86-13

The Inductive Acquisition of Temporal Knowledge

Kaihu Chen
Artificial Intelligence Laboratory
Department of Computer Science
University of Illinois

ABSTRACT

A methodology is proposed for the inductive acquisition of temporal descriptive rules and temporal decision rules from training instances. A distinction is made between descriptive and decision rules in order to distinguish rules driven by classifications from rules driven by reward/punishment. Refinement (specialization/generalization) operators for the temporal domain are given, and various heuristics in rule invocation and incremental rule generation are discussed. The method has potential applications in the acquisition of temporal knowledge in many realtime domains.

TRACK: Science

TOPICS: Knowledge Acquisition and Learning

ADDRESS:

Kaihu Chen
Department of Computer Science
University of Illinois at Urbana–Champaign
1304 West Springfield
Urbana, Illinois 61801

1. Introduction

The acquisition of characteristic descriptions from preclassified examples has been an intensively investigated field. Most of the previous researches [Michalski, 1983; Quinlan, 1983; Mitchell, 1977] were limited to the acquisition of timeless, static classification rules. The SPARC system [Michalski & Dietterich, 1985] addressed the problem of discovering rules that qualitatively predict the plausible continuation of a sequence of events in semi-temporal sequential domains, but the program actually falls into the part-to-whole generalization category instead of the instance-to-class generalization category [Michalski, Ko & Chen, 1986]. Development in temporal logic [Vere, 1983] addressed the representation and planning in realtime, but said nothing about inductive acquisition of temporal knowledge.

Here we propose a methodology for the acquisition of characteristic descriptions [Michalski, 1983] from preclassified temporally described training instances, called **episodes**. The introduction of temporal knowledge into an inductive learning system demand that the system must be able to address the following issues:

- Handling of noisy data which is compounded by the introduction of temporal knowledge. Since it is unrealistic to expect noise-free data from realtime domains, it is imperative that the system is capable of handling noisy data in an appropriate way. Moreover, since perfect class descriptions may not exist or are too costly to find, it is desirable to have the capability to generate "imperfect" rules with variable precisions, and repair the these rules incrementally until satisfaction.
- Since the acquisition of temporal knowledge is an expensive process (due to its complexity), incremental rule repair mechanism [Michalski, 1985] becomes a much more attractive alternative than the batch approach.
- Temporal domain permits the generation of almost limitless number of new temporal attributes from the existing data. This amount to open-ended constructive induction where new attributes can be dynamically generated. The inductive algorithm must be able to estimate the "interestingness" of the new attributes/relations in order to avoid wasted efforts in pursuing unpromising ones.

- With the increase of complexity of the domain, it becomes less likely that the user will be able to twiddle the parameters of the system into exactly the right setting in order to find the optimum solutions for arbitrary input data. The system should be able to learn to adjust its own inductive biases with respect to the incoming data.
- The amount of potential information introduced by the temporal knowledge is immense. It is thus imperative to harness the complexity problem by exploiting constraints hidden in the data and background knowledge. Previous efforts in this direction, such as PROMISE [Baim, 1982], and ESEL [Cramm, 1983], are essentially preprocessing filters of the inductive engine that eliminate uninteresting attributes and training instances. The need for incremental learning requires that such filtering mechanism be integrated into the inductive algorithm.

The methodology proposed here is an attempt to address the acquisition of temporal knowledge in the form of descriptive rules and decision rules (described later) from examples, while with the problems mentioned above in mind.

2. Formulation of the Problem

A training instance, called an **episode**, is given as a list of temporal descriptions of certain objects. Objects are assumed to be uniquely identifiable through time. The following is an example of a training instance:

$$<e_1 \ e_2 \ e_3 \ ...> \rightarrow Class_i$$

Each event e_i of an episode is specified as a triplet $\{d_i, \ t_{si}, \ t_{fi}\}$, where t_{si} and t_{fi} are absolute temporal marks that specify the starting time and finish time of the description d_i. The event description d_i is an expression of a representation language powerful enough to describe relationships between objects, such as first order predicate logic or annotated predicate logic [Michalski, 1983] expressions.

Two types of rules are distinguished: **descriptive rules** and **decision rules**. Both types of rules assume the same syntactic form, except that the right hand side of a descriptive rule is a classification label, while the right hand side of a decision rule is an **action**. The semantical difference between

descriptive rules and decision rules can be distinguished by the way their merits are calculated. The merit of a rule is a measurement of its interestingness to the inductive engine. The merit of a descriptive rule is bound to effectiveness that the rule covers the training instances. Such effectiveness measurements include the simplicity, generality (among other criteria), and the entropy (described later) of the rule. The merit of a decision rule is bound to the result of the action part (the right hand side) of the rule. For example, the merit of the following decision rule

$$is-a-card(object_1) \rightarrow \text{generalize the expression using refinement operator } G_5$$

is tied to the merit of the rule generated using refinement operator G_5. Decision rules, like descriptive rules, can be generalized or specialized in order to generate rules with the highest possible merits. The merit of a decision rule can be viewed as the reward/punishment values that are used to train the system toward certain desired behavior. Decision training instances are given with its merits (the reward/punishment values) instead the classification labels as in descriptive training instances. Given such training instances, the task of the inductive engine is to discover generalized descriptions for each class/action that have the highest merits using a set of refinement (generalization/specialization) operators.

2.1. The Refinement Operators

Refinement operators can be divided into two categories: event refinement operators and temporal refinement operators. Event refinement operators are those defined on the static representational language representations. Many refinements operators, such as closing range, dropping conjunctive terms, or turning constants into variables, are well-described [Michalski, 1983] so it is not repeated here.

Temporal refinement operators derive new temporal relationships from existing data. Temporal generalization from a pair of conjuncted events e_1 and e_2 can be derived by turning the temporal marks into variables, then adding (by conjunction) the relationship between these temporal variables to the expression. For example, given two events

$$e_1=\{d_1, t_{s1}, t_{f1}\}$$

$$e_2=\{d_2, t_{s2}, t_{f2}\}$$

the following temporal attributes can be generated:

$$T_1=t_{s2}-t_{s1}$$
$$T_2=t_{s2}-t_{f1}$$
$$T_3=t_{f2}-t_{s1}$$
$$T_4=t_{f2}-t_{f1}$$
$$T_5=t_{f1}-t_{s1}$$
$$T_6=t_{f2}-t_{s2}$$

These attributes can be generalized/specialized like any other linear attributes, where refinement operators, such as closing range, can be applied. For example,

$$[T_1=5]\rightarrow Class_1$$

can be generalized to

$$[T_1>0]\rightarrow Class_1$$

which says that e_2 occurs after e_1 has finished. An example of the possible generalizations from the conjunction of e_1 and e_2 is shown below:

$$\{d_1, T_{s1}, T_{f1}\}\ \&\ \{d_2, T_{s2}, T_{f2}\}\ \&\ [T_{s2}>T_{f1}]\rightarrow Class_1$$

Temporal marks, like static structured attributes, can have its own value hierarchy. Another set of refinement operators based on this value hierarchy can be created. For example, the temporal mark **12:00 pm** can be generalized to **daytime**, and **January 10** can be generalized to **winter**.

Another temporal generalization operator, similar to the closing-gap operator for linear attributes, works by closing temporal gaps. For example, the rule

$$\{d_1, t_1, t_2\}\ \&\ \{d_1, t_3, t_4\}\rightarrow Class_i$$

can be generalized by closing the temporal gap:

$$\{d_1, t_1, t_4\} \rightarrow Class_i$$

assuming that t_4 is greater than t_1.

2.2. Constructive Operators

Information not explicitly given in the training instances can be derived with the help of deductive background knowledge. A simple form of such deductive background knowledge is the construction of new attributes from the attributes given in the training instances, such as the generation of the new attribute "parity" from integer attributes. Such **derived attributes** also exist along the time axis, notably the variation of the attribute value of a certain object over time. This is close to the discrete version of the first order derivative of the attribute, which is similar to the difference variable used in the SPARC system [Michalski & Dietterich, 1985]. Higher order derivatives or similar temporal constructive operators that manifest the temporal variation of an attribute can also be used.

3. Rule Refinement

Instead of processing training instances and generating descriptive/decision rules in batch, an incremental learning program must be able to generate new rules by modifying the existing ones in the rulebase. With the set of intermediate (possibly imperfect) descriptive rules previously learned, the program must make the following decisions when given a new training instance:

- How to select from the rulebase those rules that have the highest promise of being improved by the current training instance. This is the problem of **rule invocation**. Here we essentially assume that the number of rules in the rulebase is so great that exhaustive rule invocation is impractical.
- How to repair the invoked rules in a most efficient way. This is the problem of **rule generation**.

The basic algorithm for incremental rule repairing is shown as follows:

1) Input a training instance
2) Invoke rules from the rulebase.
3) Refine the invoked rules, then put them back into the rulebase.
4) If satisfactory rules have been found, then exit; else go to (1).

From the point of view of inductive learning, the introduction of temporal information increase the repertoire of refinement operators available to the inductive engine. Classic representation independent inductive algorithms, such as Mitchell's version space [Mitchell, 1977], are still valid, but their exhaustive nature in rule invocation and rule generation make them too inefficient to handle temporal information. In the following sections heuristics for rule invocation and rule generation are discussed.

3.1. Rule Invocation

The invocation of rules in the rulebase is decided by their merits. Merits are measured differently for descriptive rules and decision rules. The merit of a descriptive rule measures the expected improvement that can be achieved, plus certain static criteria, such as LEF [Michalski, 1986], that measures desired properties of a rule. The merit of a decision rule is tied to the merit of the outcome of the action (the right-hand side of the rule).

To decide if a descriptive rule has the prospect of being improved further, one must be able to measure the static "strength" of existing rules, and the effect of a new training instance on this measurement. In the following sections two such measurements are introduced: the **uniformness** and the **expected maximal entropy reduction** measurements.

3.1.1. Entropy-Based Rule Invocation

The purpose of rule refinement is to generate descriptive rules that satisfy predefined criteria, such as the simplicity or generality of the rule. One absolute criterion for measuring the desirability of a descriptive rule can be defined in terms of its **entropy**. The entropy of a descriptive rule R is given by the following formula:

$$entropy(R) = -p^{+}\log p^{+} - p^{-}\log p^{-}$$

where p^{+}/p^{-} are the percentage of positive/negative instances covered by R respectively. A **consistent** descriptive rule covers either all positive or all negative instances, thus has zero entropy. The expected **improvement** of a descriptive rule can be defined as the expected maximal reduction in its entropy

(Expected Maximal Entropy Reduction) as a training instance is registered. Rule invocation based on their EMER has many interesting properties. For example, consider the following rules:

$$R_1 \equiv H_1 \rightarrow Class_i{:}1000{:}10$$
$$R_2 \equiv H_2 \rightarrow Class_i{:}100{:}1$$
$$R_3 \equiv H_3 \rightarrow Class_i{:}50{:}50$$
$$R_4 \equiv H_4 \rightarrow Class_i{:}1{:}1$$

where $R_i \equiv H_i \rightarrow Class_j{:}P{:}N$ means that R_i covers P positive training instances, and N negative training instances. Intuitively, R_1 is the least interesting of the four, since it is a well–established rule with 1010 evidences. R_3 is also uninteresting since it covers equally large number of positive and negative instances. R_2 and R_4 are much more interesting because new evidence can cause large change in their entropy (large amount of information gain), thus worth spending more effort for further refinement. The EMER for the four rules are respectively:

$$EMER(R_1){=}0.0045$$
$$EMER(R_2){=}0.041$$
$$EMER(R_3){=}0.0055$$
$$EMER(R_4){=}0.057$$

which correspond closely to our intuitive notion of the "interestingness" of descriptive rules. The **interestingness of a training instance** e to a rule R can be defined as the entropy change of R caused by e. This measurement provides a way to pinpoint:

- training instances that can be important new evidences in forming new hypotheses (which cause high EMER).
- spurious training instances that could be noise which cause high EMER in rules with high confidence (which cover many training instances), these events should be either verified or deleted from the training set.
- commonplace training instances (which cause low EMER) that are forgettable by the system.

Entropy reduction can be viewed as a form of information gain. Rule invocation based on EMER thus maximizes the prospect of gaining information through refinement.

3.1.2. Uniformness–Based Rule Invocation

In light of entropy, many induction algorithms [Michalski, 1983; Quinlan, 1983; Mitchell, 1977] can be viewed as algorithms that generate entropy plateaus in the description universe. By exploiting the dependencies between the rules in the rulebase and evaluating the differences of their entropy measurements, it is possible to prognosticate the internal entropy structure of a rule, hence deciding the proper refinements that should take place.

A rule might have non–zero entropy for many different reasons:

- The rule is over–generalized. In this case further specialization is required.
- The concept to be learned is intrinsically probabilistic. In this case, further specialization is fruitless. Probabilistic concepts (assuming uniform probability distributions) can be identified as uniform rules with non–zero entropies.
- The input contains spurious data, or noise. Such data should be verified, deleted, or the concept should be treated as probabilistic.
- The concepts to be learned have intrinsically fuzzy boundaries, as is the case in many human concepts.

Rule refinement based on the uniformness measurement will eventually yield rules that are entropy plateaus. Perfectly consistent zero entropy rules can be generated if they do exist, otherwise variable precision rules still can be discovered.

3.2. Rule Generation

Rule generation can be achieved by either generating totally new rules by applying refinement operators to the old rules, or repairing incrementally the existing rules. The rule refinment mechanism described in this section applies to descriptive rules as well as to decision rules.

Some rules of thumb for rule refinement are listed below:

(1) If the number of training instances that cause non–uniformness are few, then these instances can be registered as exceptions to the rule.

(2) If the number of training instances that cause non-uniformness are moderate, then generate a local generalization of these instances and registered them as exceptions to the rule [Reinke & Michalski, 1985].

(3) If no rule repairing seems to work, then generate a completely new rule. This is detailed in the following section.

Since rule repairing by registering exceptions does not requirement reevaluation of the repaired rules, it is a very efficient form of rule refinement and is preferred whenever possible.

3.2.1. Dependency Directed Rule Generation

The dependencies between rules (of the same class) are marked by the generalization relationship between them. Given two rules R and R_G such a that R_G is a generalization of R, derivable by applying generalization operator G to R:

- an over-generalization is indicated if the difference between the entropy of R_G and R is greater than certain threshold. A generalization operator G_s that is more specific than G can be used to generate an intermediately general rule $G_s(R)$.
- an over-specialization is indicated if the entropy of R_G is the same as R. In this case certain static criteria can be used to select the one rule that achieves higher score.

Another type of dependency comes in the form of temporal/spatial associations. Two rules R_1 and R_2 can be refined to a common specialization R_S (where R_S is more specialized than both R_1 and R_2), if they are temporal/spatially associated. Temporal association is marked by the facts that the two rules in question tend to be invoked at the same time (or nearly the same the time) . Spatial association is marked by the fact that the rules tend to be invoked by the same object (or same set of objects).

4. Introspective Program

In the preceding sections we have outlined a set of rules by which the system operates on, such as

If rule R is non-uniform, then specialize R.

or rules that allow us to choose relevant attributes depends on the context:

(Hypothetical)

If the patient is Caucasian,

then check his diet habit as potential cause of heart failure.

These rules constitute the **biases** of the inductive system for rule generation. When such rules accumulate in large quantity, it becomes doubtful if a fix set of biases will allow the system to act intelligently in all situations. One solution to the problem is to allow the system to shift, or generate, its biases depending on the input data. This is possible because these biases can be expressed in the form of decision rules, which makes them syntactically no difference from the training instance. The same induction engine can thus be used on **ground level data** (the training instances) in order to discover domain rules, as well as on **meta–data** in order to discover meta–rules (the biases). To collect these **meta–data**, the set of **actions** that can be adopted by the system must be explicitly defined. Such system actions may include the application of certain refinement operators to certain ground level rules, or the and assignment of credits/blames to certain attribute/relations. The merit of these meta–level decision rules are tied to the merit of the rules they eventually generated.

5. Conclusions

The applications of inductive inference to realtime domains have been hampered by many problems. Notably the presence of noisy data, the lack of expressive power in the representation languages, the lack of strong heuristics, and the batch nature of many previous systems. As the complexity of the problem domain increases, the capability to learn incrementally becomes increasingly important. The methodology presented here is an attempt to address these problems, and lay a framework for future researches in the inductive acquisition of knowledge in temporal domains.

Much of the immense complexity of temporal domain is tackled by the entropy–based criteria for rule invocation and rule generation. The information theoretic view of rule learning provides heuristics that minimize the entropy of the system, thus maximize information gain. The choice of inductive biases

for different context can be automated by representing inductive biases in the form of decision rules. Inductive biases can be acquired from experience through inductive learning, the same way that ground-level rules are learned. It is thus possible to achieve a form of "introspection" without the stratification of incoherent knowledge representations and control structures.

ACKNOWLEDGEMENTS

The author is grateful to Professor Ryszard Michalski for his comments and suggestions. This work was supported in part by the Defense Advanced Research Project Agency under grant N00014-K-85-0878, Office of Naval Research under grant N00014-82-K-0186 and National Science Foundation under grant DCR-84-06801.

REFERENCES

[1] P. Baim, "The PROMISE Method For Selecting Most Relevant Attributes For Inductive Learning Systems," ISG 82-1, UIUCDCS-F-82-898, Department of Computer Science, University of Illinois, Urbana, IL, September 1982.

[2] S. Cramm, "ESEL/2: A Program for Selecting the Most Relevant Training Events for Inductive Learning," ISG 83-1, UIUCDCS-F-83-901, Department of Computer Science, University of Illinois, Urbana, IL, January 1983.

[3] T. Dietterich and R. S. Michalski, "Learning to Predict Sequences," ISG 85-9, UIUCDCS-F-85-938, Department of Computer Science, University of Illinois, Urbana, February 1985.

[4] Forbus, K. "Qualitative Process Theory", Ph.D. thesis, M.I.T. , Cambridge, 1984.

[5] Michalski, Ryszard S. "A Theory and Methodology of Inductive Learning", In: **Machine Learning, An Artificial Intelligence Approach**, R. S. Michalski, J. G. Carbonell and T. M. Mitchell, eds. Tioga, Palo Alto, CA, 1983, pp. 461-481.

[6] Quinlan, J. R. *Learning Efficient Classification Procedures and their Application to Chess End Games.* In: **Machine Learning, An Artificial Intelligence Approach**, R. S. Michalski, J. G. Carbonell and T. M. Mitchell, eds. Tioga, Palo Alto, CA, 1983, pp. 461-481.

[7] Michalski, R. S. and J. B. Larson,. "Incremental Generation of VL1 Hypotheses: the Underlying Methodology and the Description of Program AQ11", ISG 83-5, UIUCDCS-F-83-905, Urbana, IL, 1983.

[8] Michalski, R. S., "Knowledge Repair Mechanisms: Evolution versus Revolution", Proceedings of the Third International Machine Learning Workshop, 116-119.

[9] Michalski, R. S.; Stepp, R. "Conceptual Clustering" In: **Machine Learning, An Artificial Intelligence Approach**, Volume II, R. S. Michalski, J. G. Carbonell and T. M. Mitchell, eds. Morgan Kaufmann Publishers, Inc., 1986, pp. 471-498.

[10] Michalski, R. S.; Ko, H. and Chen, K. "Qualitative Prediction: A Method and Program SPARC/G", to appear in "Expert Systems," P. Dufour and A. van Lamsweerde. eds, Academic Press Inc., 1986.

[11] Mitchell, T. M. 1977. Version Space: A candidate elimination approach to rule learning. IJCAI 5, 305-310.

[12] R. E. Reinke and R. S. Michalski, "Incremental Learning of Decision Rules: A Method and Experimentals Results," presented at the "Machine Intelligence Workshop 11," Ross Priory, University of Strathclyde, March 1985.

[13] S. A. Vere, "Planning in Time: Windows and Durations for Activities and Goals", IEEE Transaction on Pattern Analysis and Machine Intelligence, Vol. PAMI-5, No. 3, May 1983, pp246-267.

BIBLIOGRAPHIC DATA SHEET	1. Report No. UIUCDCS-F-86-964	2.	3. Recipient's Accession No.

4. Title and Subtitle

The Inductive Acquisition of Temporal Knowledge

5. Report Date 1986

6.

7. Author(s)

Kaihu Chen

8. Performing Organization Rept. No.

9. Performing Organization Name and Address

Department of Computer Science
University of Illinois
Urbana, IL 61801

10. Project/Task/Work Unit No.

11. Contract/Grant No.

NSF DCR 84-06801
ONR N000-4-82-K-0186

12. Sponsoring Organization Name and Address

National Science Foundation
Washington, DC

Office of Naval Research
Arlington, VA

13. Type of Report & Period Covered

14.

15. Supplementary Notes

16. Abstracts

A methodology is proposed for the inductive acquisition of temporal descriptive rules and temporal decision rules from training instances. A distinction is made between descriptive and decision rules in order to distinguish rules driven by classifications from rules driven by reward/punishment. Refinement (specialization/generalization) operators for the temporal domain are given, and various heuristics in rule invocation and incremental rule generation are discussed. The method has potential applications in the acquisition of temporal knowledge in many realtime domains.

17. Key Words and Document Analysis. 17a. Descriptors

Inductive Learning
Temportal Knowledge
Machine Learning
Decision Rules
Qualitative Prediction
Artificial Intelligence

17b. Identifiers/Open-Ended Terms

17c. COSATI Field/Group

18. Availability Statement	19. Security Class (This Report) UNCLASSIFIED	21. No. of Pages 15
	20. Security Class (This Page UNCLASSIFIED	22. Price