

Application of Machine Learning Algorithms to Constraint Solving and Learning Logical Implications in Spatial Domains

Carsten Gips

Sylvia Wiebrock

Fritz Wysotzki

Methods of Artificial Intelligence
Department of Computer Science, Technical University Berlin
FR 5-8, Franklinstraße 28/29, D 10587 Berlin
email: sppraum@cs.tu-berlin.de

1 Introduction

Methods of inductive generalization on examples developed in classification (concept-) learning can be applied to “prove” implications in predicate logic semantically, especially in the spatial domain, which may — in the general case — consist of an n -dimensional feature space. The learned rules provide knowledge of the type “region A is contained in region B ” (or “every $x \in$ region A is \in region B , too”) holding in the feature space defined by the parameters (variables) of the problem at hand. A special case is the learning of a decision function $A(x_1, \dots, x_n)$, which decides whether a vector $x = (x_1, \dots, x_n)$ belongs to the region where the predicate A is true. Seen from the logical point of view, this procedure provides knowledge (rules and facts) which may establish the knowledge base of an inference machine, which has usually to be given by the human user a priori.

On the other hand, one gets a tool to solve (in general only approximately) constraint satisfaction problems in the spatial domain, for example if a set of numerical constraints defines a complicated region in the feature space for which it is not easy or even not possible to find analytical boundary descriptions by some kind of formula manipulation system. The constraints in the example given below contain inequations with trigonometric functions leading to computational difficulties well known in robotics. Seen from the cognitive point of view, we claim that at least parts of human knowledge (facts and general rules) are learned by experience using the type of inductive inference introduced in this paper.

Formally, it is to be proved whether the implication (hypothesis) $\forall x(A(x) \rightarrow B(x))$ is true (as already mentioned above, proving a single constraint is a special case). A and B are expressions in predicate logic consisting of predicates describing numerical constraints. x is an n -dimensional vector of variables. I.e. the functions (truth-)value(A) and value(B) $\in \{0, 1\}$ describe (in the general case disconnected) class regions in \mathcal{R}^n , in which the constraints A and B are satisfied or not satisfied, respectively. (I.e. these functions are the membership functions of the A - and B -regions, respectively.)

If we wanted to prove the implication “semantically”, every valuation of x (by a real valued n -dimensional vector) satisfying A has to be shown to satisfy B , too. Since such testing is not possible in the complete domain of real numbers, we use inductive generalization, i.e. we generate a *finite* “training set” of points (vectors) equally distributed over a region in \mathcal{R}^n covering the A - and B -regions, respectively, and compute the truth values of $A(x)$ and $B(x)$ for all x in the training set using the given numerical constraints and the truth tables of logical functors occurring in A and B , respectively. I.e. a finite “set-of-support” for the functions value(A) and value(B) is generated. Now algorithms of classification learning (e.g., decision tree learning or Perceptron) construct classifiers which — by inductive generalization — decide the class membership of an arbitrarily chosen x (not necessarily contained in the training set) by some kind of “interpolation”. This is done in general with some generalization error arising in our case due to the unavoidable approximation of the boundaries between the A -region and the not- A -region (first classification problem to be solved) and between the B -region and the not- B -region (second classification problem), respectively. The generalization error can be measured using a test set of classified example vectors different from the training set. Decision tree learners approximate the class boundaries piecewise

and linearly by axis-parallel hyperplanes, generalized Perceptrons by hyperplanes in general position. Now the implication $\forall x(A(x) \rightarrow B(x))$ can be “decided” within the limits of approximation error by testing whether every x taken from the test set and classified as belonging to the A -region (i.e. $\text{value}(A)=1$) also belongs to the B -region ($\text{value}(B)=1$, too). This procedure has been successfully demonstrated by means of an example problem taken from the area of spatial inference using the decision tree learner CAL5 (Müller and Wysotzki 1994; Müller and Wysotzki 1997), see below. It can be shown that by gradually increasing the size of the training set together with increasing a certain parameter of CAL5, the accuracy of the class boundary approximation can (theoretically) be made arbitrarily high. This is analogous to the definition of real numbers in analysis by sequences of intervals of decreasing length. Of course, one could try to apply simple statistical tests to “prove” the implication statistically, but we are interested in explicit descriptions of class boundaries for reasons mentioned below. In Sect. 2 we demonstrate the method by means of two examples for which we performed first experiments. Some aspects of cognitive modelling are discussed in Sect. 3.

2 Examples

The following example is taken from constraint satisfaction problems arising in a new algorithm for spatial inference (Wysotzki et al. 1997; Claus et al. 1998) developed within the framework of the DFG Priority Program on “Spatial Cognition”. We explored the applicability of CAL5 for these problems by some simple examples. The scenario was always the same: it has to be determined whether an object is *right* of a person (indicated by a point S) or not. An object is *right_of* S iff every point of it is in the area bounded by the first and second diagonal (see Fig. 1). For our experiments we define that an object is in an “ A -region” in the feature space defined by the parameters of the problem (see below), iff its corners are *right_of* S . The object is in the “ B -region” iff any point of it is *right_of* S . Without loss of generality we used for the “ B -region” only one fixed point of the object, the point in the middle of the object.

2.1 Experiments with a bar

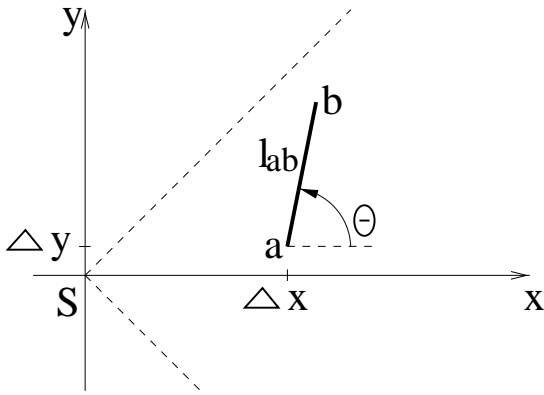


Figure 1: Bar *right* of S

Let Δ_x and Δ_y be the coordinates of end point a in the S -coordinate-system, respectively, and $\Delta_l = (\Delta_x - \Delta_y)/l_{ab}$, l_{ab} being the length of the bar. Then the A -region is determined by

$$\begin{aligned} \Delta_l &> 0 && \text{and} \\ \Delta_l &> \sin \theta - \cos \theta && \\ &&& \text{and the } B\text{-region by} \\ \Delta_l &> (\sin \theta - \cos \theta)/2 \end{aligned}$$

(In order to get a mapping into a 2-dimensional feature space defined by Δ_l and θ so that we can depict the class boundaries, we simplified the problem by considering the case $\Delta_y > 0$ only. With this restriction, in fact the relation “right_or_behind” bounded by the first diagonal is considered.)

With the abbreviation $x = \sin(\theta)$, the explicit boundary between the A -region and not- A -region can be computed in this simple case and is given by $x_A = f_A(\Delta_l) = \Delta_l/2 \pm \sqrt{1/2 - \Delta_l^2/4}$, analogous for the B -region $x_B = f_B(\Delta_l) = \Delta_l \pm \sqrt{1/2 - \Delta_l^2}$. We use it for comparison with the learned boundaries.

The two parameters θ and Δ_l span a region of the \mathcal{R}^2 . We generated a finite “training set” of points equally distributed over a subset $\theta \in [0, \pi]$ and $\Delta_l \in [-0.5, 2.0]$ of \mathcal{R}^2 and computed the truth values of A and B for each vector (θ, Δ_l) of the training set, respectively. (For these ranges, approximately 50% of the data were in class A , and approximately 65 % of the data were in class B .) Then the training set was split into a learning set and a test set for A and B , respectively. Now the decision tree learning algorithm CAL5 constructed classifiers for the A -region and B -region with the parameters `alfa = 0.25` and `threshold =`

0.85. The generalization error of both classifiers was measured. Finally we used the B -region classifier to classify the A -region test set in order to “prove” the implication $\forall \theta, \Delta_l(A(\theta, \Delta_l) \rightarrow B(\theta, \Delta_l))$.

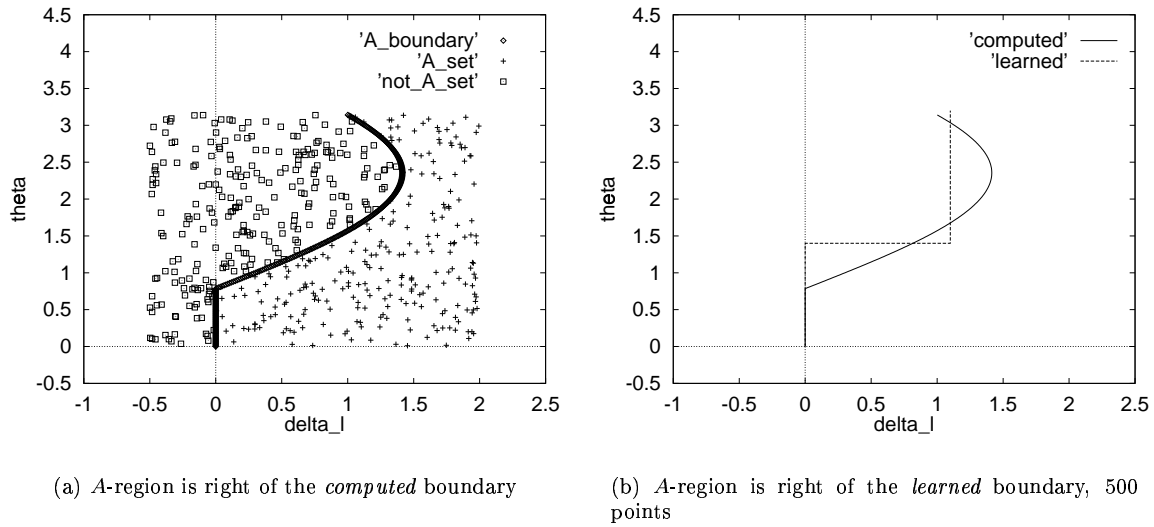


Figure 2: Computed boundary of the A -region and the learned boundary for 500 points

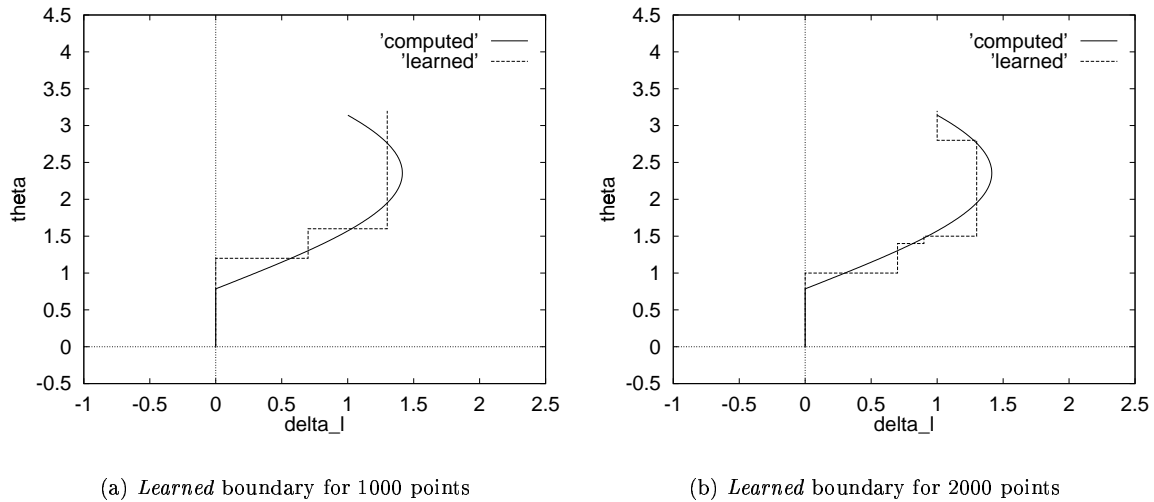


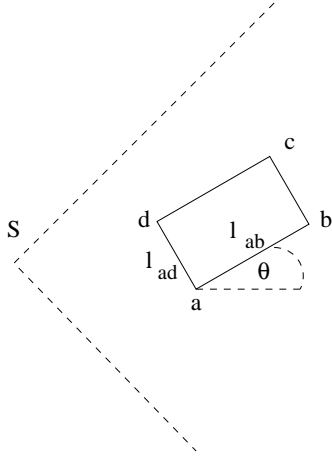
Figure 3: *Learned* boundaries compared with computed boundary

Fig. 2(a) shows the A -region with computed boundary vs. not- A -region. Fig.2(b) shows a comparison of the real (computed) boundary of the A -region with the learned one for a learning set of 500 points. There is a relatively large classifying error of 6 %.

The error shrinks with an increasing number of points for learning. This effect is visible in Fig. 3. The same holds for the B -region covering the A region (not shown in the figures). Furthermore the error for classifying the A -region test set by the classifier of the B -region is near 0 %. This means we “proved” the implication above inductively.

2.2 Experiments with a rectangle

In this section we consider the more complex problem of learning a function for deciding whether a rectangle is located completely in the *right* of region of an egocentric reference system with origin S (Fig. 4). The *right* region is bounded by the diagonals $y = x$ and $y = -x$.



The corners of the rectangle are a , b , c , and d , respectively, and the lengths of the sides are l_{ab} and l_{ad} , respectively. If we regard $a-b$ as the base side of the rectangle, θ is the angle by which the rectangle is rotated. $\Delta_x^{(S,p)}$ and $\Delta_y^{(S,p)}$, the offsets in x - and y -direction from the origin S to a point p , respectively, are the coordinates of p .

To check whether the rectangle $a - b - c - d$ is in the *right* - region of S , we have to check that all corners are within the region. This means the points have to be below the line $y = x$ and above the line $y = -x$. The inequations for the different corners are given below.

Figure 4: Rectangle $a - b - c - d$ in the *right* - area of S

	Upper bounds		Lower bounds
(a)	$\Delta_a := \Delta_x^{(S,a)} - \Delta_y^{(S,a)}$ (1)		$\Delta'_a := \Delta_x^{(S,a)} + \Delta_y^{(S,a)}$ (5)
	$\Delta := \Delta_a / l_{ad}$ (2)		$\Delta' := \Delta'_a / l_{ad}$ (6)
	$\Delta_x^{(S,a)} \geq 0$ (3)		$\ell := l_{ab} / l_{ad}$ (7)
	$\Delta_x^{(S,a)} \geq \Delta_y^{(S,a)}$		$\Delta_y^{(S,a)} \geq -\Delta_x^{(S,a)}$
	$\Delta_a \geq 0$		$\Delta'_a \geq 0$
	$\Delta \geq 0$ (4)		$\Delta' \geq 0$ (8)
(b)	$\Delta_x^{(S,b)} := \Delta_x^{(S,a)} + l_{ab} \cos \theta$		$\Delta_y^{(S,b)} := \Delta_y^{(S,a)} + l_{ab} \sin \theta$
	$\Delta_x^{(S,b)} \geq 0$		$\Delta_x^{(S,a)} \geq -l_{ab} \cos \theta$ (10)
	$\Delta_x^{(S,a)} + l_{ab} \cos \theta \geq \Delta_y^{(S,a)} + l_{ab} \sin \theta$		$\Delta_y^{(S,a)} + l_{ab} \sin \theta \geq -(\Delta_x^{(S,a)} + l_{ab} \cos \theta)$
	$\Delta_a \geq l_{ab}(\sin \theta - \cos \theta)$		$\Delta'_a \geq -l_{ab}(\sin \theta + \cos \theta)$
	$\Delta \geq \ell(\sin \theta - \cos \theta)$ (9)		$\Delta' \geq -\ell(\sin \theta + \cos \theta)$ (11)
(c)	$\Delta_x^{(S,c)} := \Delta_x^{(S,a)} + l_{ab} \cos \theta$		$\Delta_y^{(S,c)} := \Delta_y^{(S,a)} + l_{ab} \sin \theta$
	$\quad + l_{ad} \cos(\theta + 90)$		$\quad + l_{ad} \sin(\theta + 90)$
	$= \Delta_x^{(S,a)} + l_{ab} \cos \theta - l_{ad} \sin \theta$		$= \Delta_y^{(S,a)} + l_{ab} \sin \theta + l_{ad} \cos \theta$
	$\Delta_x^{(S,c)} \geq 0$		$\Delta_x^{(S,a)} \geq l_{ad} \sin \theta - l_{ab} \cos \theta$ (13)
	$\Delta_x^{(S,a)} + l_{ab} \cos \theta - l_{ad} \sin \theta$		$\Delta_y^{(S,a)} + l_{ab} \sin \theta + l_{ad} \cos \theta$
	$\geq \Delta_y^{(S,a)} + l_{ab} \sin \theta + l_{ad} \cos \theta$		$\geq -(\Delta_x^{(S,a)} + l_{ab} \cos \theta - l_{ad} \sin \theta)$
	$\Delta_a \geq l_{ab}(\sin \theta - \cos \theta) +$		$\Delta'_a \geq -l_{ab}(\sin \theta + \cos \theta)$
	$\quad l_{ad}(\sin \theta + \cos \theta)$		$\quad + l_{ad}(\sin \theta - \cos \theta)$
	$\Delta \geq \ell(\sin \theta - \cos \theta)$		$\Delta' \geq -\ell(\sin \theta + \cos \theta)$
	$\quad + (\sin \theta + \cos \theta)$ (12)		$\quad + (\sin \theta - \cos \theta)$ (14)

$$\begin{aligned}
(d) \quad \Delta_x^{(S,d)} &:= \Delta_x^{(S,a)} + l_{ad} \cos(\theta + 90) \\
&= \Delta_x^{(S,a)} - l_{ad} \sin \theta \\
\Delta_x^{(S,d)} &\geq 0 \\
\Delta_a &\geq -l_{ad}(\sin \theta + \cos \theta) \\
\Delta &\geq -(\sin \theta + \cos \theta) \quad (15)
\end{aligned}
\quad
\begin{aligned}
\Delta_y^{(S,d)} &:= \Delta_y^{(S,a)} + l_{ad} \sin(\theta + 90) \\
&= \Delta_y^{(S,a)} + l_{ad} \cos \theta \\
\Delta_x^{(S,a)} &\geq l_{ad} \sin \theta \\
\Delta'_a &\geq l_{ad}(\sin \theta - \cos \theta) \\
\Delta' &\geq (\sin \theta - \cos \theta) \quad (16)
\end{aligned}$$

Let m be the point in the middle of the rectangle. Then to prove that m is in the *right* - area of S , we have to prove the following inequalities:

$$\begin{aligned}
\Delta_x^{(S,m)} &:= \Delta_x^{(S,a)} + \frac{l_{ab}}{2} \cos \theta \\
&\quad + \frac{l_{ad}}{2} \cos(\theta + 90) \\
&= \Delta_x^{(S,a)} + \frac{l_{ab}}{2} \cos \theta - \frac{l_{ad}}{2} \sin \theta \\
\Delta_x^{(S,m)} &\geq 0 \\
\Delta_x^{(S,m)} &\geq \Delta_y^{(S,m)} \\
\Delta_x^{(S,a)} + \frac{l_{ab}}{2} \cos \theta - \frac{l_{ad}}{2} \sin \theta \\
&\geq \Delta_y^{(S,a)} + \frac{l_{ab}}{2} \sin \theta + \frac{l_{ad}}{2} \cos \theta \\
\Delta_a &\geq \frac{1}{2}(l_{ab}(\sin \theta - \cos \theta) \\
&\quad + l_{ad}(\sin \theta + \cos \theta)) \\
\Delta &\geq \frac{\ell}{2}(\sin \theta - \cos \theta) + \\
&\quad \frac{1}{2}(\sin \theta + \cos \theta) \quad (18)
\end{aligned}
\quad
\begin{aligned}
\Delta_y^{(S,m)} &:= \Delta_y^{(S,a)} + \frac{l_{ab}}{2} \sin \theta \\
&\quad + \frac{l_{ad}}{2} \sin(\theta + 90) \\
&= \Delta_y^{(S,a)} + \frac{l_{ab}}{2} \sin \theta + \frac{l_{ad}}{2} \cos \theta \\
\Delta_x^{(S,a)} &\geq \frac{1}{2}(l_{ad} \sin \theta - l_{ab} \cos \theta) \\
\Delta_y^{(S,m)} &\geq -\Delta_x^{(S,m)} \\
\Delta_y^{(S,a)} + \frac{l_{ab}}{2} \sin \theta + \frac{l_{ad}}{2} \cos \theta \\
&\geq -(\Delta_x^{(S,a)} + \frac{l_{ab}}{2} \cos \theta - \frac{l_{ad}}{2} \sin \theta) \\
\Delta'_a &\geq \frac{1}{2}(-l_{ab}(\sin \theta + \cos \theta) \\
&\quad + l_{ad}(\sin \theta - \cos \theta)) \\
\Delta' &\geq -\frac{\ell}{2}(\sin \theta + \cos \theta) \\
&\quad + \frac{1}{2}(\sin \theta - \cos \theta) \quad (20)
\end{aligned} \quad (19)$$

We ran some experiments with 100002, 200001, and 400002 points, which were randomly generated from a uniform distribution in the feature space defined by the parameters θ , $\Delta_x^{(S,a)}$, $\Delta_y^{(S,a)}$, l_{ab} , and l_{ad} . The ranges for the parameters were

$$-0.5 \leq \Delta_x^{(S,a)} \leq 20.0 \quad -20.0 \leq \Delta_y^{(S,a)} \leq 20.0 \quad (21)$$

$$0.0 \leq l_{ab} \leq 10.0 \quad \theta \in [0, 2\pi] \quad 0.0 \leq l_{ad} \leq 10.0 \quad (22)$$

For these ranges, between 30 and 35% of the data were in class A , and between 44 and 51% were in class B . Two thirds of the points were used for training, the last third was used for testing. We did not use crossvalidation because of the large running times¹. All experiments were run with the *CAL5* parameters `alfa` = 0.25 and `threshold` = 0.85. The results are shown in Tab. 1.

A is the region in the parameter (feature) space (defined by (in)equations (1) – (17)), in which the relation *right_of* holds for a , b , c , and d simultaneously. B is the *right_of* S region for the middle point m of the rectangle, defined by (in)equations (18) – (20). The last row of the table gives the error for the learned implication “if the corners of a rectangle are *right_of* S , the the middle of the rectangle is *right_of* S , too”. Note that this implies the rule “if the corners of a rectangle are *right_of* S , then all its inner points are *right_of* S ”, which is inductively proved for all values of the parameters within the region of the feature space defined by (21) and (22). This holds because for the four smaller rectangles with one of a , b , c , d , and m as opposite corners, the rule is also proved according to the generation of the training

¹In another series of experiments, we used the same data as in experiments 1, 4, and 6, but varied the *CAL5* parameters. The running times for the creation of the decision tree varied between about 8 minutes and nearly 30 hours for the data set of experiment 6 with 400002 points.

Error/ success rates	100002 points			200001 points		400002 points	
	Exp. 1	Exp. 2	Exp. 3	Exp. 4	Exp. 5	Exp. 6	Exp. 7
Learning <i>A</i>							
Error for not <i>A</i>	0.112418	0.052850	0.0841435	0.0744714	0.0919416	0.0883434	0.052451
Error for <i>A</i>	0.305488	0.39324	0.27957	0.170837	0.163193	0.209531	0.226188
Total success rate	0.827953	0.842773	0.855808	0.895923	0.886211	0.874571	0.893887
Testing <i>A</i>							
Error for not <i>A</i>	0.111044	0.055987	0.0895859	0.0786397	0.0950092	0.0969621	0.054815
Error for <i>A</i>	0.320533	0.403439	0.302439	0.184405	0.181525	0.228476	0.236501
Total success rate	0.824804	0.839143	0.844963	0.888746	0.878411	0.862436	0.889136
Learning <i>B</i>							
Error for not <i>B</i>	0.128785	0.137451	0.218854	0.147031	0.100431	0.104512	0.120771
Error for <i>B</i>	0.14392	0.223368	0.105405	0.113777	0.155403	0.12288	0.118886
Total success rate	0.863863	0.820934	0.836053	0.869081	0.873026	0.886627	0.880143
Testing <i>B</i>							
Error for not <i>B</i>	0.137712	0.137042	0.22477	0.153294	0.108291	0.107885	0.124511
Error for <i>B</i>	0.153923	0.223654	0.114377	0.121935	0.165174	0.128441	0.125308
Total success rate	0.854503	0.821414	0.828853	0.861956	0.864176	0.882146	0.875103
Error for <i>A</i> \rightarrow <i>B</i>	0.0431034	0.123944	0.0270244	0.0291371	0.0511669	0.0330386	0.031969

Table 1: Error and success rates for a series of experiments with *CAL5*

set. The parameter `threshold` of *CAL5* defines the maximally admissible error, which has been set to $1 - \text{threshold} = 0.15$ (see above). Looking at the table, it can be seen that this error is achieved for the data sets with 200001 points and 400002 points, respectively. Moreover, the error rates for the learning and test sets are nearly equal for each data set indicating a very good generalization (i.e. no overfitting). There is a large difference in the *A* and not_*A* errors which is probably caused by the corresponding difference of the sizes of the *A* and not_*A* regions in the feature space, which leads to different a priori probabilities for classes *A* and not_*A*. This effect can be compensated for by using class dependent `threshold` parameters (Müller and Wysotzki 1994; Müller and Wysotzki 1997). The more general problem behind these observations is connected with the generation of training sets by *experimentation*, whereas most of the learning algorithms that construct classifiers have to use (although in most cases this is not explicitly stated) training samples generated by a “stationary and independent source”, i.e. important statistical presuppositions are not strictly true in the case of learning by experimentation. This problem will be a topic of future research.

3 Cognitive Aspects

In Sect. 2 we mentioned the ability of the concept learning algorithm *CAL5* to decide for a class *k* using a class dependent threshold `thresholdk`. I.e. a decision for class *k* is made in an interval *I* if the probability $p(k/I)$ exceeds `thresholdk`. (This probability is estimated on a user defined confidence level $1 - \alpha$ by means of the relative class frequency in *I*.) Low `thresholdk` for some class *k* means that the learning algorithm is “more sensitive” with respect to this class or “pays more attention” to it. Since it can be

shown (Müller and Wysotzki 1994; Unger and Wysotzki 1981) that

$$\text{threshold}_k \approx \text{const}/C_k$$

where C_k is the cost for not recognizing class k , the algorithm tries to detect the "important" classes (i.e. those with low thresholds) more carefully in order not to overlook one of them. In general this leads to a higher mean error for important classes but to lower costs of misclassification.

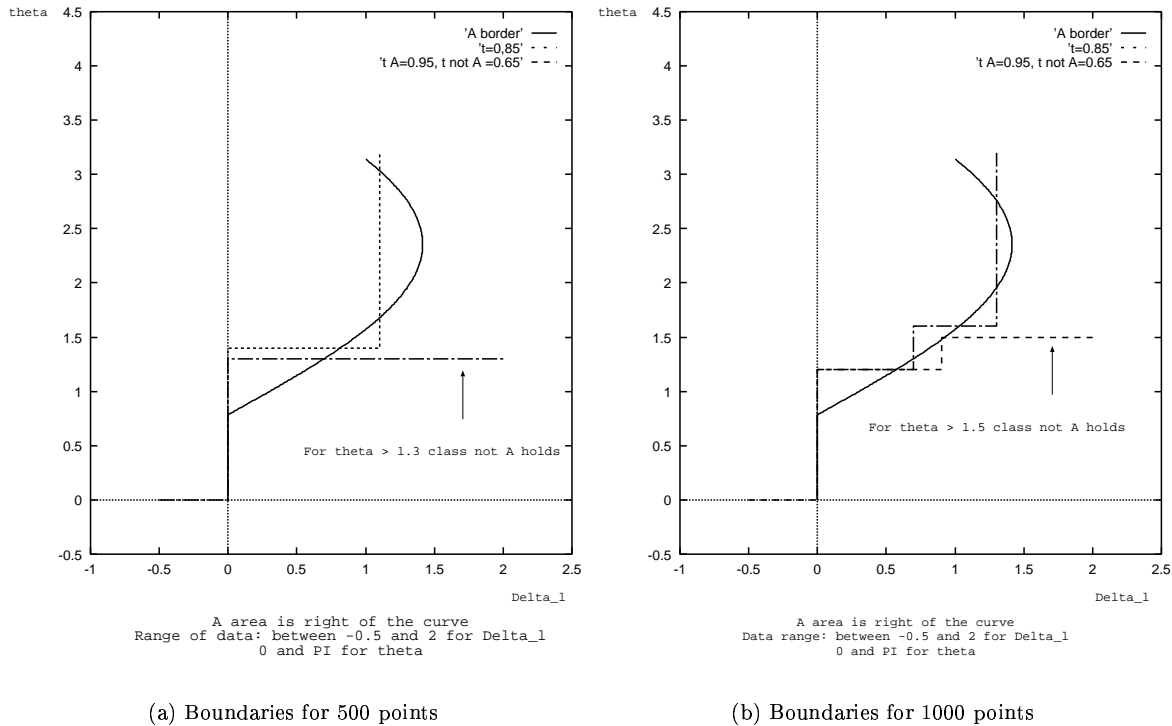
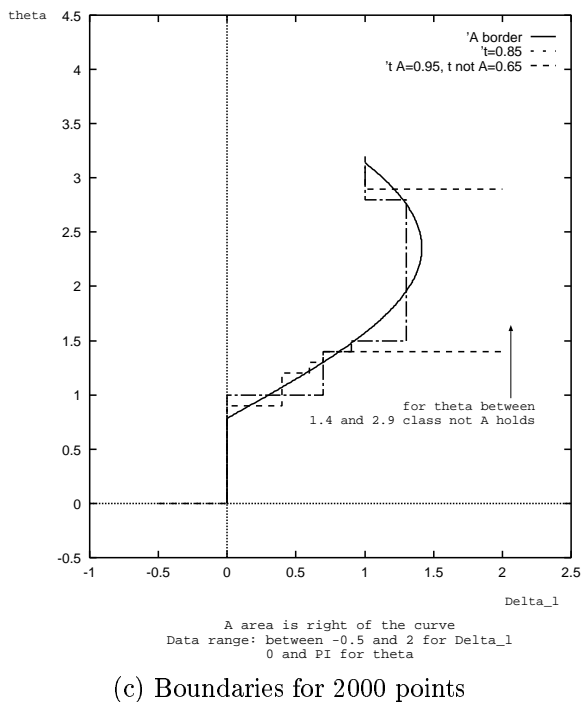


Figure 5: Comparison of boundaries for different thresholds



We performed experiments with the bar and considered not_A to be the important class which must not be overlooked. For demonstration purposes we set the threshold for not_A to the very low value of 0.65. Fig. 5 shows that the system tries to establish A-regions with small errors (i.e. if a point is classified as A, it is indeed in A) allowing higher errors (but lower costs of misclassifications) for the not_A regions. While the overall errors are large for this small number of objects, it is obvious that the computed boundaries move to the right, i.e. the probability of erroneously deciding for A when a point is in not_A is reduced.

Another cognitive aspect connected with our experiments is skill learning if we got the training samples for our learning problem from generating trajectories along which the objects (bar or rectangle in our case) move. Imagine for example a rat getting a negative reinforcement (e.g. a painful electric shock) if it runs into the not_A-region. That is it will learn by trial-error a

decision function for A/not_A -discrimination, i.e. implicitly the boundary between the classes – without constructing an analytical description in its head! The not_A -region will then be detected more carefully (for example by sensory features if there are any or by using a memory for orientations and positions of the rat) in order to avoid it. High cost means pain in this case. Another (more complicated) example is the learning of obstacle avoidance by a child or robot. Thus we can easily extend our examples discussed in Sect. 2 to prototypical cases of skill acquisition where learning takes place in the so-called configuration (motor-) space into which the original 2- or 3d-space is transformed. Finally let us note that in organismic information processing only approximations of class boundaries can be computed as in our case, i.e. some “fuzziness” is unavoidable.

4 Conclusion

In this paper we presented first experimental results on learning numerical constraints and logical implications in the spatial domain. One of the main reasons for conducting this research was the assumption that deciding whether a given object satisfies a set of complicated numerical constraints is computationally easier and faster with a learned decision function containing computationally simple operations (as compared with manipulating sets of parameterized numerical (in)equations). In our first experiments we used decision trees, i.e. test operations have to be performed only. The price one has to pay for this is a relatively rough approximation of class boundaries by axis-parallel hyperplanes. Therefore, in the case of high dimensional feature spaces (large number of parameters), large training sets are necessary to get a sufficient accuracy.

In forthcoming work other algorithms for learning classifications will be applied, too, and some theoretical investigations concerning the convergence to the real boundaries in dependence on the type of the learning algorithm and its parameters will be performed. For example piecewise linear classifiers constructed by DIPOL (Schulmeister and Wysotzki 1997) will achieve better approximations of class boundaries and therefore smaller error rates.

Finally, the tight relation to learning control and generation of “imagery” (depictions) from the learned conceptual structure will be considered. A more detailed description of the experiments and their results can be found in Geibel et al. (1998).

References

- Claus, B., K. Eyferth, C. Gips, R. Hörnig, U. Schmid, S. Wiebrock, and F. Wysotzki (1998). Reference Frames for Spatial Inferences in Text Comprehension. In C. Freksa, C. Habel, and K. F. Wender (Hrsg.), *Spatial Cognition - An Interdisciplinary Approach to Representing and Processing Spatial Knowledge*, LNAI 1404. Springer Verlag. <http://ki.cs.tu-berlin.de/~sppraum/Papers/Trier97/trier.ps>.
- Geibel, P., C. Gips, S. Wiebrock, and F. Wysotzki (1998). Learning Spatial Relations with CAL5 and TRITOP. Technical Report Fachberichte des Fachbereichs Informatik, Report No. 98-7, TU Berlin. In preparation.
- Müller, W. and F. Wysotzki (1994). Automatic Construction of Decision Trees for Classification. *Annals of Operation Research* 92, 231–247.
- Müller, W. and F. Wysotzki (1997). The Decision-Tree Algorithm CAL5 Based on a Statistical Approach to Its Splitting Algorithm. In G. Nakhaeizadeh and C. C. Taylor (Hrsg.), *Machine Learning and Statistics - The Interface*, S. 45–65. Wiley.
- Schulmeister, B. and F. Wysotzki (1997). DIPOL – A Hybrid Piecewise Linear Classifier. In G. Nakhaeizadeh and C. C. Taylor (Hrsg.), *Machine Learning and Statistics - The Interface*, S. 133–152. Wiley.
- Unger, S. and F. Wysotzki (1981). *Classification Systems Being Able to Learn*. Berlin: Akademie-Verlag.
- Wysotzki, F., U. Schmid, and E. Heymann (1997). Modellierung räumlicher Inferenzen durch Graphen mit symbolischen und numerischen Constraints. In C. Umbach, M. Grabski, and R. Hörnig (Hrsg.), *Perspektive in Sprache und Raum*. Deutscher Universitätsverlag, Wiesbaden.