

43 1022 11/4/88

P88-1

Submitted for presentation at
the Fifth International Conference on Machine Learning
Ann Arbor, Michigan, June 12-15, 1988

CONSTRUCTIVE CLOSED-LOOP LEARNING: Fundamental Ideas and Examples

Ryszard S. Michalski
Artificial Intelligence Center
George Mason University
4400 University Drive
Fairfax, VA 22030

Phone: (703) 764-6258

and

Larry M. Watanabe
Department of Computer Science
University of Illinois
Urbana, IL 61801

ABSTRACT

Constructive Closed-Loop Learning (CCL) is a multi-strategy learning that adapts its strategy to the learning situation according to the relevance of its background knowledge. When the system's prior knowledge is adequate, but not most effective for the given task, the system attempts to (deductively) reformulate its knowledge to make it more efficient. If this knowledge is inadequate, then the system (inductively) constructs a hypothetical knowledge needed for the task. In the process of creating a plausible inductive hypothesis the system employs all its relevant background knowledge, so that it is capable of constructing new concepts or variables more relevant to the task than those present in the input descriptions (thus, it can perform *constructive* induction).

In every act of learning the system first determines which previously learned knowledge is most relevant to the task at hand, and tries to take maximum advantage of it. The knowledge gained through learning is evaluated from the viewpoint of its utility, and if sufficiently useful, is fed back to the system (thus, we have *closed-loop* learning). We describe a general method for CCL, and illustrate it with an example.

Problem area: Concept Learning
General approach: Integrated Learning
Evaluation criteria: Theoretical Analysis

CONSTRUCTIVE CLOSED-LOOP LEARNING: Fundamental Ideas and Examples

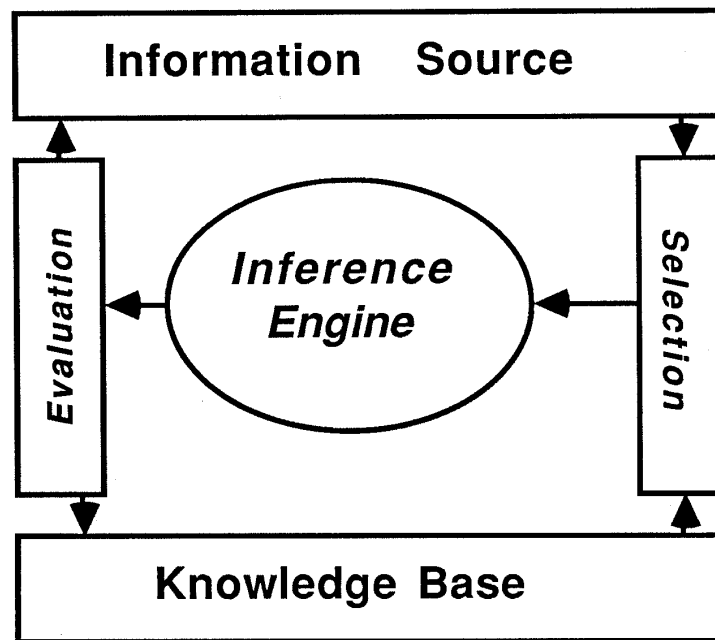
1. Introduction

Currently, there are two major directions of research in machine learning: empirical and analytical. Empirical learning performs inductive inference from the data supplied, without use (or need) of extensive amount of background knowledge. Results of learning are hypotheses that may be incorrect, and thus have to be tested on new examples. The fact that this approach does not need (or use) much background knowledge is both an advantage and a disadvantage. On one side, since it primarily relies on the examples given to the system, and examples are often easy to obtain, this approach is easy to apply to many practical problems. On the other side, because it utilizes little background knowledge, it can hardly be applied to learning in complex, knowledge intensive domains. Thus, an application of empirical methods is most advantageous in the areas where it is difficult to provide the system with much relevant background knowledge, but examples are easy to obtain (e.g., for object recognition, diagnostic problems, discovery tasks, etc.).

Analytical learning, and in particular, its most popular form explanation-based learning (EBL; Mitchell, Keller, & Kedar-Cabelli, 1986; DeJong and Mooney, 1986), requires background knowledge that is sufficient to explain a given example in terms of this background knowledge. This means, that the system must be supplied in advance with enough knowledge that it can construct a proof that a given example is a logical consequence of the background knowledge. Once such a proof is constructed, the system can use it to reformulate, or operationalize its knowledge, so that it can perform better on subsequent examples. If the system does not have adequate prior knowledge, then it cannot explain the example, and thus breaks down. Because it relies on deductive inference, it cannot in principle create fundamentally new knowledge. Thus, this approach seems to be most advantageous in the areas where it is not too difficult to handcraft the

needed domain knowledge into the system, and the task of learning is to improve or operationalize knowledge the system started with (e.g., in the task performance refinement, automatic programming, skill acquisition, etc.).

This paper describes an approach, called *constructive closed-loop learning* (CCL) that adapts the method of learning to the task at hand. Thus, it may employ an empirical learning, an analytic learning, or a combination of the two, according to the relevance of the background knowledge to the task at hand. In order to explain CCL in more detail, we first present a general architecture of constructive-closed loop learning (Figure 1).



Architecture of a Constructive Closed-loop Learning System

Figure 1.

The diagram in the figure 1 presents learning as a process of transforming the information (or knowledge) obtained from a source (a teacher, environment, or an internal source) to a new form, which is most desired for the task at hand. This process involves performing inference (by *inference engine*) on the selected parts of the source information using the learner's prior knowledge (*knowledge base*). This inference may be deductive or inductive.

The employed inference is deductive, if the knowledge base contains adequate knowledge to account for the input information, and the task is to improve in some way the knowledge in the knowledge base. The inference is inductive if the available knowledge is not adequate (insufficient, incorrect or intractable), and the system has to hypothesize a new piece of knowledge.

The *selection* module decides which part of the source information should be given attention, and which knowledge in the knowledge base is relevant to the given task of learning.

Knowledge created as a result of this process is evaluated (*evaluation module*) from the viewpoint of its usefulness, and if it passes a test, it is stored in the knowledge base for future use. The evaluation may take into consideration the estimated validity of created knowledge, its effectiveness in solving a desired class of tasks, etc. A result of evaluation may also be a demand for more information from the source (a question to the teacher, a proposition to perform an experiment, etc.)

In this learning paradigm, empirical learning and analytic learning are viewed as special cases of constructive closed-loop learning. Empirical learning represents a process that uses little background knowledge (in the learner's knowledge base), and relies mostly on the source information. The inference module in this case performs mainly inductive inference (e.g., Michalski, 1983; Michalski, 1987). Such inference is not truth-preserving, and thus the constructed knowledge must be carefully evaluated. The evaluation module is therefore heavily employed. Knowledge that passed a validity and usefulness test done by the evaluation module is fed back to the learner's knowledge base.

In analytical learning, the knowledge base of the learner is supposed to be sufficient to explain the source of information, i.e., to show that it is a consequence of the knowledge in the knowledge base. By making this demonstration, which involves deductive inference, the system can reformulate or restructure its knowledge so that it can perform better in the future (e.g., can perform more efficiently subsequent tasks). Because deductive inference is truth-preserving, the validity of results is guaranteed by the process of constructing the proof. The role of the evaluator in this case is to determine the usefulness of storing the reformulated knowledge in the knowledge base (by, e.g., estimating the trade-off between the cost of storing the knowledge and the cost of reformulating it again).

Constructive closed-loop learning (CCL) is a general learning process described by this diagram. It combines constructive induction (a knowledge-based induction; see, e.g., Michalski, 1983; Michalski, 1987), with deductive or analytic learning, and with the ability to evaluate, feed back and utilize knowledge learned previously by either process. In any act of learning, a CCL system involves all its task relevant background knowledge to test if this knowledge is adequate for explaining the new data, or, if it is not, to construct the most plausible inductive hypothesis. It may be appropriate to indicate at this point the distinction between closed-loop learning and incremental learning, widely mentioned in the machine learning literature. Incremental learning is viewed as a limited form of closed-loop learning, in which the system improves step-by step the description of the *same* concept. It cannot, like in the closed-loop learning, create or manipulate a system of concepts. Since closed-loop learning is necessary to have the above described integration of inductive and deductive learning, henceforth, for simplicity, we will call CCL alternatively also *constructive learning*. Various formal aspects of CCL and its relationship to different types of explanation are discussed by Michalski and Ko (1988), and Ko and Michalski (1988).

A CCL system is capable of knowledge accretion and tuning, as well as knowledge restructuring (Rumelhart and Norman (1978). Learning in CCL may occur at the control level as well as at the domain level. The ability to use previously learned knowledge in the current learning task has been implemented in various forms, usually rather limited, in a number of machine learning programs. Examples of such systems are AM (Lenat, 1978), MARVIN (Sammut & Banerji, 1986), LAIR (Watanabe & Elio, 1987), and TYPICAL (Haase, 1987). LAIR uses a Horn clause knowledge representation, and depends on the pedagogical order in which concepts are taught. For example, to learn the concept of cup, it must first be taught the concepts of stable, liftable, and graspable, which are used to define that concept. Systems AM and TYPICAL are discovery systems, and are essentially "closed" to the world outside their internalized domain of mathematics. MARVIN's paradigmatic mode of learning is by asking questions. This is useful in some situations, but in others, it is difficult for the system to react flexibly to data from the environment. For example, in many environments there are observations or facts from which concepts can be learned, but there may not be a teacher around to answer the many questions MARVIN generates during its learning process. Despite these shortcomings, these systems have laid much of the groundwork for constructive learning.

The ability to learn at the control level as well as the domain level is important for performance. Typically, the aim of a learning system is to acquire a knowledge base for some purpose, e.g., diagnosis or question-answering in some domain. A system that learns at the domain level may learn all of the knowledge necessary to explain the facts that it observes, but this knowledge may be inefficient for answering certain queries about new situations in the domain. To overcome this problem, a two-tiered learning scheme can be used. The system records the examples used in learning, and the queries given during its performance. In addition, the system observes how it learns from examples, and how it handles its queries. These observations are then used by the system to form inductive hypotheses about the adequacy of its knowledge, which will enable the system to optimize its learning process and performance. Given a meta-level knowledge base that allows the system to carry out transformations on its knowledge base, the system can restructure its knowledge.

This view of learning at the control level contributes to an issue in explanation-based learning: what do these systems learn and how correct is the result of learning? One can view the proofs constructed during explanation as examples of "good" or "important" things to do, and the regression proof methods as meta-level knowledge about how to transform a knowledge base to a desired form. These examples can then be fed to an inductive learning program to learn which transformations are desirable and which are not.

Thus, the results of EBL learning, as considered from this viewpoint, may be incorrect. For example, one could give the EBL system very unusual examples of some concepts. In such cases it might not be worthwhile to create special-purpose, compiled proofs, and the performance of the resulting knowledge base might be worse than the original!

2. Methodological Aspects of Constructive Learning

A constructive learning system is being developed in the context of the Intelligent Explorer (IEX) Project, which is to construct an autonomous system capable of learning, reasoning and planning in a partially known or unknown environment. One of the aims of this project is to investigate integrated learning systems in a naval ships domain. The constructive closed-loop learning system integrates knowledge accretion, tuning and restructuring with deductive and inductive transformations on its knowledge base. Background ideas for constructive learning were introduced in (Michalski and Ko; 1988).

The system starts with background knowledge in the form of rules. Examples are presented to the system in the form of inductive assertions (Michalski, 1983):

$E \Rightarrow K$

where E is a description of an example, and K is the name of a concept.

Using its background knowledge, the system attempts to form an explanation of the example, i.e., to show that :

$BK \quad |> \quad E \Rightarrow K$

The symbol " $|>$ " is the specialization operator stating that the right-hand-side is a logical consequence of the left-hand-side; in other words, that $BK \Rightarrow (E \Rightarrow K)$ is a theorem.

If the explanation succeeds, then an explanation-based type method is used to create a new rule that corresponds to an operationalized or compiled version of the explanation. In contrast to EBL systems, however, this new rule is not added immediately to the knowledge base. Instead, it is passed to an auxiliary knowledge base for further evaluation.

If the explanation fails, then the system either a) creates a new inductive hypothesis, or b) does inductive restructuring of its background knowledge. We will consider a) first.

The new hypothesis is created by using the incremental version of the INDUCE algorithm (Michalski, 1983) ? to create a new hypothesis H such that

$BK \ \& \ H \quad |> \quad E \Rightarrow K$

where H is in the form $Description \Rightarrow K$

This new hypothesis is evaluated and, if it passes a quality test, is added to the background knowledge together with an estimate of its certainty and frequency of use. Further examples may cause the INDUCE algorithm to be reactivated with the new definition of K .

In addition to creating a new hypothesis, the system may do inductive restructuring of its background knowledge. This essentially corresponds to applying generalization or other inference operators to the background knowledge so that they result in a restructured knowledge base, BK' , such that

$$BK' \vdash E \Rightarrow K$$

We will describe briefly how this method can be used for two of the generalization operators described in (Michalski, 1983). The first is a modified version of the general constructive induction rule: if there are two rules of form

$$P_1 \& P_2 \& \dots \& P_{i-1} \& P_i \Rightarrow K$$

$$P_1 \& P_2 \& \dots \& P_{i-1} \& P_j \Rightarrow K$$

and there is a concept Q such that

$$P_i \Rightarrow Q$$

$$P_j \Rightarrow Q$$

then try to replace the two rules by

$$P_1 \& P_2 \& \dots \& Q \Rightarrow K$$

This step requires use of memory of negative examples for each concept that is learned. The examples for P_i are checked against the new rule for P_i to see if they are consistent, in which case the new rule can be added.

The dropping conditions operator can also be used for knowledge restructuring. For example, if we have two rules

$$P_1 \& P_2 \& \dots \& P_{i-1} \& P_i \Rightarrow K$$

$$P_1 \& P_2 \& \dots \& P_{i-1} \& P_j \& P_{j+1} \& \dots P_n \Rightarrow K$$

Then if

$$P_1 \& P_2 \& \dots \& P_{i-1} \Rightarrow K$$

does not contradict the negative examples for K, then the two former rules can be replaced by the latter rule.

After a new hypothesis is formed or the knowledge base is restructured, the example can be explained:

$$BK \& H \vdash E \Rightarrow K$$

The explanation is compiled, evaluated and stored in the auxiliary knowledge base.

During the course of learning, the same explanation may be created many times in the auxiliary knowledge base. Each time the explanation is recreated, this information is recorded.

Similarly, during performance, explanations are created, recreated, and stored in the auxiliary knowledge base. Based on this record, rules created or used many times may be activated for consideration during the performance task. Other rules that prove to be of limited value may be deleted from the auxiliary knowledge base. This allows the system to do selective, deductive restructuring of its knowledge base to meet performance requirements. In contrast to extant EBL systems, the usage record provides some evaluation of the performance of the compiled rule. This helps to avoid an overloading of the knowledge base with many rules of limited value.

4. A Simple Example of Constructive Learning

This section presents a very simple example of how the CCL system would learn in the SHIPS domain currently being studied in the IEX project. The system starts with certain background knowledge of the SHIPS domain that it has acquired by being told. Suppose that the task of a ship is to scout enemy bases while avoiding enemy ships. Finding intelligence about bases is assumed to be valuable enough to risk the ship being subjected to a hostile fire from the land. However, the risk associated with being fired on by an enemy ship is assumed to be too high. Initially, the system might be told or might learn from examples that:

If something is detected by radar on the water that fires at the ship
Then don't scout it

If something is detected by radar on the land that fires at the ship
Then scout it

If something is detected visually on the water and it has a Iranian flag
Then don't scout it

Later, the system might learn also such rules as

If something has an Iranian flag
Then it's an enemy

If something fires at the ship
Then it's an enemy

We will consider two types of learning: learning by modifying rules, and learning by modifying background knowledge.

Learning by modifying rules

At this point, the system looks at the rules that include the descriptors "flag" and "fires." It finds that one or the other of these occur in all of the rules for the concept "scout." Therefore, it attempts to modify these rules, testing if the new rules are consistent with all remembered examples:

If an enemy is detected by radar on the water
Then don't scout it

If an enemy is detected by radar on the land
Then scout it

If an enemy is detected visually on the water
Then don't scout it.

Finally, the system might learn from examples some rules for ships:

If a large elongated physical object is detected visually or by radar on the water
Then it's a ship

If something fires on the ship from the water
Then it's an enemy ship

If something is detected on land
Then it's not a ship.

These new rules would allow it to reformulate the previous rules to:

If something is an enemy ship
Then don't scout it.

If something is an enemy but not a ship
Then scout it.

Learning by modifying background knowledge

The next example shows how hypotheses can be formed by modifying the prior knowledge rather than the current hypotheses. Suppose the ship is given a rule about things to avoid:

If something is detected on the water by sonar that fires on the ship
Then don't scout it.

One could alter the current hypotheses by adding this rule as another disjunct in the concept "don't scout." However, if the CCL system determined that its knowledge of "scout" was more certain than its knowledge of ships, then it might modify its rule for "ship":

If a large, elongated object is detected visually, by sonar, or by radar on the water
Then it's a ship.

If the above rule for "ship" is consistent with the remembered examples for "ship," it can be added to the knowledge base. Adding this rule to the knowledge base is sufficient to account for the fact that an object on the water that fires on the ship should not be scouted.

The system might now go on to observe other positive and negative examples of things to be scouted. Many of these examples can already be accounted for by the learned knowledge of the system. In this case, the system uses analytic learning to demonstrate that the examples are accounted for by its knowledge. These proofs are stored in an auxiliary knowledge base. The system might find that some proofs are used very often; for example, it might find that the rule:

If something fires on the ship from the water
Then don't scout it,

accounts for 85% of the examples. In this case, the frequency of use of this rule is enough to justify adding it to the knowledge base, even though it is not needed to explain any observations. Note that this rule is the same as one of the original rule, minus the redundant condition that the enemy be detected by radar. The system therefore adds this rule to the knowledge base, augmented by a tag that notes that this rule was formed to meet

Lenat, D. B. (1983). The role of heuristics in learning by discovery: three case studies. In Michalski, R. S., Carbonell, J. G. and Mitchell, T. M. (Eds.), *Machine Learning: An Artificial Intelligence Approach*, Vol I. Tioga: Palo Alto.

Michalski, R. S. (1983). A theory and methodology of inductive learning. In Michalski, R. S., Carbonell, J. G. and Mitchell, T. M. (Eds.), *Machine Learning: An Artificial Intelligence Approach*, Vol I. Tioga: Palo Alto.

Michalski, R. S. (1987) Concept Learning, *Encyclopedia of Artificial Intelligence*, E. Shapiro (ed.), John Wiley & Sons, January.

Michalski, R.S. and Ko, H. (1988). On the nature of explanation or Why did the wine bottle shatter, A paper accepted for the AAAI Workshop on Explanation-based Learning, Stanford University, March 1988.

Mitchell, T. M. (1977). Version spaces: a candidate elimination approach to rule learning, *Fifth International Joint Conference on Artificial Intelligence*, (pp 305-310). Morgan Kaufmann Publ, Cambridge, Massachusetts.

Mitchell, T. M., Keller, R. & Kedar-Cabelli, S. (1986). Explanation-based generalization: a unifying view, *Machine Learning Journal*, 1, 11-46.

Rumelhart, D. E. & Norman, D. A. (1978). Accretion, tuning, and restructuring: three modes of learning. In J. W. Cotton and R. Klatzky (Eds.), *Semantic Factors in Cognition*. Erlbaum Associates: Hillsdale, New Jersey.

Sammur, C., & Banerji, R. B. (1986). Learning concepts by asking questions. In R.S. Michalski, J.G. Carbonell, and T. M. Mitchell (Eds.), *Machine Learning: An Artificial Intelligence Approach*, Vol II. Tioga: Palo Alto.

Segre, A. M. (1987). On the operationality/generality trade-off in explanation-based learning. *Proceedings of the Tenth International Joint Conference on Artificial Intelligence*, (pp 242-248). Milano, Italy: Morgan Kauffman.

Stepp, R. E. (1987). Concepts in conceptual clustering. *Proceedings of the Tenth International Joint Conference on Artificial Intelligence*, (pp 211-213). Milano, Italy: Morgan Kauffman.

Watanabe, L. & Elio, R. (1987). Guiding constructive induction for incremental learning from examples. *Proceedings of the Tenth International Joint Conference on Artificial Intelligence*, (pp 293-296). Milano, Italy: Morgan Kauffman.