Information-Based Selection of Abstraction Levels

Stephan Schulz
Institut für Informatik, Technische Universität München, Germany
schulz@informatik.tu-muenchen.de

Abstract

We model the learning of classifications as a combination of abstraction and class assignment. We discuss the problem of selecting the most suitable of multiple abstractions for this purpose. Weaker abstractions perform better on training sets, but typically do not generalize very well. Stronger abstractions often generalize better, but may fail to include important properties. We introduce the *relative information gain* as a criterion to determine an optimal balance between precision and generality of abstractions. Experimental results with abstractions used for the classification of terms indicate the success of this approach.

Introduction

Any real-world application of artificial intelligence starts with the selection of a certain abstraction for the objects it wants to reason or argue about. The choice of a suitable level of abstraction can very well be crucial for the success of an application. Too high levels of abstraction may ignore certain important properties, while an abstraction that is too weak will burden an implementation with many unnecessary details.

While basic modeling decisions have to be made by human beings, *machine learning* can be used to automatically find useful abstractions for many tasks, in particular for the classification of objects represented in some given data format. In fact, many symbolic machine learning algorithms create an abstraction function that partitions the space of all objects into a (usually finite) number of subsets and assign the most frequent class of objects in a *training* set to all new objects mapped into this class. Examples are the large family of decision tree learning algorithms from ID3 to C4.5(Quinlan 1996; 1992) or CN2-like algorithms (Clark & Niblett 1989) that construct hypotheses as conjunctions of elementary feature tests. The quality of the generated classifiers depends on the level of the selected abstraction. On the one hand, a very weak abstraction (as an extreme example, consider the case of using no abstraction, i.e. learning by hard) will perform very good on a given training set. However, it is unlikely to generalize well to new examples. On the other hand, an abstraction that is to strong may ignore important information and result in a suboptimal classification.

Typically, the abstractions used in standard symbolic machine learning algorithms are generated by successively examining individual and at least conceptually independent features of the objects to be classified. Most induction tree algorithms use a greedy algorithm, repeatedly partitioning the set of training examples while optimizing the *information gain* of each split. CN2, on the other hand, performs a heuristic greedy search on the space of all possible hypotheses. For both kinds of algorithms, there is no satisfiable criterion about when to stop refining the abstraction. For decision trees, the tree is typically grown until the subsets of the training sets generated by the tree contain only examples from a single class, and is then pruned using heuristic criteria. For CN2, an arbitrary significance level is used to stop refining hypotheses if no sufficiently significant new features can be added.

This weakness in determining the optimal level of abstraction also shows if some features of an object are not binary, but discrete or even continuous. In this case, there are multiple ways to select a feature test: Either using a single threshold value for a binary split or multiple thresholds for a more refined split. A refined split typically yields more information, but often leads to less accuracy on unknown examples. This is acknowledged e.g. in (Quinlan 1992), and a proposed solution is to always replace a possible multi-way split with a number of binary splits. This may, however, result in unnecessary large trees and in suboptimal feature selection.

For our application of machine learning, classification of search decisions for the automated theorem prover E (Schulz 1999; 2000), we needed to find and compare different abstractions for first-order *terms*, recursive structures built over a finite set of symbols. Natural abstractions are initial parts of the terms (see the *experiments* section below), however, these different abstractions are in no way independent from each other, and a weaker abstraction always yields more information on a test set. We therefore had to find an objective criterion to prefer one abstraction to the other.

If we consider a pre-classified set of training examples and a given partition of this set, the partition always contains two different kinds of information: Information that helps us to determine the class of an object, and superfluous information only introduced by the partition. The core idea of our approach is to not greedily optimize for information gain (towards the desired classification), but to compare the useful information gained from an abstraction to the additional amount of information necessary to compute the abstraction.

We will formalize this notion in the following sections and present experimental results that demonstrate the success of this approach.

Abstractions

Let D (the *domain*) be a set of objects. An *abstraction* of D is a tuple (A, d), were d is an (usually finite) set (of abstraction values) and A is a surjective mapping $A : D \to d$. An abstraction induces a partitioning of the domain, $D = \uplus_{i \in d} A_{\bar{i}}$, where $A_{\bar{i}} = \{x \in D | A(x) = i\}$. We also say $A(x)$ is the abstraction of x (under A).

Now consider two abstractions (A, d) and (A', d') for the same domain D. We say (A, d) is *weaker* (than (A', D')), if $|d| > |d'|$. We say (A, d) is *more precise*, if there exist a function $f : d \to d'$ so that $f \circ A = A'$, and that it is strictly more precise, if f is not injective. Note that an abstraction that is strictly more precise than another also is weaker, but not necessarily vice versa.

We will usually consider d to be implicitly given. Examples of abstractions are e.g. the test of a multi-valued feature or the mapping of trees to fixed size initial parts of the trees.

Note that from an information-theoretical point of view, the application of a finite abstraction corresponds to a *probabilistic experiment* with the possible results of the abstraction values. Given a representative sample of objects (a *training set*[1]) T, we can estimate the probability distribution of the experiment using the relative frequency, i.e. $P(i) = (|T \cap A_{\bar{i}}|)/|T|$ for all $i \in d$. The relative frequency is not always the best estimator for probabilities, however, it is widely used and in many cases adequate. If we know more about the probability distribution of the domain, better estimators may be available.

Entropy and Information Gain

We give a short introduction to information theory as originally introduced by Shannon (Shannon & Weaver 1949) and introduced to machine learning by Quinlan. Let $A = \{a_1, \ldots, a_n\}$ be an experiment with the possible results or observations $a_1, \ldots, a_n$ and let $P : A \to \mathbf{R}$ be a probability distribution on the results of A (i.e. $P(a_i)$ is the probability of observing a_i as the result of A). The amount of *information* gained from the observation of an event (or the *information content* of this event) is $I(a_i) = -log_2(P(a_i))$. The *entropy* of A is the expected information gain of performing A, $H(A) = \sum_{i=1}^{n} P(a_i) I(a_i)$.

If we consider more than one experiment, they may not be completely independend. The outcome of one experiment may already tell us something about the expected outcome of the other. Formally, let $A = \{a_1, \ldots, a_n\}$ and $B = \{b_1, \ldots, b_m\}$ be two experiments with probability distributions P_a and P_b. $P(a|b)$ is the *conditional probability* of a under the condition b. The *conditional information content* of a result a_i under the condition b_j is $I(a_i|b_j) = -log_2(P(a_i|b_j))$. The *conditional entropy* of A under the condition b_j is $H(A|b_j) = \sum_{i=1}^{n} P(a_i|b_j) I(a_i|bj)$.

If we want to determine the value of an experiment B (e.g. a feature test or, more generally, the application of an abstraction function) for getting information about the outcome of A (the class of an object), we again need to average over the possible outcomes of B to determine the *remaining entropy* of A. Our expected *information gain* from the experiment B then is exactly the difference in the entropy of A and the remaining entropy of A after performing B. Formally, the *remainder entropy* of A (after performing B) is $H(A|B) = \sum_{j=1}^{m} P_b(b_j) H(A|b_j)$, and the (expected) *information gain* of performing B (to determine the result to A) is $G(A|B) = H(A) - H(A|B)$.

Relative Information Gain

The expected information gain is e.g. used by Quinlan in C4.5 to select the feature test that yields the most useful information at each stage of the decision tree. In many cases this works quite well, because the features are conceptually independent from each other and often only weakly correlated in a statistical sense. However, if we consider the more general case of different abstractions, it becomes clear that maximal information gain is not necessarily optimal. Consider a simple case of classifying integer numbers from the set $\{1, \ldots, 10\}$ into even and odd numbers. Both the optimal abstraction (corresponding to the partitioning into the desired classes) and the identity function result in optimal classification on this set, and hence yield the same amount of information with respect to the task. However, the first one is clearly superior for classifying unknown numbers – a fact that is also intuitively clear. The reason for this is that in order to determine the abstraction of a number under the identity abstraction needs a lot more information - in our example, the correct classification needs one bit, while the determination of the exact number needs about 3.33 bits. The 2.33 bit not contributing to the correct classification do make th abstraction more precise, and hence do lead to a los of generality. In the general case, weaker abstractior are likely to yield less information but to generalize be ter. A more precise abstraction always yields at lea

[1]More exactly, we have to deal with *multi-sets* here. However, given standard terminology (Dershowitz & Manna 1979), the notation is identical.

the same amount of information than a less precise one, and is likely to yield more.

To find an optimal balance between useful information gain and generality of an abstraction, we do not need to optimize the useful information gain, but the ratio of useful and useless information. Formally, we define the *relative information gain* of an experiment B to determine the outcome of another experiment A as $R(A|B) = G(A|B)/(H(B) - G(A|B))$. For the borderline case, where B exactly determines A and this expression is undefined, we assume a value of $+\infty$, which is held to be bigger than any real number.

Now consider a training set $B = B_1 \uplus \ldots \uplus B_n$ of pre-classified examples from a domain D, and a set of abstractions $\mathcal{A} = \{A_1, \ldots, A_n\}$ for D. Both B and the A_i define probabilistic experiments (via the relative frequencies) as described above. We say A_i is an *information-optimal* abstraction from $\mathcal{A}$ (with respect to B), if $R(A_i|B) = max_{A_j \in \mathcal{A}} R(A_j|B)$.

We believe that the experimental results in the following sections justify our belief that an information-optimal abstraction indeed strikes an excellent balance between precision and generality.

Simple Term Classification

In this section, we will introduce *flat term space mapping* for the classification of terms. For information about general term space mapping, a class of learning algorithms for the classification and evaluation of terms, see (Schulz 2000). A more detailed discussion of first order terms can also be found there or in any introductory text on logic, e.g. (Chang & Lee 1973).

Terms and Term Abstractions

Let F be a (finite) set of function symbols with associated arities (written as f/n if f is a symbol of arity n), and let V be an enumerable set of variable symbols, $F \cap V = \emptyset$. The set $Term(F, V)$ of *terms* over F and V is defined recursively: Each $x \in V$ is term. If $f/n \in F$ and $t_1, \ldots, t_n \in Term(F, V)$, then $f(t_1, \ldots, t_n) \in Term(F, V)$. In this context, we can usually just treat variables as additional function symbols of arity 0. A term without any variables is called *ground*.

It is easy to see that terms can be represented as ordered, labeled trees. More generally, we can represent a term as a labeled, ordered, directed acyclic graph. This gives us the ability to share some identical subterms in a term. If *no* subterm is replicated in a graph representation of a term t, we call this graph the *maximally shared graph representation* of t.

We now define a variety of abstractions on terms. First, the *arity* of the top function symbol of a term is such an abstraction. The *top term* of t at level i, $top(t, i)$, is the term resulting if we replace every node reachable from the root node by a path of length i in the tree representation of t with a fresh variable. The *alternate top term* of t at level i, $top'(t, i)$, is the term resulting if we replace every *distinct subterm* at a node reachable from the root node by a path of length i in the tree representation of t with a fresh variable. The *compact shared top term* of t at level i, $cstop(t, i)$, is the term resulting if we replace every node reachable from the root node by a path of length i in the maximally shared graph representation of t with a fresh variable. The *extended shared top term* of t at level i, $estop(t, i)$, is the term resulting if we replace every node reachable from the root node by a path of length i, but not by any shorter path, in the maximally shared graph representation of t with a fresh variable. Finally, the *identity function* also is a (maximally weak) abstraction on terms.

Flat Term Space Mapping

Now consider a finite training set of pre-classified terms, $B = B_1 \uplus \ldots \uplus B_n$ and a term abstraction (A, d). We construct a new, finite abstraction (A', d') for terms as follows: $d' = A(B) \cup \{\perp\}$ for a new element $\perp \notin d$. $A'(t) = A(t)$ if $A(t) \in A(B)$, $A'(t) = \perp$ otherwise. The new element $\perp$ is called *the representation of unmapped terms.* (A', d', B) represent a *flat term space map* $tsm_{A,d,B} : Term(F, V) \to \{B_1, \ldots, B_n\}$ that defines a classification on terms: If $A'(t) = x \in d$, then $tsm_{A,d,B}(t) = B_i$, where B_i is the most frequent class in $A_{\overline{x}} \cap B$. Otherwise, $tsm_{A,d,B}(t) = B_j$, where B_j is the most frequent class in B.

Again, if we have a set of term abstractions and a training set, we call a flat term space map *information-optimal* (with respect to the training set and the abstractions), if the corresponding abstraction is information-optimal.

Experimental Results

We have developed term space mapping and the relative information gain criterion in the context of our work on learning search control knowledge for a saturating theorem prover. However, the performance of a theorem prover is influenced by a lot of different variables, and moreover, testing theorem provers is very expensive in terms of CPU time. We have therefore performed some artificial classification experiment to test the relative information gain criterion independently.

Setup

We have generated two sets of 20000 pseudo-random ground terms over the set of function symbols $\{f_0/0, \ldots, f_3/0, f_4/1, f_5/1, f_6/1, f_7/2, f_8/2, f_9/3\}$ using a recursive constructive algorithm[2], one containing only unique terms, the other one also allowing repeated terms.

[2]Note that as the set of terms is infinite for all non-trivial signatures, it is not obvious what a natural probability distribution on terms is. Our algorithm (Schulz 2000) strongly prefers smaller terms and thus stochatically terminates relatively fast.

We have designed three experiments, each with a positive and a negative class: *Topstart*, *Symmetry* and *Memorization*. For the *Topstart* experiment, positive terms start with a non-constant function symbol and carry a constant or unary symbol at the top position of the first argument term. We use the two term sets as they were generated. The second experiment, *Symmetry*, tries to recognize terms which start with a binary function symbol and where both argument terms are identical. As very few such terms are generated by our algorithm, we randomly selected 10000 of the terms and transformed them into symmetric terms by adding a binary function symbol and using the original term for both arguments. Finally, for *Memorization* we simply assign one of two classes at random, but use two copies of each term.

We have used all abstractions described above for the algorithm: Top symbol arity, identity, and the 4 different term top abstractions for depths between 1 and 5. We only report data for the most interesting subset of abstractions, always including the best and the information-optimal abstraction. To get statistically significant results, we have performed 10-fold cross-validation for all experiments. We also contrast the results with those of a random guesser (guessing according to the class distribution in the training set) and a naive learner (always guessing the most frequent class).

Results

Table 1 shows the results for the *Topstart* experiment. For the set of unique terms, the positive class contains 9299 terms, the negative class 10701 terms. Hence a random guesser would achieve 50.246%, a naive learner would achieve 53.505%. For the training set that allows multiple occurrences of the same term, the split is 10319/9681, hence a random guesser would achieve 50.051% and a naive guesser 51.595%. As we can see, all abstractions achieve significantly better results. However, the outstanding results are achieved by the term top abstraction for depth two, which is the weakest of the abstractions that contains enough information for correct classification. The information-optimality criterion correctly identifies this as the best abstraction.

Abstraction	Unique Terms	Repeated T.
Arity	58.4±1.12	56.6±1.07
Identity	53.5±1.08	60.3±1.01
$top(t, 1)$	58.4±1.12	56.6±1.07
$top'(t, 1)$	58.3±1.10	56.6±1.06
$top(t, 2)$	**99.2±0.16**	**99.7±0.15**
$top'(t, 2)$	98.0±0.40	97.7±0.43
$estop(t, 3)$	72.9±0.94	72.3±0.84
$top'(t, 4)$	60.3±0.96	61.1±0.99
I.-Optimal	**99.2±0.16**	**99.7±0.15**

Table 1: Correct classification for *Topstart* (Percent)

The results for the *Symmetry* experiment are summarized in Table 2. Due to the special construction of the term sets for this experiment, both sets are very well balanced, and random guesser and naive learner would both score between 50% and 51% for both experiments. For the case where the term set contains only unique terms, the optimal abstraction is $top'(t, 1)$, which is correctly identified by our criterion. It is interesting to note that the identity abstraction, corresponding to *learning by hard*, contains *no* information for this classification task.

The second case, including repeated terms, the information-optimality criterion fails to identify the optimal abstraction. The reason for this is the particularly skewed term distribution. Since repeated terms are allowed in term set B, and since most generated terms are small terms, nearly all terms with arity 2 are artificially generated symmetric terms. Just checking the arity of the top function symbol results in nearly 85% classification correctness. The relative information gain criterion prefers this 4-way split with high accuracy (relative information gain 0.568412) to the much larger split with perfect accuracy it gets for $top'(t, 2)$, although only barely. The relative information gain for this split is 0.52221. However, we believe that the underlying reason is that the frequency count in this case is a bad estimator for probabilities. Note that we had to artificially create the symmetric terms to get an acceptable ratio for the two classes, and that for a term generation scheme that is not biased against large terms (with high-arity top symbols), the relative information gain for the arity abstraction would drop.

Abstraction	Unique Terms	Repeated T.
Arity	77.5±0.95	82.7±0.80
Identity	49.3±0.81	62.0±0.70
$top(t, 1)$	77.5±0.95	82.7±0.80
$top'(t, 1)$	**100.0±0.02**	**100.0±0.00**
$top(t, 2)$	95.9±0.96	96.8±0.53
$top'(t, 2)$	97.6±1.66	97.8±1.19
$estop(t, 3)$	74.0±14.07	77.9±8.90
$top'(t, 4)$	55.9±5.79	66.0±3.44
I.-Optimal	**100.0±0.02**	82.7±0.80

Table 2: Correct classification for *Symmetry* (Percent)

For the memorization experiment, all terms are duplicated anyways. Hence we have restricted the test to the case where the original term set contained unique terms only. Table 3 contains the full data for all abstractions we have tried. As the classes have been assigned randomly, there is no particular feature that can be recognized. The success rate can only be increased by recognizing known terms. Please note that for 10-fold cross validation the chance for a term in the test set to also occur in the training set in this case is about 90% (90.0025 to be exact: there are two copies of each term, the remaining copy for an arbitrary term in the training set is either one of the 3999 terms in the test set or one of the 36000 terms in the training set). In other words, the best performance we can expect is about 95% (about 90% for the case of perfect memorization, and 5% by randomly guessing the class of the remaining 10% of terms). As the positive and the negative classe are of exactly the same size, both the random guesser and the naive learner would only achieve 50%.

The identity abstraction achieves the optimal performance (up to statistical significance), and is correctly identified as the best abstraction by the information-optimality criterion.

Abstr.	Successes	Abstr.	Successes
Arity	50.0±0.59	$top'(t,3)$	78.7±0.72
Identity	**94.8±0.57**	$cstop(t,3)$	79.4±0.66
$top(t,1)$	50.4±0.46	$estop(t,3)$	80.1±0.66
$top'(t,1)$	50.4±0.46	$top(t,4)$	91.2±0.66
$cstop(t,1)$	50.4±0.46	$top'(t,4)$	91.6±0.69
$estop(t,1)$	50.4±0.46	$cstop(t,4)$	91.9±0.71
$top(t,2)$	53.1±0.50	$estop(t,4)$	92.0±0.70
$top'(t,2)$	55.2±0.58	$top(t,5)$	94.5±0.59
$cstop(t,2)$	55.3±0.54	$top'(t,5)$	94.5±0.59
$estop(t,2)$	55.5±0.51	$cstop(t,5)$	94.6±0.59
$top(t,3)$	76.6±0.51	$estop(t,5)$	94.6±0.59
I.-Optimal	**94.8±0.57**		

Table 3: Results for *Memorization* (Percent)

All in all, we consider these results to be very encouraging.

Conclusion

We believe that the relative information gain is a very powerful method for the evaluation of abstractions. In our application area, automated theorem proving, we found that the best abstraction among those we tested usually is the identity, i.e. search decisions are best represented by learning complete facts (patterns of first-order clauses). This was indicated by the relative information gain and is reflected by the success rates of the theorem prover (Schulz 2000). This agreement of direct evaluation and success in the application area is additional evidence for the value of this criterion.

In the future, it would be very interesting to incorporate this criterion into more traditional machine learning algorithms. As an example, the selection of optimal multi-way splits in decision trees (as described in the introduction) seems to be possible field of application. Moreover, it can also be used as a pruning criterion for complete decision trees or other, similar machine learning algorithms.

As our main field of research is automated theorem proving, we would very much encourage other researchers to make use of these results.

References

Chang, C., and Lee, R. 1973. *Symbolic Logic and Mechanical Theorem Proving.* Computer Science and Applied Mathematics. Academic Press.

Clark, P., and Niblett, T. 1989. The CN2 Induction Algorithm. *Machine Learning* 3.

Dershowitz, N., and Manna, Z. 1979. Proving Termination with Multiset-Orderings. *Communications of the ACM* 22(8):465–476.

Quinlan, J. 1992. *C4.5: Programs for Machine Learning.* San Mateo, CA: Morgan Kaufmann.

Quinlan, J. 1996. Indiction of Decision Trees. *Machine Learning* 1:81–106.

Schulz, S. 1999. System Abstract: E 0.3. In Ganzinger, H., ed., *Proc. of the 16th CADE, Trento,* number 1632 in LNAI, 297–391. Springer.

Schulz, S. 2000. *Learning Search Control Knowledge for Equational Deduction.* Number 230 in DISKI. Akademische Verlagsgesellschaft Aka GmbH Berlin. Ph.D. Thesis, Fakultät für Informatik, Technische Universität München.

Shannon, C., and Weaver, W. 1949. *The Mathematical Theory of Communication.* University of Illinois Press.