ARTIFICIAL INTELLIGENCE IN DESIGN

W5

IJCAI 93

INTERNATIONAL JOINT CONFERENCE ON ARTIFICIAL INTELLIGENCE

CHAMBERY SAVOIE FRANCE

13th INTERNATIONAL JOINT CONFERENCE ON ARTIFICIAL INTELLIGENCE

ARTIFICIAL INTELLIGENCE IN DESIGN

→ W5

Organizer :
Boi Faltings

IJCAI 93

13th International Joint Conference on Artificial Intelligence

Workshop on Artificial Intelligence in Design

Saturday, August 28th, 1993
Chambéry, France

Working Notes

Workshop Committee:

Boi Faltings, EPFL Lausanne (Chair)
John Gero, University of Sydney
Leo Joskowicz, IBM T.J.Watson Research Center
Pearl Pu, University of Connecticut
Gerhard Schmitt, ETH Zurich
Brian Williams, Xerox PARC

Table of Contents

An Application of Constructive Induction to Engineering Design

Janusz Wnek, Ryszard S. Michalski
Center for Artificial Intelligence
George Mason University
4400 University Dr.
Fairfax, VA 22030
(703) 993-1719
{wnek, michalski} @aic.gmu.edu

Tomasz Arciszewski
Civil Engineering Department
Wayne State University
5050 Anthony Wayne Dr.
Detroit, MI 48202
(313) 577-3766
Arciszewski@MTS.cc.wayne.edu

ABSTRACT

The paper presents a method for applying constructive inductive learning to conceptual engineering design. The method allows a problem-oriented transformation of the representation space spanned over the initially given attributes. This process involves a generation of new more problem-relevant attributes, and an abstraction or removal of the less relevant ones. The search for new attributes is based on the analysis of iteratively generated hypotheses, hence the method is called *hypothesis-driven constructive induction,* or HCI. The applicability of the method to engineering problems is illustrated by the problem of learning design rules for wind bracings in tall buildings. The developed program, AQ17-HCI, was used for learning four different types of rules: design recommendation rules, standard rules, avoidance rules and infeasibility rules. The learned rules achieved an average predictive accuracy 96.7% on unseen examples. An analysis of the rules by a domain expert has also indicated their high comprehensibility. The results obtained indicate a great promise of the method for the investigated application and engineering design in general.

Introduction

At present, the development of various knowledge-based design tools is considered crucial in engineering to improve the design and manufacturing productivity. However, the development of these tools for practical purposes requires the acquisition of knowledge about complex, often not completely understood problems. Traditional methods of manual knowledge acquisition are inadequate in engineering, and machine learning is the solution to the present knowledge acquisition bottleneck.

However, most existing machine learning inductive methods search for a hypothesis in a description space defined by a priori given attributes. The produced hypothesis thus involves only attributes selected from the original ones. If the original attributes are insufficiently relevant to the problem at hand, then it is not possible to produce a high quality hypothesis, measured as empirical error rate. In contrast to such *selective induction* methods, a *constructive induction method* performs a problem-oriented transformation of the original description space (Michalski 1978; Rendell 1985; Utgoff 1986; Muggleton 1987; Matheus 1989; Pagallo & Haussler 1990; Wnek & Michalski 1991). Such a transformation aims at generating new, more problem-relevant attributes, and producing a description space in which it is possible to determine a high quality inductive hypothesis.

Constructive induction methods may employ different strategies for constructing new descriptors (attributes, relations, functions used for characterizing given entities). Based on the way new descriptors are generated, existing systems can be divided into four categories: data-driven (DCI), hypothesis-driven (HCI), knowledge-driven (KCI), and multistrategy (MCI) (Wnek & Michalski 1993b). The DCI methods analyze and explore input data, specifically, interrelationships among attributes, examples, concepts, etc., in order to determine new descriptors. The HCI methods determine new descriptors by analyzing recursively generated inductive hypotheses. The KCI methods apply expert-provided domain knowledge to construct and/or verify new descriptors. Finally, the MCI methods combine different approaches and methods for constructing new descriptors.

The method presented here belongs to the class of HCI methods. Other HCI methods include BLIP (Morik 1989; Wrobel 1989), FRINGE (Pagallo & Haussler 1990). HCI methods are often a part of multistrategy constructive induction methods that combine HCI with other methods. For example, STABB combines DCI & HCI (Utgoff 1986). Other systems, such as, Duce (Muggleton 1987), CITRE (Matheus 1989), CLINT (De Raedt & Bruynooghe 1991), involve HCI & KCI.

The proposed method uses a rule representation of the generated descriptors and hypotheses earlier version of the method implemented in AQ17-HCI was successfully applied to several DNF- problems and compared with other systems in terms of prediction accuracy, convergence to a de: accuracy, and complexity of descriptions (Thrun et al. 1991; Wnek & Michalski 1991, 1993al consistently outperformed symbolic selective methods implemented in AQ15 and C4.5 (I backpropagation and a genetic algorithm based classifier system, as well as constructive induc methods such as FRINGE, GREEDY3, and GROVE.

The earlier version of AQ17-HCI used only the *rule agglomeration operator* for constructing attributes. The operator performed a selection of the best rules from a hypothesis and assembled t into a description of a new attribute. This paper presents a substantial extension of this method, w involves five operators: condition agglomeration, value agglomeration, rule agglomeration, attri extraction, and value extraction.

The HCI method was applied to a conceptual design domain exemplified by the proble learning design rules for wind bracings in tall buildings. Conceptual design is an early stage i design process whose objective is to analyze needs, limitations, and available knowledge to pro one or several feasible abstract descriptions of the engineering system being designed. Some wor already been done on the application of machine learning to conceptual design knowledge acquis (Gero 1988, 1992; Arciszewski & Rossman 1992) but the application of a rule learning system t on constructive induction is a novel one.

The method was evaluated from two perspectives: 1) machine learning, to show its advan over selective methods, and 2) engineering design, to show its applicability in structural d knowledge acquisition. The first evaluation involves a comparison with the C4.5—decision tre AQ15—decision rule learning systems using two empirical error rates, the second one, inv interpretation of the acquired knowledge, along with changes in the representation space, by a h expert.

Hypothesis-Driven Constructive Induction: HCI

The hypothesis-driven constructive induction (HCI) method combines an inductive rule lea algorithm AQ with a procedure for iteratively transforming a representation space. In each iteratio method changes the representation space by adding new attributes, removing insufficiently rel attributes, and/or agglomerating values of some attributes into larger units. The quality c hypothesis generated in each iteration is evaluated by applying the hypothesis to a subset of tra examples. The set of training examples prepared for a given iteration is split into the primary set set), which is used for generating hypotheses, and the secondary set (the S set), which is use evaluating the prediction accuracy of the generated hypotheses. Figure 1 presents a diagram illust the HCI method.

In the implemented system, AQ17-HCI, the input consists of training examples of one or concepts, and background knowledge about the attributes used in the examples (which specifie types and legal value sets). For the sake of simplicity, let us assume that the input consists of pc examples, E^+ and negative examples, E^-, of only one concept. If there are several concepts to examples of each concept are taken as positive examples of that concept, and the set-theoretical of examples of other concepts is taken as negative examples of that concept.

The method consists of two phases. Phase 1 determines the representation space by a proc iterative refinement. In each iteration, the method prepares training examples, produces rules, ev their performance, modifies the representation space, and then projects the training examples i new space. This phase is executed until the *Stopping Condition* is satisfied. This condition re that the prediction accuracy of the learned concept descriptions exceeds a predefined threshold, o is no improvement of the accuracy over the previous iteration. Phase 2 determines final c descriptions in the acquired representation space from the complete set of training example output consists of concept descriptions, and definitions of attributes constructed in Phase 1. Bel detailed description of both phases, and the basic modules of the method.

Phase 1 consists of six modules. The first, "Split of Examples" module, divides positive and negative training examples into the primary set, P, and the secondary set, S (in the experiments the split was according to the ratios 2/3 and 1/3, respectively). The set of primary positive (negative) examples is denoted P^+ (P^-), and the set of secondary positive (negative) examples is denoted S^+ (S^-). Thus $P = P^+ \cup P^-$, and $S = S^+ \cup S^-$. The primary training set, P, is used for initial rule learning, the secondary set, S, for an evaluation of intermediate rules, and total set, $P \cup S$, is used for the final rule learning (in Phase 2).

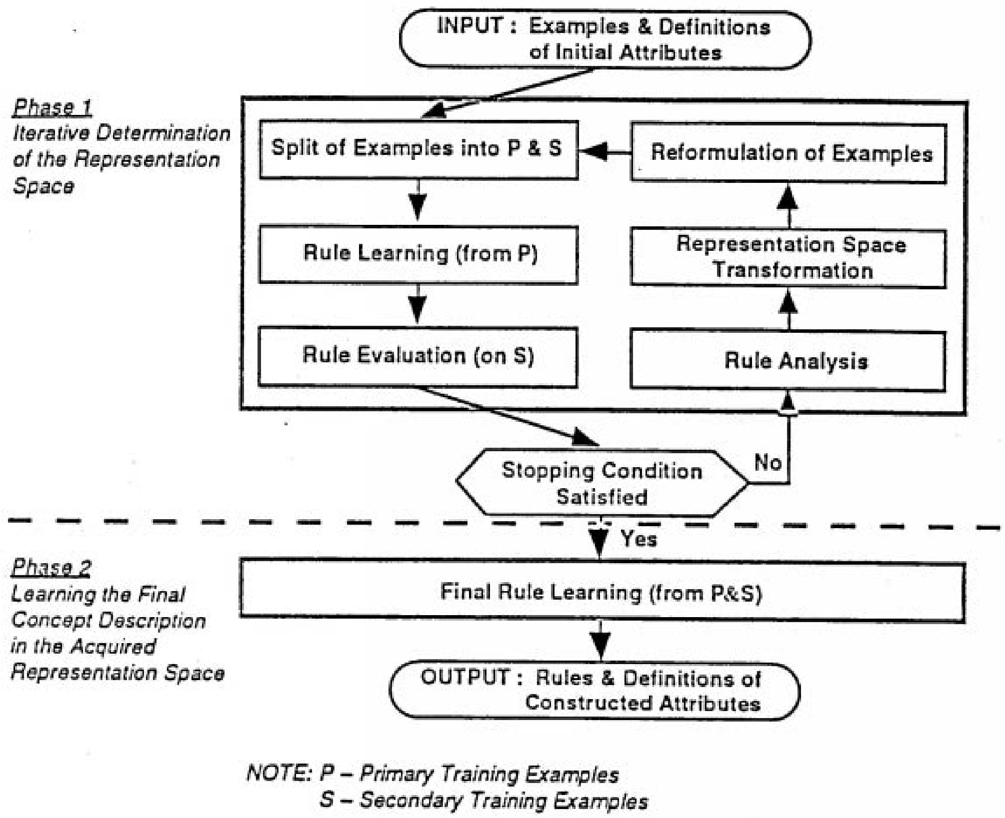

Figure 1. Hypothesis-driven constructive induction learning algorithm.

The "Rule Learning" module induces a set of decision rules for discriminating P^+ from P^-, i.e., a cover COV (P^+/P^-) of positive primary examples against negative primary examples. This is done by employing the AQ15 inductive learning program (Michalski et al., 1986). The program is based on the A^q algorithm for solving general covering problem (Michalski, 1973).

The "Rule Evaluation" module estimates the prediction accuracy of the rules by applying them to the secondary training set, S. The accuracy of the rules in classifying the examples from S is determined by the ATEST procedure implemented in the AQ15 program (Reinke, 1984). If the Stopping Condition criterion is not satisfied, the control passes to the "Rule Analysis" module, otherwise, it passes to Phase 2.

The "Rule Analysis" module determines an *admissible ruleset*, i.e. the set of rules that have sufficient strength according to rule strength measure (Wnek 1993). The HCI method works iteratively. Each iteration generates a complete and consistent set of rules, i.e., a ruleset that covers all positive examples and none of the negative examples. In order to speed up the process of determining strong patterns, and avoid searching through rules that are weak and/or have low validity, the me selects the admissible ruleset.

The "Representation Space Transformation" module analyzes the rules in this ruleset to deter desirable changes in the representation space. It removes redundant or insignificant attributes, mo existing attributes (by attribute value agglomeration), and generates new attributes. This inv detecting and evaluating patterns according to pattern strength measure. After patterns are sele new attributes can be defined. A new attribute definition consists of a name, a defining expression a similarity measure for assigning values. The attribute is assigned a unique name after the conc was created from. The defining expression is the related pattern. The measure can result in assi real or discrete values from the closed interval 0 to 1. In the simplest case, value 1 is assigned pattern is satisfied, and value 0 otherwise. More sophisticated measures may express the dis between an instance and a pattern in real values. For example, Bala, Michalski & Wnek (1992 rule pattern attributes with a real-valued similarity measure for the task of texture recognition.

The "Example Reformulation" module projects all training examples into the new represent space, and the whole inductive process is repeated.

In Phase 2, for each concept learned, the final ruleset is determined by applying the "Rule Lear module to all training examples projected into the final representation space determined in Phase 1

The transformation of an attribute set is done through its reduction or expansion. Represent space reduction can be accomplished in three ways: (1) elimination of an attribute value, (2) elimin of an attribute, (3) elimination of a part of the representation space. The first two ways are explanatory and are implemented as *attribute and value extraction* operators. The third way ass that the part of the representation space may be considered irrelevant. The representation reduction simplifies the learning process or at least leads to a simplification of the descriptions lea

Another kind of change in the representation space is performed by adding new attribut extending the domains of existing attributes. This is done through three operators: agglomeration, condition agglomeration, and rule agglomeration. The application of a parti operator involves a search for one of three types of patterns in learned rules: condition-pattern, pattern, and rule/set-pattern. Condition-pattern is detected if the analyzed hypothesis con conditions that repeatedly involve the same values of a single attribute. The value agglomer operator than combines those values into a single value and adds it to the domain of the attribute. pattern is searched among conjunctions of conditions (rules). After detecting such a patter condition agglomeration operator creates new attribute and uses the pattern in attribute defin Ruleset-pattern is a selection of the best performing rules from a hypothesis.

Structural Design Domain

Domain of conceptual design of wind bracings in steel skeleton structures of tall buildings is desc by eight attributes. The description of the design problem and the wind bracing itself was develop Mustafa (1989). The attributes used are sufficient to describe various types of flat frame, trus truss-frame bracings which are appropriate for buildings in the six to thirty-story height range. attributes are based on a general description of wind bracings in tall buildings propos Arciszewski (1985). The attributes can be divided into three major components: 1) a description building for which a given bracing is intended (design case or design requirements), 2) a descript the wind bracing structural system, and 3) an evaluation of the structural worth of a given bracing for the design case considered. Each instance in this domain relates the design requireme the selection of components of a wind bracing structural system and the structural worth c system.

The attributes, *Number of Stories*, *Bay Length*, and *Importance Factor* (the effect of wind zc structural design), describe the design case (design requirements) considered. Attributes, *J Number of Bays*, *Number of Vertical Trusses*, *Number of Horizontal Trusses*, describe the stru system of wind bracing itself. The decision attribute, *Unit Steel Weight*, identifies the nominal va the relative unit weight of the steel structural system of a wind bracing described by the other attributes.

Decision rules are generated according to the four values of the decision attribute *Unit Steel Weight.* Values of the unit weight are high, average, low, and infeasible. Therefore, the decision rules which specify designs with low unit weights are called *Recommendation Rules.* Similarly, decision rules which produce average unit weights are called *Standard Rules,* and rules which produce high unit weight are called *Avoidance Rules.* Rules producing infeasible wind bracings are called *Infeasibility Rules.*

Examples of the structural designs were prepared by Mohamad Mustafa (1989) as part of his doctoral research on an engineering methodology of automated knowledge acquisition. All detailed design assumptions regarding loads, dimensions, steel grade, etc., were determined in cooperation with practicing structural designers.

Induction of Decision Rules

The learning system conducted a multi-stage knowledge acquisition process. In the first stage, decision rules were generated in the original representation space. The analysis of the rules revealed that the *Importance Factor* attribute plays insignifficant role in the description and therefore the attribute was eliminated from the training set. This result was not entirely unexpected, because wind intensity, represented by *Importance Factor,* is rarely a decisive factor in the structural shaping of wind bracings. In the next stage of knowledge acquisition, the system constructed three new attributes: two of which were constructed using condition agglomeration operator and one using value agglomeration operator. These new attributes were defined by the system as follows:

```
IF      (Number of Stories IS 6) &
        (Number of Vertical Trusses IS 0)
THEN    c1 = 1    ELSE c1 = 0
```

```
IF      (Number of Stories IS 12..24) &
        (Joints ARE Rigid or Mixed)
THEN    c2 = 1    ELSE c2 = 0
```

```
IF      (Number of Vertical Trusses IS 1..3)
THEN    c3 = 1    ELSE c3 = 0
```

In the third stage of knowledge acquisition, the system constructed one new attribute with five values using rule agglomeration operator. Each value is in a standard normal form, i.e., in the form of a disjunct of several complexes (decision rules). This new attribute is defined as follows:

```
IF      (c1 = 1) &
        (Number of Bays IS 1 or 2) &
        (Number of Horizontal Trusses IS 0..2)
        OR
        (c1 = 1) &
        (Bay Length IS 30) &
        (Number of Horizontal Trusses IS 0..2)
        OR
        (c1 = 1) &
        (Bay Length IS 30) &
        (Number of Bays IS 1 or 2) &
        THEN    c4 = 1
```

```
IF      (Number of Stories IS 12..30) &
        (Number of Bays IS 3) &
        (Number of Vertical Trusses IS 0) &
        (Number of Horizontal Trusses IS 0..2)
        OR
```

(c2 = 1) &
(Joints ARE rigid or mixed) &
(Number of Bays IS 1 or 2) &
(Number of Horizontal Trusses IS 0 or 2 or 3)
OR
(Number of Stories IS 18 or 24) &
(Joints ARE mixed) &
(Number of Horizontal Trusses IS 1 or 3)
THEN c4 = 2

IF (c3 = 1) &
(Number of Stories IS 6..24) &
(Number of Bays IS 2 or 3)
OR
(Number of Stories IS 12..30) &
(Joints ARE hinged) &
(Number of Horizontal Trusses IS 1..3)
THEN c4 = 3

IF (Number of Stories IS 30) &
(Joints ARE rigid or mixed) &
(Number of Bays IS 1)
THEN c4 = 4

IF (none of the above formulas is satisfied)
OR
(more than one formula is satisfied)
THEN c4 = 5

All values of the constructed attribute (c4) have a clear structural engineering meaning and c interpreted in the terms of structural shaping of wind bracings. For example, the value 1, (c4 = 1 be interpreted in the following way:

In order to obtain a low unit weight for a six-story building, avoid designing a wind bracing as a rigid frame and the following three combinations of structural attributes:	(Number of Stories IS 6) & (Number of Vertical Trusses IS 0)
(1) a single one-bay or two one-bay structural system with or without one or two horizontal trusses.	(Number of Bays IS 1 or 2) & (No H. Trusses IS 0 or 1 or 2)
OR (2) with a wide bay and with or without one or two horizontal trusses	(Bay Length IS 30) & (No H. Trusses IS 0 or 1 or 2)
OR (3) with a wide bay and as a single one-bay or two one-bay structural system	(Bay Length IS 30) & (Number of Bays IS 1 or 2)

In the fourth stage of learning, the system produced four classes of decision rules, includir *Avoidance Rules*, five *Standard Rules*, three *Recommendation Rules*, and one *Infeasibility R* an example, Avoidance Rules are presented and their domain interpretation provided:

Avoidance Rules (AR):

AR1: IF (c4 IS 1)
THEN *a high unit weight of bracing should be expected*

The interpretation is given above as the explanation of the constructed attribute *c4*

<u>AR2:</u> IF (Number of Bays IS 1 or 3) &
(Number of Vertical Trusses IS 0) &
(Number of Horizontal Trusses IS 1) &
(c4 IS NOT 2) &
(c4 IS NOT 4)
THEN *a high unit weight of bracing should be expected.*

Avoid designing a one- or three-bay wind bracing in the form of a rigid frame, use one horizontal truss and make sure that the value of constructed attribute c4 is neither two nor four.

<u>AR3:</u> IF (Number of Stories IS 12) &
(Bay Length IS 20) &
(Joints ARE mixed) &
(Number of Bays IS 2) &
(Number of Horizontal Trusses IS 3)
THEN *a high unit weight of bracing should be expected.*

Avoid designing twelve-story buildings with a narrow bay with wind bracings in the form of two one-bay rigid frames and three horizontal trusses.

<u>AR4:</u> IF (Number of Stories IS 18) &
(Number of Vertical Trusses IS 0) &
(Number of Horizontal Trusses IS 0) &
(c4 IS NOT 2)
THEN *a high unit weight of bracing should be expected.*

Avoid designing eighteen-story buildings with wind bracings in the form of rigid frames and make sure that the constructed attribute c4 is not equal two.

<u>AR5:</u> IF (Number of Stories IS 18) &
(Joints ARE mixed) &
(Number of Bays IS 1) &
(c4 IS NOT 2)
THEN *a high unit weight of bracing should be expected.*

Avoid designing eighteen-story buildings with wind bracings in the form of a single one-bay rigid frame with horizontal truss or trusses and make sure that the constructed attribute c4 is not equal two.

Performance Analysis

The previous section analyzed the method from the viewpoint of comprehensibility and usefulness of acquired rules. In this section, the performance accuracy of constructive induction-based learning system in the area of structural design knowledge acquisition is analyzed. The performance can be formally measured by various empirical error rates, which are determined through tests. In each test, a learning system uses a given body of examples to make predictions about other known examples which have not been included in its input. Each test can then be compared to a real-life situation, when a designer uses a decision support system to predict the structural worth of a wind bracing to minimize its weight. Therefore, empirical error rates are highly relevant to both machine learning research, which is concerned with the performance of learning systems, and to structural design, which is concerned with the optimal decision making.

Two empirical error rates were used: 1) *the reclassification error rate*, and 2) *the leaving-one-out error rate* (Duda & Hart 1973; Weiss & Kulikowski 1991; Arciszewski, Dybala & Wnek 1992). The reclassification error rate, sometimes called the apparent error rate, is calculated on all training examples. Since any unknown example is tested, the error rate is usually too optimistic, but it provides lower limit for other types of empirical error rates. It also gives a fast evaluation of the available data, especially when learning from real-world data.

The leaving-one-out is a preferred method for testing on relatively small samples of c (Lachenbruch & Mickey 1968; Efron 1982; Weiss & Kapouleas 1989). For a sample size (n classifier is trained on (n-1) examples and tested on the remaining one. This is repeated (n) times f given sample.

In our experiment, the sample size (n) was 336. The error rates were determined for selec induction, C4.5 decision tree learning system and AQ15 decision rule learning system, hypothesis-driven constructive induction AQ17-HCI decision rule learning system. Individual e rates are shown in the Table 1.

Table 1. Comparison of Empirical Error Rates for Decision Trees and Decision Rules.

Method	Reclassification Error Rate %	Leaving-One-Out Error Rate %
C4.5 (unpruned)	1.8	10.6
C4.5 (pruned)	10.7	11.0
AQ15	0.0	4.5
AQ17-HCI	0.0	3.3

Both, unpruned and pruned decision trees as constructed by C4.5 do not fit data complete Therefore, small 1.8% (6 examples out of 336) reclassification error rate made by an unpru decision tree assures the quality of data and sets a benchmark for this kind of learning. Relatively la error rate of the pruned decision tree, 10.7%, suggests that pruning does not improve performa accuracy in this domain. Both AQ15 and AQ17-HCI do not generate any reclassification errors means that data is consistent, and moreover, it assures that the changes in the representation space, not lead to inconsistency in the domain description.

The leaving-one-out testing method yields higher but more realistic error rates than reclassification error rate method. There is a significant improvement in performance (more than 30 percent) between the system based on selective induction represented by C4.5 and AQ15, and system based on constructive induction.

Conclusions

The presented method of constructive induction changes the problem representation space by analyz and abstracting inductive hypotheses, rather than by directly combining different attributes or apply expert-provided knowledge to construct new attributes. This way the search for new attributes is v efficient, although it is limited by the representational formalism in which the method is implemente

The performance analysis shows that constructive induction method is more effective in terms empirical error rates than traditional selective induction based on the C4.5 learning algorithm. approximately 300 percent decrease of the error rate is quite significant for such applications as ci engineering. It would be desirable to produce and analyze learning curves for empirical error rates both systems, to learn more about their performance in a multistage automated knowledge acquisit process, and this work is planned.

The results presented in this paper show that the use of constructive induction in structural des knowledge acquisition is feasible. The decision rules produced are relatively simple and their structu interpretation is possible, although not always easy, particularly when complex constructed attribu are used. The changes in the representation space are acceptable to human experts. These changes similar to those associated with the growing human understanding of a given domain, and theref can also stimulate the process of human learning.

The application of the HCI method is one of many stages in a long process of structural desi knowledge acquisition. In this process, the knowledge representation stage was particularly difficult required extensive study and the cooperation of experts. The identification of relevant attributes a their nominal values began in the early seventies, and these attributes have undergone numerous changes and modifications before a final acceptable set was produced.

The agglomeration and extraction operators used by the HCI method contribute to a fast evaluation of the quality of attributes used in problem description and suggest possible modifications. For example, the attribute extraction operator eliminated the *Importance Factor* attribute and domain experts fully agree with it. The value agglomeration operator suggests simplification of the *Number of Verical Trusses* attribute by combining three values into one.

The feasibility study was conducted in the area of structural design knowledge acquisition, and therefore all conclusions produced are valid only in this area. However, it could be inferred by analogy that similar results, in terms of clarity of decision rules, good performance, and proposed changes in representation should be expected in other areas of civil engineering.

Machine learning research has already reached a fair level of maturity, and has resulted in various experimental and commercial learning systems. These systems could be used in civil engineering to improve productivity in knowledge acquisition and the development of knowledge-based decision support systems. However, the application of learning systems is currently delayed by the lack of a methodology of their use, and the development of this methodology is becoming crucial for further progress. This paper provides some initial methodological results, but much more needs to be done. Any work on the methodology of applying learning systems to civil engineering will be difficult, but this is a challenge which must be met.

Acknowledgements

This research was conducted in the Center for Artificial Intelligence at George Mason University. The Center's research is supported in part by the Advanced Research Projects Agency under Grant No. N00014-91-J-1854, administered by the Office of Naval Research, and the grant No. F49620-92-J-0549, administered by the Air Force Office of Scientific Research, in part by the Office of Naval Research under Grant No. N00014-91-J-1351, and in part by the National Science Foundation under Grant No. IRI-9020266.

References

Arciszewski, T. and Rossman, L. (Eds.), *Knowledge Acquisition in Civil Engineering*, American Society of Civil Engineers, 1992.

Arciszewski, T., Dybala, T. and Wnek, J., "Method for Evaluation of Learning Systems," *Journal of Knowledge Engineering "Heuristics* to appear 1992.

Arciszewski, T., "Decision Making Parameters and their Computer-Aided Analysis for Wind Bracings in Steel Skeleton Structures," *Advances in Tall Buildings*, Van Nostrand, 1985.

Bala, J., Michalski, R.S. and Wnek, J., "The Principal Axes Method for Noise Tolerant Constructive Induction," *Proceedings of the Ninth International Conference on Machine Learning Conference*, Scotland, pp. 20-29, 1992.

DeRaedt, L. and Bruynooghe, M., "Constructive Induction By Analogy: A Method to Learn How to Learn?" *Proceedings of EWSL-89*, pp. 189-200, Montpellier, France: Pitman, 1991.

Duda, R. and Hart, P., *Pattern Classification and Scene Analysis*, New York: Wiley, 1973.

Efron, B., "The Jackknife, the Bootstrap and Other Resampling Plans," in *SIAM*, Philadelphia, PA, 1982.

Gero, J.S. (Ed.), *Artificial Intelligence in Engineering: Design* , Elsevier, New York, 1992.

Gero, J.S. (Ed.), *Proceedings of the Second International Conference on Artificial Intelligence in Design '92*, Pittsburgh, 1992.

Lachenbruch, P. and Mickey, M., "Estimation of Error Rates in Discriminant Analysis," *Technometrics*, pp. 1-110, 1968.

Matheus, C., "Feature Construction: An Analytic Framework and Application to Decision Trees," *Ph.D. Dissertation*, Urbana-Champaign: University of Illinois, 1989.

Michalski, R.S., "AQVAL/1--Computer Implementation of a Variable-Valued Logic System VL_1 and Examples of its Application to Pattern Recognition," *Proceedings of the First International Joint Conference on Pattern Recognition*, Washington, DC, pp. 3-17, 1973.

Michalski, R.S., "Pattern Recognition as Knowledge-Guided Computer Induction," *Rep*
Computer Science, University of Illinois, Urbana, 1978.

Michalski, R.S., Mozetic, I., Hong, J. and Lavrac, N., "The Multi-Purpose Incremental Learn
System AQ15 and its Testing Application to Three Medical Domains," *Proceedings of AAAI-86*,
1041-1045, Philadelphia, PA, 1986.

Morik, K., "Sloppy Modeling," in K. Morik (Ed.), *Knowledge Representation and Organizatio*
Machine Learning. Springer-Verlag, 1989.

Muggleton, S., "Duce, and Oracle-Based Approach to Constructive Induction,"*Proceedings*
IJCAI-87, pp. 287-292, Milan, Italy, 1987.

Mustafa, M., "Engineering Methodology of Automated Knowledge Acquisition: Structu
Application," *Ph.D. Proposal*, Civil Engineering Dept, Wayne State University, Detroit, 1989.

Pagallo, G. and Haussler, D., "Two Algorithms that Learn DNF by Discovering Relev
Features," *Proceedings of the 6th International Machine Learning Workshop*, pp. 119-123, Itha
NY: Morgan Kaufmann, 1989.

Quinlan, J.R., "Induction of Decision Trees," *Machine Learning, 1*, 81-106, 1986.

Rendell, L., "Substantial Constructive Induction Using Layered Information Compressi
Tractable Feature Formation in Search," *Proceedings of IJCAI-85*, pp. 650-658, 1985.

Thrun, S.B., Bala, J., Bloedorn, E., Bratko, I., Cestnink, B., Cheng, J., DeJong, K.
Dzeroski, S., Fahlman, S.E., Hamann, R., Kaufman, K., Keller, S., Kononenko, I., Kreuziger,
Michalski, R.S., Mitchell, T., Pachowicz, P., Vafaie, H., Van de Velde, W., Wenzel, W., Wnek
and Zhang, J., "The MONK's Problems: A Performance Comparison of Different Learni
Algorithms," *Technical Report*, Carnegie Mellon University, December 1991.

Utgoff, P.E., "Shift of Bias for Inductive Concept Learning," in R.S. Michalski, J.G. Carbonell
T.M. Mitchell (Eds.), *Machine Learning: An Artificial Intelligence Approach* (Vol. II). Los Altos, C
Morgan Kaufmann, 1986.

Weiss, S.M. and Kapouleas, I., "An Empirical Comparison of Pattern Recognition of Neural N
and Machine Learning Classification Methods," *Proceedings of IJCAI-89*, Detroit, MI, pp. 781-7
1989.

Weiss, S.M. and Kulikowski, C.A., *Computers that Learn*, Morgan Kaufmann, San Mateo, C
1991.

Wnek, J. and Michalski, R.S., "Hypothesis-driven Constructive Induction in AQ17: A Method a
Experiments," *Proceedings of the IJCAI-91 Workshop on Evaluating and Changing Representatio*
K. Morik, F. Bergadano and W. Buntine (Eds.), pp. 13-22, Sydney, 1991.

Wnek, J., "Hypothesis-driven Constructive Induction," *Ph.D Dissertation*, School of Informati
Technology, George Mason University, 1993.

Wnek, J. and Michalski, R.S., "Comparing Symbolic and Subsymbolic Learning: Three Studie
in *Machine Learning: A Multistrategy Approach*, Vol. IV, R.S. Michalski and G. Tecuci (Ed
Morgan Kaufmann, San Mateo, CA, to appear 1993a.

Wnek, J. and Michalski, R.S., "Hypothesis-driven Constructive Induction in AQ17: A Method a
Experiments," Special Issue on Representation, *Machine Learning*, to appear 1993b.

Wrobel, S., "Demand-Driven Concept Formation," in K. Morik (Ed.), *Knowledge Representati*
and Organization in Machine Learning. Berlin Heidelberg: Springer Verlag, 1989.