Refinement of Rule Sets with JoJo

Dieter Fensel and Markus Wiese

Institut für Angewandte Informatik und Formale Beschreibungsverfahren,
University of Karlsruhe, P.O. Box 6980, 7500 Karlsruhe, Germany
e-mail: fensel@aifb.uni-karlsruhe.de

Abstract. In the paper we discuss a new approach for learning classification rules from examples. We sketch out the algorithm JoJo and its extension to a four step procedure which can be used to incrementally refine a set of classification rules. Incorrect rules are refined, the entire rule set is completed, redundant rules are deleted and the rule set can be minimized. The first two steps are done by applying JoJo which searches through the lattice of rules by generalization and specialization.

1 Introduction

Learning from example deals with the task of learning general descriptions (decision trees or rules) from examples. In the paper we discuss a four step procedure which can be used to refine a set of classification rules. First, the rules which become incorrect because of new negative examples are refined. Each incorrect rule is replaced by a set of rules which covers the positive examples but not the new negative ones. In a second step, the rule set is extended to cover the new positive examples. Third, redundancy of the rule set is corected by deleting rules. In a fourth step, a minimal subset can be computed which covers all positive examples. Steps one and two are carried out by applying the JoJo-algorithm. The main feature of JoJo is that it integrates generalization and specialization into one heuristic search procedure.

The *version space* algorithm is one of the earliest algorithms which obtained classification rules from a set of examples [13]. It applies a dual search strategy in the lattice of possible rules. If a negative example is given, all general rules which cover it are specialized. If a positive example is given, all special rules which do not cover it are generalized. The search procedure starts at the top and the bottom of the lattice and converges to the final rule which covers all positive and no negative examples. The version space algorithm does a complete search and can therefore only be applied to small data sets because it is impossible to find the minimal hypothesis (Occam´s Razor) consistent with the given examples in polynomial time [1].

Heuristic (i.e. incomplete) search procedures like *AQ* [14], *C4* [16], *CN2* [3], *CABRO* [11], *FOIL* [16], and *PRISM* [2] work by *specialization* only. They start with very general descriptions and specialize them until they are correct. This is done by adding additional premises to the rule or by restricting the range of an attribute which is used in a premise.

In [5] we discussed the heuristic search procedure *RELAX* which works by *generalization* only. It starts with very special descriptions and generalizes them as long as they are not incorrect. It regards every example as a very specific rule which is generalized. This is done by deleting premises from the rule. The motivation for this procedure are algorithms used for minimizing electronic circuits [12].

Algorithms which use specialization as search strategy generally have the problem of overspecialization. In an i-th specialization, a specialization can performed which could

possibly make earlier specialization unnecessary. If for example, three premises are added to a rule in three steps, it could be that the premises which were added in the second and third step make the premises of the first step unnecessary (cf. [6] for an example).[1] The rule could be made more general by deleting the first premise without decreasing its correctness. Therefore, when using specialization as a search direction it cannot be guaranteed that the result is really a maximal-general description. This problem does not exist when using generalization as search strategy. In every step one tries to delete an unnecessary premises. The procedure stops if no such premisses exist anymore. Therefore, it is guaranteed that the final rule is a most-general description, i.e. no other rule which is more general is correct. On the other hand, learning most-specific descriptions with generalization as a search direction would have the dual problem that one cannot be sure that the final result is really most-specific.[2]
The problem of rules which are constructed by specialization is discussed by [15] as necessity to *prune* rules. Rules which are constructed by specialization are pruned in a second step. It is determined whether they can be generalized without changing their correctness. Therefore, [15] proposes a specific combination of both search paradigms. First, rules are constructed by specialization and in a second generalization phase it is determined whether they can be generalized.

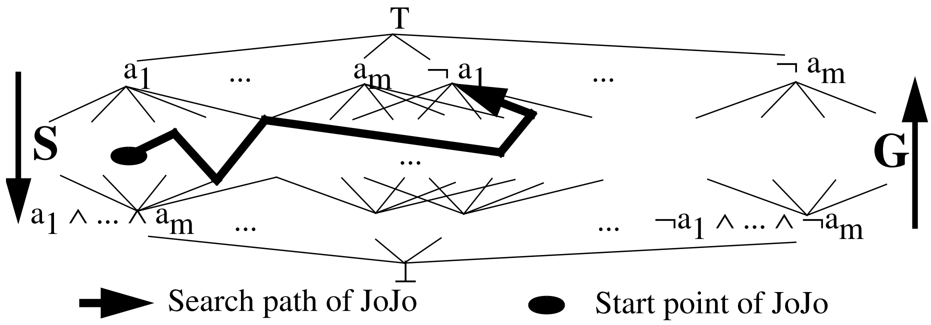

Figure 1. Specialization and generalization as different search directions in the lattice of conjunctions.

Because in general it cannot be determined which search direction is the better one, we developed the procedure JoJo which integrates both search directions into one procedure similar to the version space algorithm.[3] The next chapter sketches the main ideas of JoJo and in chapter three we characterize the four-step procedure which can be used to refine given rule sets according to new examples.

2 The Main Ideas of JoJo

JoJo is a procedure which learns rules from examples. It integrates generalization and specialization into one heuristic search process. The procedure can start at an arbitrary point in the lattice of complexes and generalize and specialize as long as the quality or correctness of the descriptions regarded can be improved, i.e. until a local optimum can be found, or the search resources (e.g., computation time) are not consumed.

2.1 Choice of the Starting Points

1. The problem can be compared with the problem of multi-colinearity of some variables.
2. This is a simple consequence of the duality principle of lattices.
3. The two search directions and their advantages and disadvantages are discusses in [7].

Procedures which can only specialize have a predefined starting point for the search. They must start as generally as possible, because a possible solution could not be found if it is more general than the starting point. Similiarly, procedures working only with specialization must start as specifically as possible. JoJo is able to start at an arbitrary point in the lattice because it can use both search directions.
A starting point can be described by two parameters, its vertical position (the length of the description) and its horizontal position (the chosen attributes).
Criteria for choosing a vertical position

- An expert can approximate the possible length of the rules or has some experience from earlier program runs.
- Rules are produced randomly for every length and the distribution of their quality[1] is used to decide the position.
- The procedure starts with a small sample or very limited resources and arbitrary positions in order to find a good starting point.
- The starting point is randomly chosen. In the average case this is no worse than always starting with the bottom or top element as other procedures do.
- Heuristic: Few positive examples and maximal-specific descriptions indicate long rules, whereas few negative examples and maximal-general descriptions indicate short rules.

Criteria for choosing a horizontal position

- If the vertical position is chosen, the premises with the highest correlation to the goal concept (or the combination of premises if this is not to expensive) can be selected.

In general, it is possible to carry out several program runs with different starting points. Rules which were already produced by JoJo or other algorithms can be used as starting points for further refinement and improvement.[2]

2.2 Search Process in the Lattice

The gist of JoJo consists of three components: a *generalizer*, a *specializer*, and a *scheduler*.
The *generalizer* computes, validates and orders the descriptions which can be reached by the next generalization step using a predefined generalization strategy and a predefined preference criterion (g-preference). An example of a simple generalizer is H-RELAX [6]:

- Conjunctions are generalized by deleting a premise.
- The g-preference applied is:[3]

$$1 - \frac{\text{number of covered negative examples} + 0{,}5}{\text{number of all covered examples} + 0{,}5}$$

The *specializer* computes, validates and orders the descriptions which can be reached by the next specialization step using a predefined specialization strategy and a predefined preference criterion (s-preference). An example of a simple specializer is:

- Conjunctions are specialized by adding a premise.

---

1. A possible quality criterion is the average correctness of the rules with the same length.
2. It is possible to check rules for overspecialization when that rules are produced by an algorithm using specialization as search direction.
3. 0,5 is added in order to prevent division by zero and to favor rules which cover more examples when several rules do not cover any negative example.

- The s-preference applied is equal to the g-preference.

It is evident that other generalizers or specializers with different strategies and preference criteria are possible.

The *scheduler* selects the next description out of the set of all possible generalizations and specializations using a predefined (total) t-preference. An example of a simple scheduler is:

- Specialize, if the error rate of the rule is higher than the specified threshold.
- Otherwise, choose the best generalization if a possible generalization exists, i.e. a generalization with allowable error rate.
- Otherwise stop.

The scheduler would prefer maximal general (but correct) descriptions.[1]

2.3 Creating a Rule Set with JoJo

Given is a set of positive and negative examples. JoJo searches for a first general and correct rule "r". Then the covered positive examples are deleted and the procedure is repeated until the number of remaining positive examples is less than a specified threshold.

3 Incremental Refinement of Rules with JoJo

In the following we extend the ideas presented above to form a learning procedure which works incrementally (cf. [10], [4]). It modifies given rules according to additional given examples. The input is a set of rules describing a hypothesis and a set of old and new positive and negative examples for the hypothesis described by the rules. Its task is to transform the set of rules so that it covers all positive and no negative examples. Alternatively, some lower thresholds for errors of the rules can be given to the algorithm. The output of the algorithm are the refined rules and the example set. Additional examples can lead to a further application of JoJo. JoJo searches for a new set of rules which is *correct*, *complete*, *non-redundant*, and (if necessary) *minimal*.

3.1 Correctness

The *first* step modifies overly general rules, i.e. rules which cover too many negative examples. An incorrect rule is replaced by a set of rules covering all positive examples which are covered by the incorrect rule but which does not cover the negative ones.

By specializing initial general descriptions examples (special descriptions) can be transformed into more general descriptions. One possibility is to specialize the given rules as long as they do not cover the new negative examples. A rule could be specialized by adding an additional premise to its premisses. The problem of this approach is that the addition of one attribute involves a minor syntactical but a major semantical change of the rule. In the case of boolean attributes the number of the covered cases are halved. In the case of non-boolean attributes, the change is even much more drastic.[2] In addition, each specialization procedure deals with the problem of overspecialization. A specialization in an i-th step can allow some previous specializations to be undone. Therefore, each learning procedure applying specialization requires additional pruning of its results to prevent overspecialization

---

1. Depending on the chosen preferences and strategies, there is a danger that loops can arise. If the value of the preference criterion has to increase for every step completed this danger does not exist.
2. (Cardinality of the range of the attribute)$^{-1}$

Alternatively, a rule can be replaced by a set of rules which covers a subset of the examples which were covered by the old rule. Therefore, we apply JoJo for the sub-space which is defined by the incorrect rule.

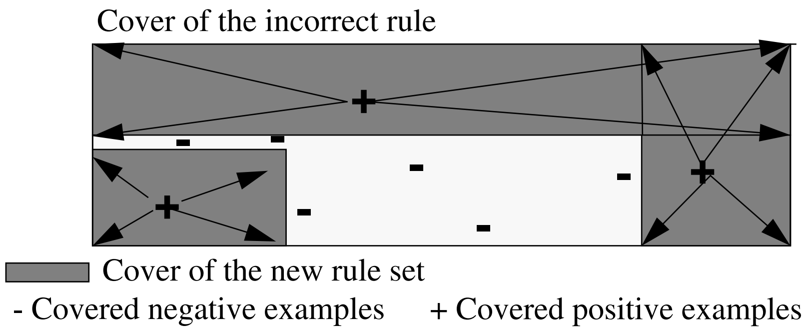

Figure 2. Specializing an incorrect rule by a set of rules.

This is done for each incorrect rule. Then the incorrect rules are replaced by the corresponding rule sets.

3.2 Completeness

After this step every rule is correct in the sense that no rule covers more negative examples than is specified by the treshold. In a ***second*** step, the set of rules must be completed. This is done by applying JoJo to the set of uncovered positive example and all negative examples. JoJo computes new rules which are correct. These rules are computed as long as the number of positive examples which remain uncovered is larger than that which is specified by a threshold.

3.3 Non-Redundancy and Minimality

In this step, rules are deleted which are more special than other rules. This is done by checking the subset relation between the sets of premisses of the rules. A rule is more special than a second one if its set of premises is a superset of the second one. In a fourth (optional) step a ***minimal set of complexes*** is computed. A minimized set of rules has the same cover as the original one but consists of a minimal number of complexes. Because this problem is NP-complete we apply the following heuristic search:

1. Look for the best rule of the rule set (i.e. the one which covers most examples).
2. Remove all positive examples which are covered by this rule.
3. Add the best rule to the final rule set.
4. If more examples remain than are specified by a threshold go to step 1.

4 Implementation and Empirical Evaluation of JoJo

JoJo is implemented in C and available under Sun-Unix and MS-DOS. It is integrated into the RJ-environment [9] which preprocesses unknown-values, ordinal, multi-valued, and continuous attributes. It has been tested with several data sets of the machine learning library and with data sets of the ESPRIT project StatLog (data sets with more than 10.000 objects and 40 attributes). The major result of these tests is its ability to learn very brief, i.e. general, descriptions of classes (cf. [8]) than algorithms like CN2 or C4.

5 Conclusion

The paper introduced a new approach for learning rules by examples which uses a more flexible heuristic search strategy than other algorithms which only generalize very specific or specialize very general descriptions. JoJo can start with arbitrary rules and generalize or specialize them as long as the quality of the rule which is being regarded is improved. The algorithm JoJo is extended to a four-step closed-loop learning procedure which allows the stepwise refinement of given rule sets according to new examples.

Acknowledgement

We thank Jörg Klein and Monika Zickwolff for helpful and fruitful discussions.

References

1. A. Blumer, A. Ehrenfeucht, D. Haussler, and M. K. Warmuth: Ocam´s Razor, In Information Processing Letters, vol 24, 1987, pp. 377-380.
2. J. Cendrowska: PRISM: An algorithm for inducing modular rules. In *Int. J. Man-Machine Studies,* vol. 27, 1987, pp. 349-370.
3. P. Clark and R. Boswell: Rule Induction with CN2: Some recent Improvements. In *Proceedings of the European Workshop on Machine Learning (EWSL´91)*, March 6-8, Porto, Portugal, 1991, pp. 151-163.
4. C. Decaesteckert: Incremental Classification: A multidisciplinary viewpoint. In *Proceedings of the Conference Symbolic-Numeric Data Analysis and Learning*, Versailles, September 18-20, 1991, pp. 283-295.
5. D. Fensel and J. Klein: A new approach to rule induction and pruning. In *Proceedings of the 3rd International Conference on Tools for Artificial Intelligence (ICTAI´91)*, Palo Alto, November 11-13, 1991.
6. D. Fensel and J. Klein: Solving a Generalization Task by Generalization: RELAX, H-RELAX, and I-RELAX. Three Algorithms for Rule Induction and Pruning. In Forschungsbericht des Institut für Angewandte Informatik und Formale Beschreibungsverfahren der Universität Karlsruhe (TH), no 232, Januar 1992.
7. D. Fensel: A Comparison and Integration of Generalization and Specialization. In research report, Institut für Angewandte Informatik und Formale Beschreibungsverfahren, University of Karlsruhe, no 249, January 1993.
8. D. Fensel and M. Pechowski: An Evaluation of B-RELAX and JoJo. In research report, Institut für Angewandte Informatik und Formale Beschreibungsverfahren, University of Karlsruhe, March 1993.
9. D. Fensel, J. Klein, and U. Neubronner: RJ - An Environment for Learning from Example. In research report, Institut für Angewandte Informatik und Formale Beschreibungsverfahren, University of Karlsruhe, February 1993.
10. J. H. Gennari, P. Langley, and D. Fisher: Model of Incremental Concept Formation. In *Artificial Intelligence*, vol. 40, no. 1-3, September, 1989, pp. 11-62.
11. T.T.T. Huyen and H.T. Bao: A method for generating rules from examples and its application. *In Proceedings of the Conference Symbolic-Numeric Data Analysis and Learning*, Versailles, September 18-20, 1991, pp.493-504.
12. E. J. McCluskey: Minimizing of Boolean Functions. In *Bell System Tech. Journal*, vol. 35, no. 5, November 1956, pp. 1417-1444.
13. T.M. Mitchell: Generalization as Search. In B. Webber et.al. (eds), *Readings in Artificial Intelligence*, Tioga Publishinh Co., Palo Alto, 1981.
14. R. S. Michalski, I. Mozetic, J. Hong and N. Lavrac: The Multi-Purpose Incremental Learning System AQ15 and its Testing Application to Three Medical Domains. In *Proceedings of the 5th National Conference on AI (AAAI-86)*, Philadelphia, PA, August 11-15, pp. 1041-1045, 1986.
15. J.R. Quinlan: Simplifying decision trees. In Gaines et.al. (eds.), *Knowledge-Based Systems, vol. 1*, Academic Press Ltd, London, 1988, pp. 239-252.
16. J. R. Quinlan: Learning Logical Definitions from Relations. In *Machine Learning*, vol 5, no 3, 1990, pp. 239-266.
17. J.R. Quinlan: Probabilistic Decision Trees. In *Machine Learning. An Artificial Intelligence Approach, vol. III*, Y. Kodratoff et.al. (eds.), Morgan Kaufmann Publisher, San Mateo, CA, 1990.