Pre-publication draft of a paper which appeared in the Proceedings of the Seventh European Knowledge Acquisition for Knowledge-Based Systems Workshop (EKAW '93), pages 263-275

Control, capabilities and communication: Three key issues for machine – expert collaborative knowledge-acquisition

Geoffrey I. Webb
School of Computing & Mathematics, Deakin University
Geelong, Vic. 3217, Australia

Abstract. Machine learning and knowledge elicitation are different but complementary approaches to knowledge acquisition. On the face of it there are large potential gains to be reaped from the integration of these two knowledge acquisition techniques. Machine-expert collaborative knowledge acquisition combines these approaches by placing the machine learning system and the human expert as partners in the knowledge-acquisition task. This paper examines three key issues facing machine-expert collaborative knowledge-acquisition—where should control reside, what capabilities should each partner bring to the task and how should the partners communicate?

1 Introduction

Most approaches to knowledge acquisition fall into one of the two categories—knowledge elicitation or machine learning. Knowledge elicitation, the most widely practiced form of knowledge acquisition, involves the encoding of knowledge that is expressed by a human expert. Machine learning involves formal automated analysis of example cases from which a knowledge-base is induced.

Knowledge elicitation and machine learning from examples each have distinct and unique capabilities. Human experts are able to draw upon book learning (the accumulated wisdom of thousands of years of human endeavour), practical experience, general and common-sense knowledge and are situated in the social and operational context in which the knowledge is to be employed. A machine learning system is able to perform exhaustive logical and/or statistical analysis of large sets of examples and may be free from conceptual bias derived from book learning and social context.

Thus, a machine learning system may:

- consider solutions which a human expert's biases lead him to overlook; and
- induce knowledge from examples in contexts in which human expertise is unable to provide insight, for example, because no one has previously developed solutions to the problem under consideration.

Conversely, human expertise may:

- provide solutions in circumstances for which the available example cases are not adequate for a machine learning system to derive a suitable solution; and
- discriminate between potential solutions on the basis of knowledge external to the formal considerations available to the machine learning system, for example, that, although a solution is correct, it will not be acceptable to some of the intended

To appear in *Proceedings of the 1993 European Workshop on Knowledge Acquisition for Knowledge-based Systems*, Toulous, France

users.

Further, a machine learning system is only able to operate within the constraints of the problem description with which it is provided. If a particular factor is necessary for correct analysis of a domain and the machine learning system is not provided with knowledge of that factor, it is not possible for the machine learning system to produce a correct analysis. The *domain model* (set of primitive knowledge representation terms, predicates and operators with which the system is provided) defines the space of possible solutions that it can generate. If an adequate solution does not lie within that space then it is not possible for the machine learning system to find one.

By contrast, human experts are accomplished at extending and refining domain models as circumstances demand and have ready access to multiple sources of insight that may help guide such a process. Examples of such sources of insight include:

- skills and experience with problem formalisation and description;
- extensive domain specific and general knowledge and experience; and
- access to other knowledge sources, such as books and other experts, as required.

In view of the differing and (on the face of it) complementary capabilities of knowledge elicitation and machine learning, there is considerable potential for gain through collaboration between a machine learning system and a human expert during knowledge acquisition. Such collaboration will be called Machine – Expert Collaborative Knowledge-acquisition (MECK).

A number of previous approaches to MECK have been developed.

Teiresias [5] and a similar system developed by Smith, Winston, Mitchell & Buchanan [6] support refinement of a knowledge-base using analysis of system performance on an individual task at a time. By analysing failures on a task, these systems are able to suggest to a user forms of refinement that may improve performance. For example, they may propose the addition of a new rule with a specific consequent. However, the interaction with the expert is limited to the machine learning component providing suggestions to the human expert. There is no provision for the human expert to advise the machine learning component. Further, the approaches are only applicable to the final refinement of substantially developed knowledge bases. They cannot be applied to the initial development of a knowledge base.

A number of systems have been developed that enable a human expert to assist a decision tree induction system [7, 8]. These systems support structured induction [9] allowing the human to use the machine learning system to construct individual decision trees for intermediate reasoning steps. However, the level of cooperation during the construction of a single decision tree is minimal. The user may alter the test at a node or the induction system parameters that apply at that node and have the system develop a new subtree from that point. However, it is not possible to use the induction system to refine (as opposed to replace) complete trees that the expert deems to be flawed.

MECK should be distinguished from supervised learning [10] in which a machine learning system is able to ask questions of a human expert in order to confirm or deny hypotheses derived through analysis of example cases. In MECK, both parties are able to pose and critique hypotheses for inclusion in the knowledge-base.

This paper examines a number of key issues for MECK—who should have control and how should it be managed; what capabilities should each partner contribute; and how should the partners communicate. As appropriate, solutions to

these issues are illustrated by reference to *Einstein*, a fully implemented MECK tool [11, 12].

Einstein operates in the context of the acquisition of production rules. Unlike the decision tree based techniques, where production rules may be derived from the decision trees once knowledge acquisition is complete, production rules are used at all stages of the knowledge acquisition process. Quinlan [13] identifies two reasons for developing production rules rather than decision trees—

- production rules are widely-used and well-understood; and
- decision trees can be difficult for a human to understand and modify whereas production rules are modular and hence relatively transparent.

While Quinlan presented these reasons as motivation for transforming decision trees into production rules, they are even more compelling when considering collaboration during knowledge acquisition. The wide-spread use and understanding of production rules lowers the barriers to the initial use of the approach. Their relative modularity increases the ease of use. A further advantage of the use of production rules is that existing production rule based expert systems may be refined using the methodology.

2 Control

Central to any collaborative effort is the issue of control. Where does the ultimate authority lie? If there is disagreement, who makes the final decision? Who sets the agenda? A related issue is that of initiative. Which partner can initiate what activities in what contexts?

As it is the human partner that will use the knowledge-base that is developed (or at least, is situated closest to the context in which it will be used) and who will (at least in the short term) judge its value, it should be the human partner that has the final authority in all cases. If the machine learning system is able to override the human partner's decisions, it will diminish the human partner's feeling of ownership and decrease the likelihood of satisfaction with the final product.

However, although ultimate control should be in the hands of the human expert, both parties should be able to take the initiative in appropriate circumstances. For example, when the machine learning system identifies opportunities to significantly improve the knowledge-base it is appropriate for it to bring these to the human partner's attention. However, they should be presented as suggestions with the ultimate decision as to whether they should be incorporated in the knowledge-base being left to the expert.

To avoid frustration, the human expert should have the ability to perform arbitrary changes to the knowledge-base at any stage during the collaborative interaction. There is nothing more frustrating than believing that a change is essential but being unable to directly implement it.

These principles apply not only to control over operations on the knowledge-base under development, but also control over the management and planning of the collaboration itself.

However, the degree of control that each partner can exert will necessarily be constrained both by the capabilities at his or its disposal and the communication that is possible. These issues are explored in the following sections.

3 Machine learning capabilities

One important factor that will greatly influence the success of a platform for MECK will be the precise range of operations that the machine learning system is empowered to perform.

Whereas most research into machine learning has examined techniques for creating new knowledge-bases, any non-trivial collaboration for knowledge acquisition is going to require that the machine learning system be able to refine successive drafts of the knowledge-base. Such refinement should be able to occur after and take full account of any form of input that the human expert may provide.

A wide variety of machine learning approaches to knowledge-base refinement have been developed [5, 6, 14, 15, 16, 17, 18, 19, 20, 21, 22, 23, 24]. However, many of these allow only very limited forms of modification. Some are restricted to deleting rules [17, 23] from the knowledge-base. Some are limited to modifying rule weights [14, 20]. SEEK2 [15, 16] is restricted to modifying existing rules' antecedents and is unable to generate new rules.

Of the approaches that are able to generate a comprehensive range of refinements, a number fail to constrain the extent of change to the knowledge-base [19, 21, 24]. Thus, for these systems, although the process uses the initial knowledge-base to construct the new knowledge-base, the similarities between the initial knowledge-base and the refined knowledge-base may be minimal. This is undesirable in the context of MECK, as it is important that the input of the human partner should be retained unchanged unless there is extremely good reason to modify it.

Thus, the only existing inductive knowledge-base refinement approaches that are suited to application in a general MECK context are those of Davis & Lenat [5], Smith, Winston, Mitchell & Buchanan [6], Ourston & Mooney [18] and Webb [22]. Those of Davis & Lenat [5] and Smith, Winston, Mitchell & Buchanan [6] are designed to identify a single refinement at a time whereas those of Ourston & Mooney [18] and Webb [22] are designed to modify an entire rule-base (or set of rules leading to a single conclusion) at a time.

Each of these approaches is likely to be suited to different MECK contexts. Indeed, it may be desirable for a general purpose MECK system to support both modes of operation. The induction of a complete rule-base will, in most circumstances, be desirable if the induction system is developing the first draft of the knowledge-base (if the initial knowledge-base is empty). In other contexts, selection between the induction of a single refinement or a complete set of refinements at a time is most likely to be a matter of personal preference.

4 Human expert capabilities

The power and flexibility of the facilities under the human partner's control is as important as the power and flexibility of the machine learning component. Whenever the human partner feels that he has something to contribute it is important that he have a mechanism by which to make that contribution.

At the very least it is essential that the human partner be able to make arbitrary changes to the knowledge-base. The direct ability to edit the knowledge base is simple to provide. All that is required is an appropriate knowledge base editor. However, as important as the immediate capacity for the user to perform editing is

that the machine learning component should both note and respect during induction the changes previously made by the expert. This is why it is essential that the machine learning system should attempt to minimise the change that it wreaks upon the knowledge base.

Einstein achieves this through the use of the DLGref machine learning algorithm [22], an algorithm that explicitly seeks to minimise the extent to which the initial knowledge base is transformed during induction.

Also important is the ability of the user to provide general advice to the machine learning system, such as placing constraints upon the rules that can be developed, and the ability to identify deficiencies in the knowledge-base without being required to specify a solution. Mechanisms to these ends are discussed below in the section on communication.

It is desirable that the expert be able to specify both precise rules (rules which he believes to be correct) and approximate rules (rules which he believes capture aspects of a correct solution but which may require further refinement). *Einstein* supports these two forms of user supplied rules through the mechanism of rule annotations. Each rule in the knowledge base is annotated with an indication of status. If the user indicates that a rule has the status *accepted*, then the machine learning component will not modify it. This is the appropriate action for precise rules. If the user indicates that a rule has the status *revisable*, the machine learning component is free to modify it. This is the appropriate action for approximate rules.

This mechanism could be further extended by allowing annotation of individual clauses within the condition of a rule. Thus the user could specify that some clauses should not be altered but that others may. However, it is not clear that the extra facility that this would introduce warrants the added complexity that it would entail, in terms of both representing to the user the status of small subparts of the knowledge base, and making explicit the exact consequence of each such specification.

It is also important that the human partner be able to modify and extend the domain model within which the knowledge-base is constructed. That is, it should not only be possible to modify rules, but also possible to create new primitive terms and new predicates and to alter the ranges of existing predicates. For example, if the initial domain model defines temperature as a predicate over the domain *low*, *medium* and *high*, the domain expert should be able to change the range of the predicate to a real number.

While this point may appear trivial, it is frequently the case that a domain expert will realise that a particular domain model is inadequate only after considerable knowledge-base development has occurred. Indeed, it may only be possible to determine that a particular domain model is inadequate by first attempting to create a knowledge base within it.

It is important that changes to the domain model should be supported in such a manner as to enable as much as possible of the knowledge-base developed under the old domain model to be transferred into the new model and which enable the machine learning system to utilise the full power of the new model in all subsequent interactions. It will greatly hinder progress if it is necessary to start afresh every time that a new domain model is required.

Although there has been considerable research into machine learning techniques for extending domain models by learning new terms and predicates [1, 2, 3] the new terms and predicates so learned are necessarily derived from the primitive terms and predicates that are provided. Such approaches extend the space of possible solutions

but still fail to enable a machine learning system to develop adequate solutions when the initial domain model lacks key terms, predicates or operators. Morik [4] has investigated techniques for identifying deficient domain models but still requires knowledge elicitation to rectify the identified deficiencies.

In view of these factors, it seems apparent that

- domain model revision is an important knowledge acquisition capability; and
- this capability can only be satisfactorily provided by a human expert. Machine learning cannot provide it on its own.

The capacity for domain model revision is one of the more important extensions that MECK provides to the power of autonomous machine learning.

Einstein allows the user to modify the domain model by adding or deleting new attributes at any stage. When a new attribute is added, all existing example cases are extended to incorporate the new attribute with the value set to unknown. When an attribute is deleted, it is immediately deleted from all example cases and rules.

Einstein's facility could usefully be extended to enable the range of an attribute to be altered. For example, real valued attributes could be converted to integer values, or the number of values in a categorical attribute could be increased or decreased, with appropriate adjustment of all existing rules and example cases.

5 Communication

Successful collaboration requires the ability for the collaborating parties to communicate. Communication requires a language. It is not feasible with current technology to support domain independent specialist natural language interaction. One alternative is the use of a formal language for communication. However, the need for the human expert to learn a formal language of sufficient power and complexity to support sophisticated discussion about a knowledge base and its strengths and deficiencies would present a tremendous barrier to the use of MECK. The choice of a suitable language or languages for communication is going to be vital to the success or failure of a collaborative system.

Given that the aim of MECK is to produce a knowledge-base and that a knowledge base must be expressed within a formal language, the target language, this language could be used for communication between the collaborating parties. However, while it is important to be able to communicate in the target language, it is also important to be able to communicate *about* expressions in the target language. As many knowledge representation languages do not support meta-statements, this will often require the use of an alternate language for communication. As already stated, the use of a complex formal language for this purpose creates a major barrier to the use of such a system.

6 Annotations to the draft knowledge-base

One simple mechanism for communication about the target language that has been developed for use in *Einstein* is the addition of rule annotations. All communication in *Einstein* is centred around a draft knowledge-base. This knowledge-base consists of a set of rules expressed in the target production rule language. Each of these rules is annotated with a simple evaluation, either *accepted*, indicating that the rule is considered correct and should not be modified, *revisable*, indicating that the rule is a

hypothesis that is not considered immutable, or *rejected*, indicating that the rule is considered unacceptable.

As described above, these annotations are used by the machine learning system during induction. Accepted rules are considered sacrosanct and are never modified by the machine learning system. Revisable rules may be freely modified during induction. Rejected rules are deleted during induction.

In addition to the simple evaluation, further annotation describing the reasons for a particular evaluation can be useful. The current implementation provides only for explanations for the reason why the machine learning system changes the evaluation—the machine learning system has no ability to reason about the human partner's motives for providing an evaluation. Thus, any evaluation provided by the user is described as *user evaluation*.

If the machine learning system rejects a rule due to it misclassifying example cases, the annotation *invalid* is provided. If a rule is rejected because its inclusion in the rule set does not affect the expert system's performance, the annotation *redundant* is provided.

The machine learning component is never able to set a rule's evaluation to *accepted*, so no annotations explaining accepted evaluations are supported.

As it is difficult to provide a meaningful yet succinct explanation as to why the machine learning component should propose any particular new rule, no explanations are provided for revisable evaluations (the evaluation provided by the machine learning component for rules that it induces), either.

This extremely simple mechanism is easy to master and allows simple communication about potential rules for inclusion in the knowledge-base. However, it does not enable communication of complex background knowledge or constraints. To this end, more complex communication mechanisms are required.

7 Case-based communication

The consideration of example cases provides an alternative paradigm for communication to that of formal knowledge-representation languages. Experts are accustomed to and proficient at considering cases and of communicating about expertise through the consideration of cases. Further, it is precisely the analysis of example cases that machine learning systems are structured around and good at performing. In consequence, consideration of example cases provides a powerful mechanism for communication between the human expert and the induction system.

Cases form a natural means of communicating the reasons for the machine learning system's actions to a human user. The primary consideration on which a machine learning system bases its decisions is how well the rules perform on example cases. Thus, one of the key aspects of a correct answer to a question about why a particular rule was developed by a machine learning system is that it correctly handled a particular set of cases. Further, the key to why a particular rule was not developed will usually be that it fails to correctly handle a particular set of cases. While it is true that this is not the complete explanation, (other factors, such as the relative complexity of the two rules may also be involved), it is clear that the use of example cases provides a powerful medium for accurately communicating to the user key aspects of the underlying basis for the induction system's decisions.

However, communication through example cases is not limited to a coarse explication of whether an individual case is handled correctly or incorrectly. Cases can be analysed in two dimensions. In one dimension they are distinguished by whether a rule has fired or not. In the other dimension they are distinguished by whether it would be appropriate for that rule to fire or not. Cases for which a rule fires and for which it is appropriate for that rule to fire (covered positive examples) provide direct evidence of the value of the rule. Similarly, cases for which the rule has not fired and for which it is not appropriate for it to fire (uncovered negative examples) also provide evidence of the rule's value. Cases for which the rule has fired and for which it is not appropriate for it to fire (covered negative examples) provide evidence against the rule. Similarly, cases for which the rule has failed to fire, but for which it would be appropriate to fire also provide evidence of potential shortcomings of the rule.

Covered examples provide evidence relating to potential specialisations of a rule. Covered positive examples can be used to explore the limits to potential specialisations to a rule. It will usually be undesirable to specialise a rule so that it ceases to cover a covered positive example.

By contrast, covered negative examples provide guidance for the potential extent for specialisations. It will generally be desirable to specialise a rule so that it no longer covers a covered negative example.

Similarly, uncovered examples provide evidence relating to potential generalisations. Uncovered positive examples can be used to find ways in which it might be useful to generalise a rule whereas uncovered negative examples place constraints on the desirable generalisations.

Each of these classes of examples can be used by the machine learning component to communicate different information to the human expert. If asked why was a rule developed it can reply by showing the rule's covered positive examples and uncovered negative examples. If asked why an alternative rule was not developed it can reply by showing the covered negative examples and uncovered positive examples for that rule. In general, the four sets of examples provide a general summary of the machine learning system's evaluation of any rule or potential rule.

Rather than providing a mechanism for the human expert to pose questions for which the different sets of examples can be used as answers, *Einstein* at all times provides an explicit list of each of the covered positive, the covered negative and the uncovered positive examples. These lists are of such high value and are so frequently consulted that it would become a burden to require the user to explicitly request them each time that they are to be consulted.

Uncovered negative examples are not provided as it is natural for the human user to expect negative examples to not be covered and thus it is only important to draws attention to violations of this expectation in the form of covered negative examples.

The availability of each of these lists of examples also serves another important role—training set verification. It is not unknown for errors to be included in the set of example cases that are made available to the machine learning system. These will often cause the induction of anomalous rules. Investigation of these rules, through perusal of the appropriate lists of examples, is likely to lead to the identification of the erroneous examples which can then be corrected.

Another use for example cases is to provide a summary of the general accuracy of a rule set when applied to a set of cases. Such summaries are routinely used to

evaluate the performance of machine learning systems. This mechanism serves to provide a useful overview of progress for the expert during system development.

Example cases can also be used to demonstrate the interactions of rules by stepping through the application of a rule set to an example case. Such a facility greatly enhances the ability of the human expert to gain a detailed understanding of the knowledge-base. It also serves to demonstrate how the rules within a rule set interact.

Not only do cases serve as a vehicle for the machine learning system to communicate with the human expert, they also provide a natural and powerful means for the human to communicate with the induction system.

Where an expert is unable to specify exactly how a rule should be changed, but is aware of deficiencies, the provision of examples provides a simple mechanism for expressing those deficiencies. Counter-examples can be used to demonstrate errors in a rule while positive examples provide a means for expressing how a rule should be extended. For example, if a rule stating that *100000 ≤ Urinary Red Blood Cell Count ≤ 500000 is a sufficient condition for diagnosing acute tubular necrosis* is incorrect, rather than requiring the human user to learn a formal language for communicating this information to the system it is far easier for the user to simply provide examples for which the condition *100000≤Urinary Red Blood Cell Count≤500000* applies but for which acute tubular necrosis should not be diagnosed. Experts are typically accustomed to the use of examples as an aid to the communication of concepts and, in informal studies, have little difficulty in adopting this form of communication.

Any such examples developed by the expert need only be added to the training set for the machine learning system to have complete access to their full import.

As the description of detailed examples can be tedious, it is advantageous if the expert is able to provide partial examples, with irrelevant details left unspecified. This implies, of course, that the machine learning system must be able to handle such partial examples.

It is possible to further extend the power of case-based communication by allowing the use of invalid examples. These are examples of cases that cannot occur. For instance, if an expert believed that a particular combination of symptoms could not be associated with disease *X*, but did not wish to provide an example with a particular outcome that was incompatible with *X* (assuming that *not X* was not a possible conclusion in the knowledge base under construction), it would be possible to provide an invalid example with the relevant symptom and the diagnosis *X*, thereby communicating the desired information to the system.

When this mechanism is coupled with the use of partial examples, it can provide a powerful mechanism for expressing constraints. For example, to communicate that males cannot be pregnant, it is only necessary to create an invalid example which is male and pregnant and for which all other values are unspecified.

In effect, normal examples specify portions of the space of possible knowledge-bases that are desirable whereas invalid examples specify sections of the space of possible knowledge-bases that are undesirable. Covered, uncovered, positive and negative examples provide expressions of how a particular knowledge-base relates to the specification provided by the set of examples.

While case-based communication is powerful, flexible and easy to use, it is difficult to use it for expressing complex constraints and background knowledge. For example, there is no simple manner in which example cases can be used to

express that $E=mc^2$. If it is not possible to express this in the target language, and it is relevant, it would be necessary to use a formal meta-language in order to communicate it.

The use of example cases for communication is not new. This means of communication is widely used in human discourse. In the knowledge acquisition context, it has been used to some extent by all previous MECK systems. However, previous researchers have not identified the role that it plays and there has been no previous attempt to map out its scope and limitations.

8 Other communication mechanisms

So far, we have examined mechanisms that allow the partners to communicate about the knowledge-base under development. However, collaboration requires communication not only about the objective but also about the means by which that objective is to be reached. Thus, there is a need to communicate about planning or, at very least, control of the collaborative work. Again, a formal language capable of supporting complex task planning and control would be extremely complicated and its use would create a major barrier to the application of MECK.

Einstein, places initiative firmly in the hands of the human partner. All long term planning is left entirely to the human partner. All control communication takes the form of commands issued by the human partner to the machine learning system. For ease of use, these commands are issued via a standard menu and dialogue interface.

However, there is clearly great potential value in allowing the machine learning component to assume initiative in opportune circumstances. For example, if the system were to observe the human partner make a series of changes to the knowledge base that led to a decrease in performance, it might be opportune for it to notify the human partner of this and to suggest steps that might rectify the situation.

In the belief that it is important for the human partner to feel in control of the joint project, such computer generated seizures of initiative should not be obtrusive and should take the form of suggestions rather than commands. Thus, the computer should wait until the human partner is not engaged in an important operation before taking the initiative and all such initiatives should be subject to approval by the human partner before any irrevocable action occurs. Further, such actions should be kept to a minimum and should only occur when the potential gains are substantial.

As the forms of computer generated initiative are likely to be restricted, a simple dialogue mechanism should suffice for communication in this context.

9 Communication with *Einstein*

Einstein implements most of the facilities described above into a Macintosh based knowledge acquisition environment. During operation , *Einstein* continually displays the relevant windows. Thus, the user has available at most times windows displaying the following:

- the current knowledge base;
- all examples;
- all covered positive examples for the current rule;
- all counter (covered negative) examples for the current rule;

- all uncovered positive examples for the current rule;
- all examples for which it is not possible to determine (due to missing values) whether or not they are covered by the current rule.

At any time other than when the system is employing machine learning, he may

- edit any visible rule or example in situ;
- add new examples;
- revise the domain model;
- apply the knowledge base in an interactive environment;
- provide direction to the machine learning component.

All of these mechanisms are readily assimilated, even by users with relatively little computer experience and no knowledge acquisition experience. Although these communication mechanisms are simple, as outlined above, they support extremely powerful dialogue.

Most importantly, the mechanisms are sufficiently familiar for untrained users to employ them with little or no tuition. New users with no previous exposure to knowledge acquisition rapidly enter into dialogue with *Einstein* without even having apparent conscious awareness of the deep messages being carried by the simple surface interactions. For instance, it is so natural for a human expert to provide a counter-example that he does not need to explicitly consider the deep message that it conveys (that the rule under critique is deficient in that it cannot accommodate this new case *and* that *Einstein* should do something about this deficiency).

10 Conclusions

Two decades of intensive research into machine learning has seen impressive results and the development of numerous useful automated knowledge acquisition tools. However, an autonomous machine learning system will always be limited by the comprehensiveness of its training set. Unless every relevant combination of factors is represented in the training set, no matter how infrequently it occurs in practice, a machine learning system cannot be expected to produce a correct knowledge base. As it will frequently be the case that a training set will not be sufficiently comprehensive, autonomous machine learning is necessarily limited in the extent of its applicability.

Nevertheless, even though a machine learning system will not be able to produce a perfect knowledge base from an incomplete training set, it may still be able to derive valuable insights therefrom. Frequently, these insights will be quite different from those otherwise available to the human expert.

In consequence, there is every reason to believe that machine learning is able to provide an adjunct source of insight during the knowledge-acquisition process. Machine-expert collaboration for knowledge-acquisition provides one approach to harnessing that insight.

Techniques for MECK are still in their infancy. This paper has identified three key issues facing MECK—control, capabilities and communication. It is argued that control should be placed in the hands of the human expert, who is likely to be situated in the context in which the knowledge-base is to be employed and who will have considerable influence, once knowledge-acquisition is completed, upon the success or failure of its application. However, although the expert should have control, both parties should be able to provide initiative.

The range of capabilities that are required is likely to depend greatly on the context. However, it is essential that the machine learning system be able to refine successive drafts of the knowledge-base and that the human expert be able to perform arbitrary changes to the knowledge-base and provide advice and guidance to the machine learning system.

The communication facilities should be easy to master while allowing the expression of complex knowledge, meta-knowledge and cooperative control information. It is appropriate to use a number of distinct languages for communication. Knowledge can be expressed in the target language. Simple meta-knowledge can be expressed by annotations to expressions in the target language. Complex knowledge and meta-knowledge can be expressed through the use of example cases. Finally, a simple command language can be used to manage the process of collaboration.

These facilities are well within the reach of current technology. Indeed, most are implemented in the *Einstein* system.

While much remains to be done, the integration of machine learning with knowledge elicitation is finally at hand.

Acknowledgments

This research has been supported by the Australian Research Council and the Apple University Development Fund.

References

1. E. Bloedorn, R. S. Michalski: Data-driven constructive induction in AQ17-PRE: A method and experiments. In *Proceedings of the 1991 IEEE International Conference on Tools for Artificial Intelligence*, San Jose, CA, 1991, pp. 30-37.
2. Z.-J. Zheng: Constructing conjunctive tests for decision trees. *AI '92*. Singapore: World Scientific, 1992, pp. 355-360.
3. S. Yip, G. Webb: Discriminate attribute finding in classification learning. *AI '92*. Singapore: World Scientific, 1992, pp. 374–379.
4. K. Morik: Sloppy modeling. In K. Morik (Ed.) *Knowledge Representation and Organization in Machine Learning*. New York: Springer-Verlag, 1989, pp. 107-134.
5. R. Davis, D. B. Lenat: *Knowledge-Based Systems in Artificial Intelligence*. New York: McGraw-Hill, 1982.
6. R. G. Smith, H. A. Winston, T. M. Mitchell, B. G. Buchanan: Representation and use of explicit justifications for knowledge base refinement. In *Proceedings of the Ninth International Joint Conference on Artificial Intelligence*. San Mateo, Ca: Morgan Kaufmann, 1985, pp. 673-680.
7. Attar Software. (1989) *Structured Decision Tasks Methodology for Developing and Integrating Knowledge Base Systems*, Leigh, Lancashire: Attar Software, 1989.
8. J. L. O'Neill, R. A. Pearson: A development environment for inductive learning systems. In *Proceedings of the 1987 Australian Joint Artificial Intelligence Conference*, Sydney, 1987, pp. 134-145.

9. A. Shapiro: *The Role of Structured Induction in Expert Systems*. PhD Thesis, University of Edinburgh Machine Intelligence Unit, 1983.
10. C. Sammut, R. B. Banerji,: Learning concepts by asking questions. In R. S. Michalski, J. G. Carbonell & T. M. Mitchell (eds) *Machine Learning: An Artificial Intelligence Approach, vol II*. Los Altos: Morgan Kaufmann, 1986, pp. 167-191.
11. G. Webb: Einstein: An interactive inductive knowledge-acquisition tool. In *Proceedings of the Sixth Banff Knowledge Acquisition for Knowledge-Based Systems Workshop*, Banff, 1991, pp. 22-1–22-16.
12. G. Webb: Man-machine collaboration for knowledge acquisition. *AI '92*. Singapore: World Scientific, 1992, pp. 329–334.
13. J. R. Quinlan: Generating production rules from decision trees. *Proceedings of the Tenth International Joint Conference on Artificial Intelligence*. Los Altos: Morgan Kaufmann, 1987, pp. 304-307.
14. R. A. Caruana: The automatic training of rule bases that use numerical uncertainty representations. In L. N. Kanal, T. S. Levitt & J. F. Lemmer (Eds.) *Uncertainty in Artificial Intelligence 3*. Amsterdam: Elsevier Science, 1989, pp. 347-356.
15. A. Ginsberg: *Automatic Refinement of Expert System Knowledge Bases*. London: Pitman., 1988
16. A. Ginsberg, S. M. Weiss, P. Politakis: Automatic knowledge base refinement for classification systems. Artificial Intelligence 35, 197-226 (1988).
17. Y. Ma, D. C. Wilkins: Improving the performance of inconsistent knowledge bases via combined optimization method. *Proceedings of the Eighth International Machine Learning Workshop*, 1991, pp. 23-27.
18. D. Ourston, R. J. Mooney: Changing the rules: A comprehensive approach to theory refinement. *AAAI-90*, 1990, pp. 815-820.
19. M. J. Pazzani, C. A. Brunk: Detecting and correcting errors in rule-based expert systems: An integration of empirical and explanation-based learning. Knowledge Acquisition 3, 157-173 (1991).
20. R. Rada: Gradualness facilitates knowledge refinement. IEEE Transactions on Pattern Analysis and Machine Intelligence, PAMI-7, 523-531 (1985).
21. R. E. Reinke, R. S. Michalski: Incremental learning of concept descriptions: A method and experimental results. In J. E. Hayes, D. Michie, J. Richards (eds) *Machine Intelligence 11*, Oxford: Clarendon Press, 1988, pp. 263-288.
22. G. Webb. DLGref2: techniques for inductive knowledge refinement. In Proceedings of the IJCAI Workshop W16, Chambery, France, 1993.
23. D. C. Wilkins, B. G. Buchanan: On debugging rule sets when reasoning under uncertainty. In *AAAI-86: Proceedings of the Fifth National Conference on Artificial Intelligence*, Philadelphia, 1986, pp. 448-454.
24. W. D. Lee, S. R. Ray: Rule refinement using the probabilistic rule generator, *Proceedings of the Fifth National Conference on Artificial Intelligence*. San Mateo, CA: Morgan Kaufmann, 1986, pp. 442-447.