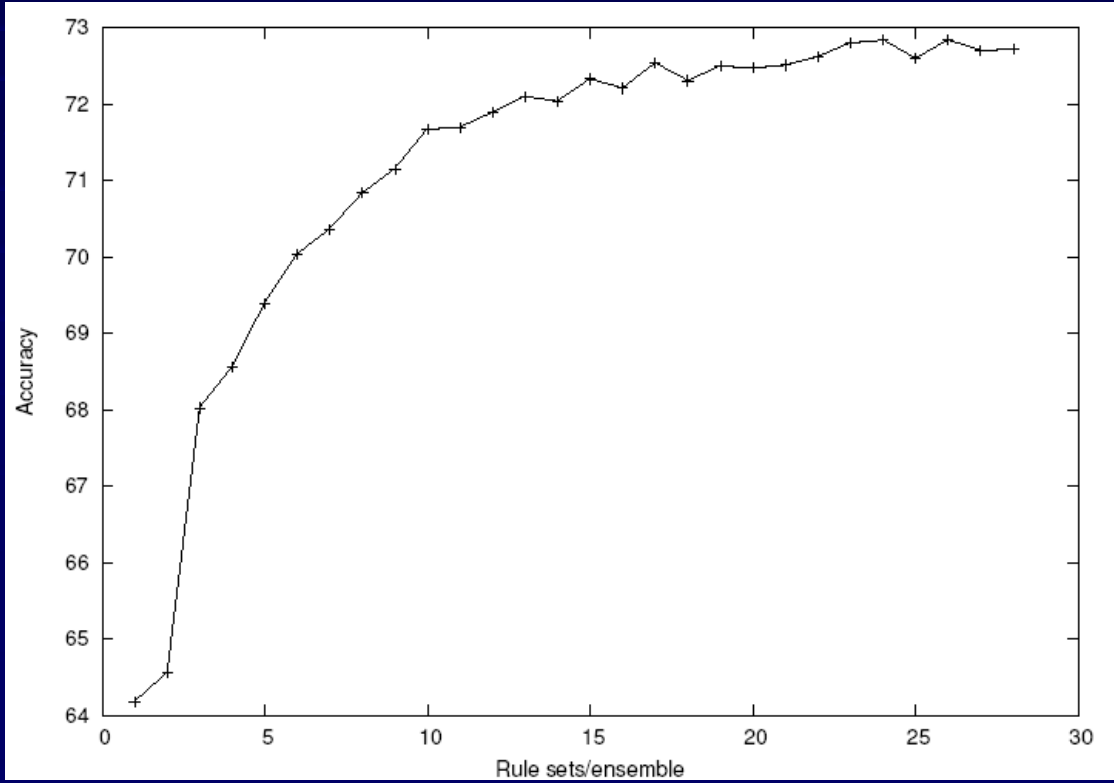
- Technique inspired in bagging: combining models trained with slighly different views of the dataset using a flat consensus voting
- Our approach is slightly different:
  - The dataset used to generate each model is the same
  - However, different random seeds have been used for each model

Ensemble for consensus prediction

1. Gassist/BioHEL is run N times on the original training set, each of them with a different random seed
2. From each of the N runs, a rule set is extracted: the best individual at the end of the training process
3. Exploitation stage: for each new instance, the N models produce a prediction. The majority class is used as the ensemble prediction

Ensemble for consensus prediction

- Prediction of Secondary Structure
- Accuracy increased by almost 9% with ~25 rule sets



Ensemble for ordinal classification

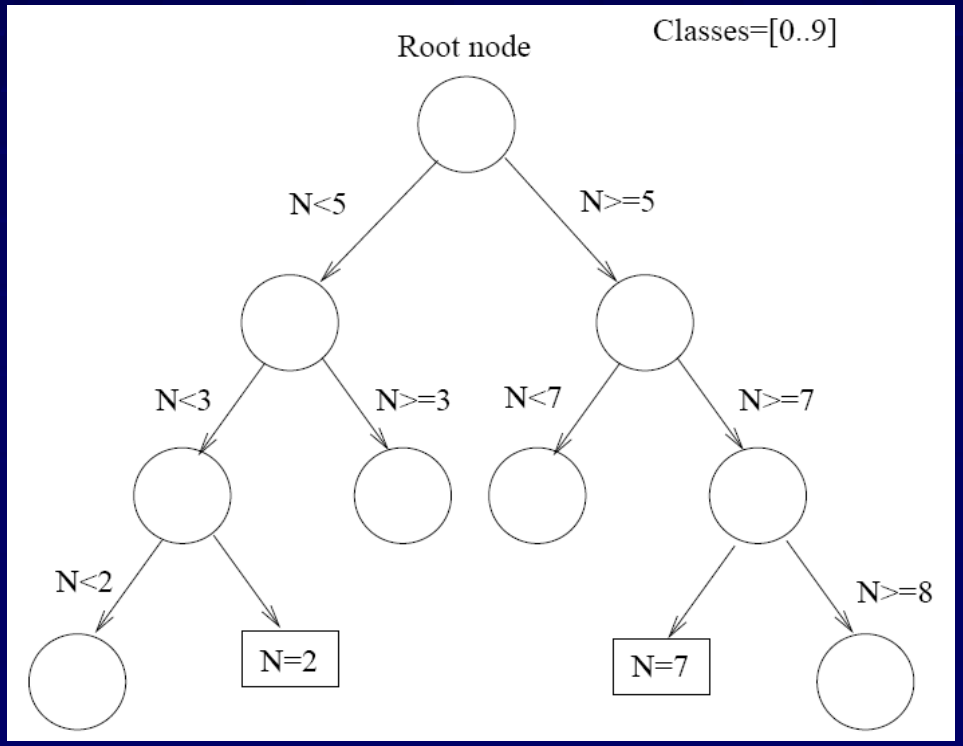
- Motivation
  - In general it can be difficult to learn datasets with high number of classes
  - In the case of ordinal datasets, it is important that the prediction errors stay local, i.e., predicting class 2 for an instances of class 1, instead of class 10

Ensemble for ordinal classification

- As it is usual in these kind of approaches, the original dataset is decomposed into several simpler datasets, usually only with 2 classes, exploting the ordered nature of the classes
- The hierarchical ensemble has two main parts
  - Criterion for decomposing the dataset
  - Integration of the binary predictions into a final N classes prediction

Ensemble for ordinal classification

- Criterion for decomposing the dataset
  - Cut points always tries to balance number of instances at each branch of the tree



Ensemble for ordinal classification

- Integration of the binary predictions into a final N classes prediction
  1. New instances query the binary classification model at the root of the tree
  2. If model predicts class 0, next step is to query the model in the left branch of the root node
  3. Otherways, query the model in the right branch of the root node
  4. Process is repeated until a leaf has been reached
- Models in each node of the tree are an ensemble itself, a consensus prediction.

THE ATTRIBUTE LIST KNOWLEDGE REPRESENTATION

Motivation

- Learning from datasets with hundreds of attributes (PSSM representation) is extremely slow
- Even by using parallel implementations and SSE code [Llora et al., 08] for match operations
- Can we find an alternative way of reducing this cost?

Motivation II

- Example of a rule for predicting Secondary Structure:
  - Att $Leu_{-2} \in [-0.51,7]$ and $Glu \in [0.19,8]$ and $Asp_{+1} \in [-5.01,2.67]$ and $Met_{+1} \in [-3.98,10]$ and $Pro_{+2} \in [-7,-4.02]$ and $Pro_{+3} \in [-7,-1.89]$ and $Trp_{+3} \in [-8,13]$ and $Glu_{+4} \in [0.70,5.52]$ and $Lys_{+4} \in [-0.43,4.94]$ → alpha
  - 9 attributes out of 300 were expressed in the rule
  - This means that while evolving this rule 291 out of the 300 attribute match operations were irrelevant

The representation

- What if we only keep in the rule the relevant intervals? → The attribute list knowledge representation [Bacardit & Krasnogor, 08]
- In this way match operations will be much faster in domains with high number of attributes
- Also, as we only keep the intervals for the relevant attributes, the representation can potentially explore the search space more efficiently → learning better

The representation

- Each rule has….

#Expr. Atts.	4							
Expr. Atts.	1	3	4	7				
Intervals	L_1	U_1	L_3	U_3	L_4	U_4	L_7	U_7
Class	C_1							

How to identify the relevant attributes?

- Initialization will randomly express a subset of attributes for each rule in the population
- Parameter defines the expected value of number of expressed attributes per rule
- Afterwards, two operators will be applied to the population after mutation to
  - Add attributes to the list (specialize)
  - Remove attributes from the list (generalize)
  - As in ECL [Divina et al., 03]
- Operators will have a probability of application

Experiments

- Goals
  - Validate if the representation can *indeed* identify the relevant features
  - Sensitivity to the probability of generalize and specialize (from 5% to 25%)
  - Check if the representation can actually learn better, as we have hypothesized
  - Compared to the NAX representation [Llorà et al., 07] that uses SSE vectorial instructions

Experiments

- Datasets
  - 16 datasets from UCI repository
    - Various sizes in #inst and #atts. Used to verify if representation can learn properly
  - 2 PSP datasets
    - To check the full power of the representation
    - CN prediction (Kinjo dataset) [Cuff & Barton, 99]
      - ~260000 instances, 180 attributes
    - SS prediction (CB513 dataset) [Wood & Hirst., 05]
      - ~90000 instances, 300 attributes

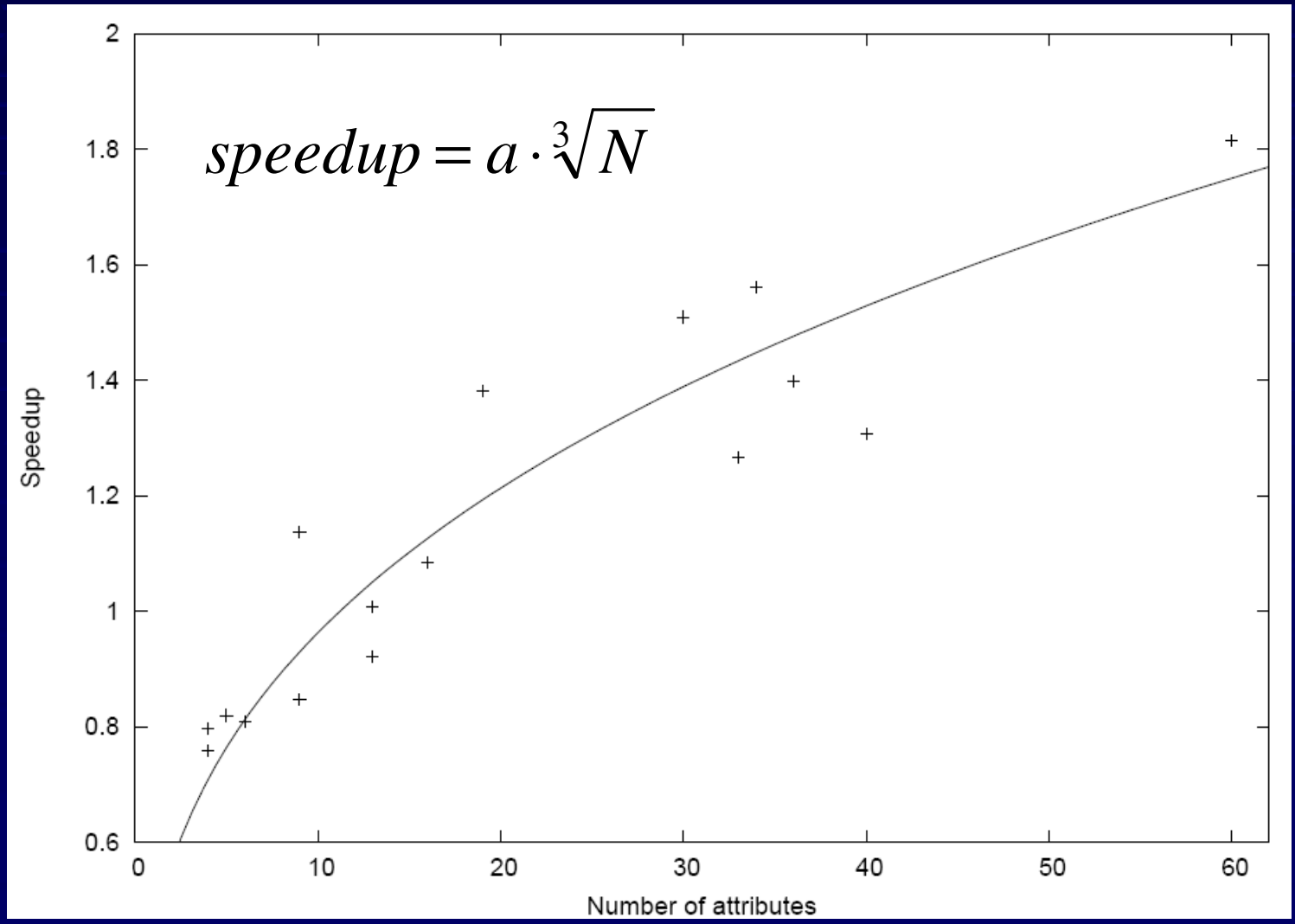
Accuracy results

Dataset	NAX	Prob. of Generalize and Specialize in Att. List KR				
		0.05	0.10	0.10	0.20	0.25
bal	88.0±3.2	87.4±3.9	88.2±3.6	88.7±3.4	87.7±4.4	88.2±4.1
bpa	68.9±7.2	69.5±7.5	70.0±7.2	69.7±6.3	68.6±8.4	69.3±7.9
gls	74.0±10.4	75.0±8.2	74.4±8.5	75.4±9.4	76.2±7.9	77.8±7.5
h-s	78.9±8.7	78.5±6.4	79.3±9.1	79.0±8.7	77.9±7.9	77.9±7.4
ion	93.1±4.7	93.0±3.8	92.7±4.5	92.5±4.2	92.0±4.3	91.6±4.0
irs	94.4±4.6	93.6±4.7	93.1±5.0	93.6±4.7	93.8±4.5	94.0±4.7
pen	95.2±0.8	94.8±1.0	94.8±1.0	94.6±0.9	94.0±1.1	94.0±1.2
sat	88.5±1.3	88.5±1.2	88.2±1.0	88.5±1.2	88.5±1.0	88.4±1.1
seg	97.1±0.8	97.0±1.0	97.0±0.8	97.0±0.9	97.2±0.8	97.1±0.8
son	81.3±9.5	82.3±9.0	82.7±9.2	81.4±11.6	82.9±7.2	83.7±7.3
thy	94.3±5.3	93.0±4.6	94.1±4.2	93.2±5.1	93.0±4.2	93.5±5.2
wav	84.3±1.7	84.7±1.4	84.6±1.5	85.1±1.6	84.6±1.6	84.7±1.7
wbcd	95.3±2.6	95.5±2.3	95.4±2.6	95.5±2.1	95.5±2.4	95.4±2.5
wdbc	95.9±2.7	95.8±2.6	96.4±2.9	96.2±2.5	96.1±2.9	96.0±2.5
wine	93.0±6.5	92.3±6.5	92.4±6.8	91.5±5.5	92.3±6.3	92.4±6.9
wpbc	78.5±7.5	78.4±7.8	78.5±8.1	77.3±6.5	77.4±7.9	78.4±7.5
Ave.	87.5±8.6	87.5±8.2	87.6±8.2	87.5±8.2	87.4±8.3	87.7±8.0

Run-time results

Dataset	NAX	Prob. of Generalize and Specialize in Att. List KR				
		0.05	0.10	0.10	0.20	0.25
bal	9.4±0.5	12.5±0.6	12.4±0.7	12.4±0.6	11.9±0.6	11.9±0.6
bpa	6.0±0.3	7.3±0.4	7.4±0.4	7.4±0.4	7.3±0.4	7.4±0.4
gls	4.5±0.3	5.5±0.4	5.4±0.4	5.4±0.4	5.2±0.3	5.3±0.3
h-s	4.4±0.3	5.0±0.4	4.8±0.3	4.7±0.3	4.5±0.3	4.5±0.3
ion	3.7±0.5	2.5±0.3	2.5±0.3	2.4±0.3	2.4±0.3	2.4±0.3
irs	1.3±0.2	1.6±0.2	1.6±0.2	1.6±0.2	1.6±0.2	1.6±0.2
pen	174.0±13.7	179.2±15.0	167.1±13.6	160.4±13.5	152.0±12.3	146.4±11.8
sat	110.3±6.1	81.3±3.9	79.6±3.6	78.9±3.5	75.4±3.5	76.2±3.4
seg	24.9±1.5	20.4±1.3	19.0±1.2	18.0±1.1	17.3±1.0	16.9±1.0
son	4.4±0.3	2.5±0.2	2.5±0.2	2.4±0.2	2.4±0.2	2.4±0.2
thy	1.6±0.2	2.0±0.2	1.9±0.2	2.0±0.2	1.9±0.2	2.0±0.2
wav	105.1±1.5	80.1±1.3	80.4±1.3	80.4±1.3	79.6±1.3	78.9±1.3
wbcd	3.6±0.4	3.8±0.5	3.4±0.5	3.1±0.4	3.0±0.4	2.9±0.4
wdbc	4.9±0.3	3.4±0.3	3.3±0.3	3.3±0.3	3.2±0.2	3.2±0.3
wine	1.2±0.1	1.3±0.1	1.2±0.1	1.2±0.1	1.2±0.1	1.2±0.1
wpbc	3.7±0.3	3.0±0.3	3.0±0.3	2.9±0.3	2.8±0.3	2.9±0.3
Ave.	28.9±50.6	25.7±47.0	24.7±44.5	24.2±43.1	23.2±41.1	22.9±40.1

Speedup over NAX



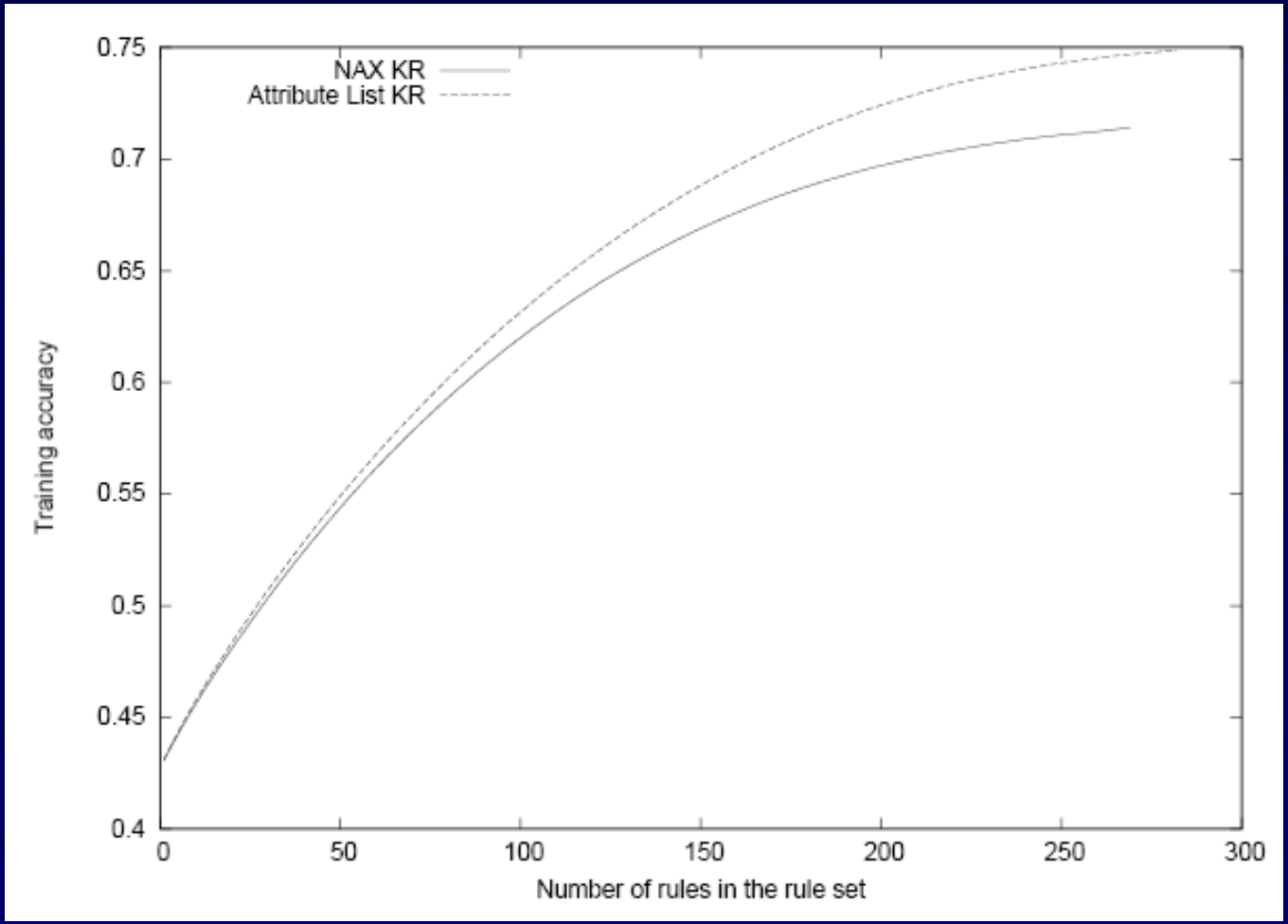
Bioinformatics datasets

- In these datasets, the new representation learns better and is much faster

Dataset	Result	NAX	Prob. of Generalize and Specialize in Att. List KR				
			0.05	0.10	0.10	0.20	0.25
SS	Acc.	72.4±1.0	73.3±0.8	73.4±0.9	73.3±0.8	73.3±0.8	73.2±0.7
	#rules	268.7±13.6	290.9±10.4	281.6±10.3	271.4±10.3	263.4±7.8	253.3±9.1
	#exp. att.	13.1±3.0	14.6±3.2	14.4±3.2	14.1±3.2	13.7±3.2	13.4±3.2
	run-time (h)	16.1±0.9	6.4±0.4	6.0±0.6	5.9±0.6	5.7±0.4	5.6±0.4
CN	Acc.	80.9±0.4	81.1±0.4	81.1±0.4	81.1±0.4	81.0±0.4	81.0±0.4
	#rules	263.2±12.6	284.7±12.5	275.1±13.3	265.5±13.4	255.5±11.2	245.1±11.8
	#exp. att.	14.3±2.9	16.3±3.0	16.1±3.1	15.7±3.1	15.2±3.1	14.8±3.1
	run-time (h)	45.7±2.5	30.9±2.1	29.8±2.3	28.9±2.3	28.1±1.8	26.7±2.0

Bioinformatics datasets

- Better learning process



Interpretability in PSP

- To interpret our rules, we use standard physico-chemical properties of amino acids
  - Hydrophobic AA: ACFGHIKLMSTVWY
  - Polar AA: CDEHKNQRSTWY
  - Charged AA: DEHKR
    - Positively Charged: HKR
    - Negatively Charged: DE
  - Small: ACDGPNST
  - …..

Examples of rules

- Example of a rule set for predicting high(1)/low(0) CN

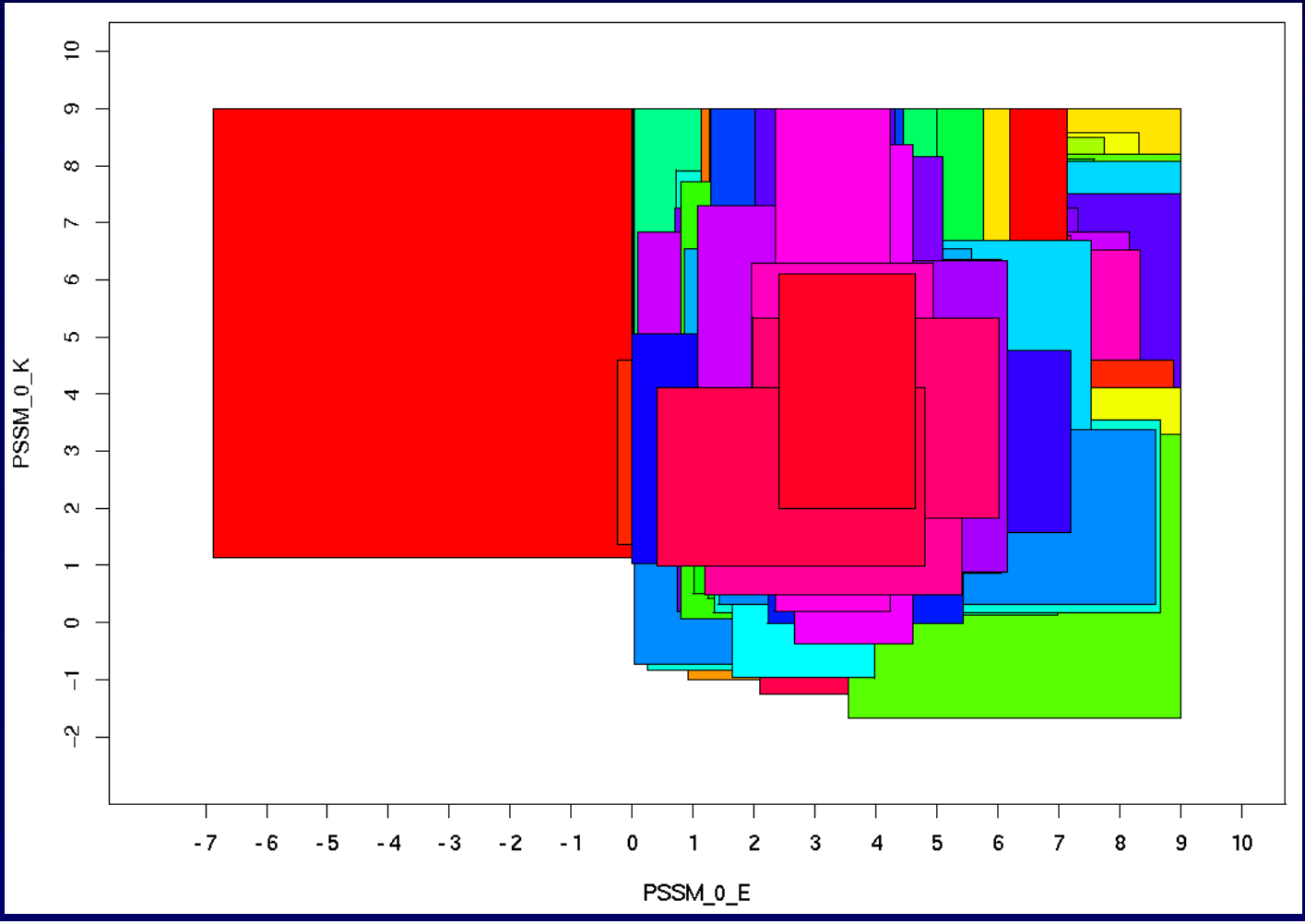
1. If $AA_{-4} \notin \{X\}$ and $AA_{-3} \notin \{D, E, Q\}$ and $AA_{-1} \notin \{D, E, Q\}$ and $AA \in \{A, C, F, I, L, M, V, W\}$ and $AA_1 \notin \{D, E, P\}$ and $AA_2 \notin \{X\}$ and $AA_3 \notin \{D, E, K, P, X\}$ and $AA_4 \notin \{E, K, P, Q, R, W, X\}$ then class is 1

2. Default class is 0

- All AA types associated to the central residue are hydrophobic (core of a protein)
- D, E consistently do not appear in the predicates. They are negatively charges residues (surface of a protein)

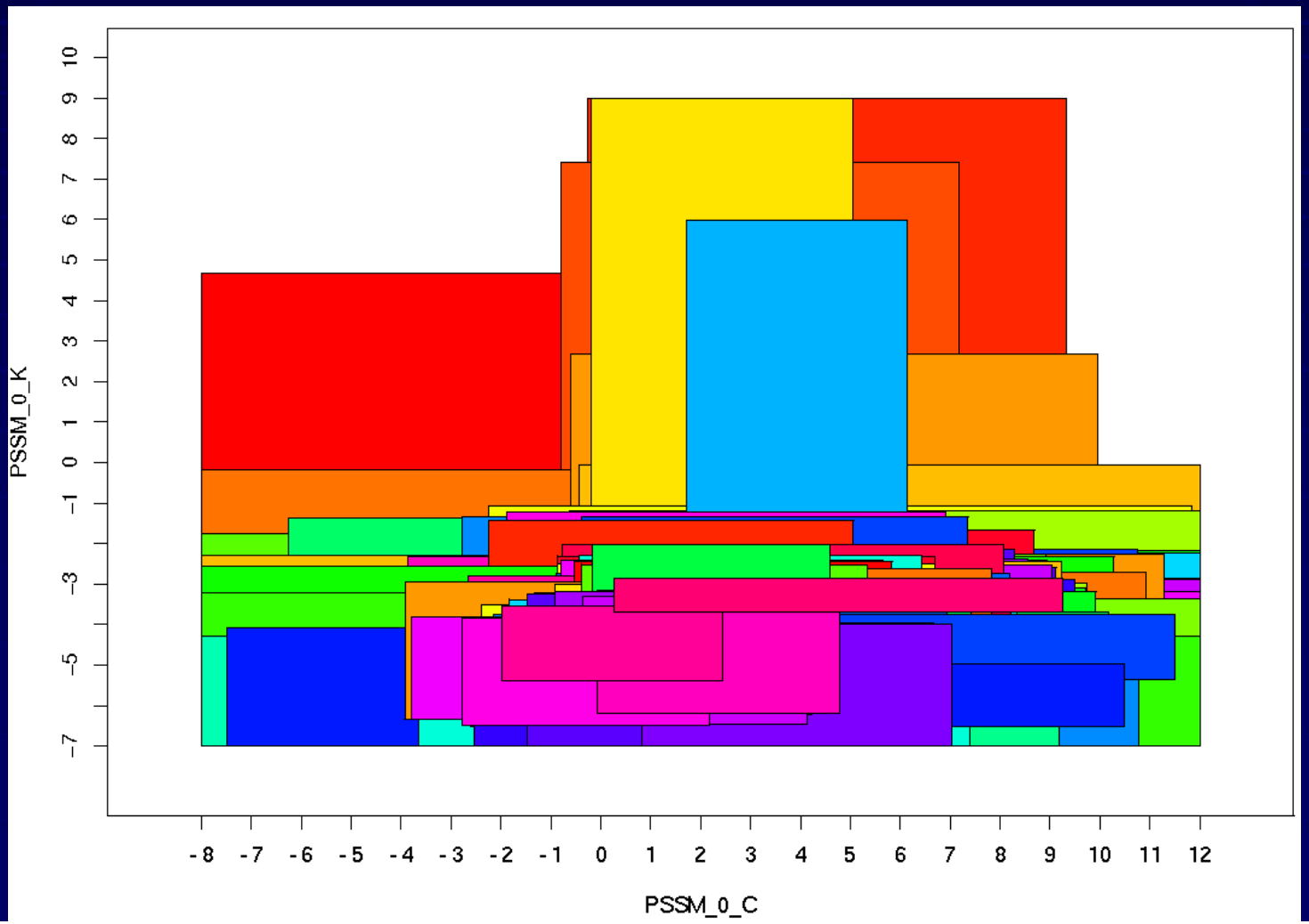
Interactions between attributes

- Interactions can be visualized by ploting rectangles from the rule intervals. SA prediction, interaction between polar AA types



Interactions between attributes

- Interaction between hydrophobic and polar attributes has a different shape for RCH



What lies in the future?

- Improvements in many dimensions of the learning process
  - Representations
  - Learning paradigms
  - Inference mechanisms (ensembles)
- However, all these components cannot be used blindly, they have to be adjusted accordingly to the characteristics/dimensions of the problem

What lies in the future?

- Theoretical analysis of the different facets of a GBML system can help
  1. Understand better why/when can the components perform well
  2. Design robust policies that can take the best of the techniques at hand
- GBML systems are highly flexible, with good explanatory power, and can have good scalability. If we know how to use them properly, we can take over the world !!! ☺