KNOWLEDGE ACQUISITION AND HEURISTIC CLASSIFICATION IN WEAK-THEORY DOMAINS

BRUCE W. PORTER
E. RAY BAREISS
ROBERT C. HOLTE

FEBRUARY 1989 AI89-96

* Support for this research was provided by the Army Research Office under grant ARO-DAAG29-84-K-0060 and the National Science Foundation under grant IRI-8620052.
Knowledge Acquisition and Heuristic Classification in Weak-Theory Domains

Bruce W. Porter
Department of Computer Sciences,
University of Texas, Austin, Texas 78712

E. Ray Bareiss
Computer Science Department,
Vanderbilt University, Nashville, Tennessee 37235

Robert C. Holte
Department of Computer Sciences,
University of Texas, Austin, Texas 78712

1Support for this research was provided by the Army Research Office under grant ARO-DAAG29-84-K-0060 and the National Science Foundation under grant IRI-8620052.
Abstract

This paper describes a successful approach to knowledge acquisition for heuristic classification. Almost all current programs for this task create or use explicit, abstract generalizations. These programs are largely ineffective for domains with weak or intractable theories. An exemplar-based approach is suitable for domains with inadequate theories, but raises two additional problems: determining similarity and indexing exemplars. Our approach extends the exemplar-based approach with solutions to these problems. An implementation of our approach, called Protos, has been applied to the domain of clinical audiology. After reasonable training, Protos achieved a competence level equaling that of human experts and far surpassing that of similar programs. Additionally, an “ablation study” has identified the aspects of Protos that are primarily responsible for its success.
1 Introduction

This paper describes a successful approach to the task of knowledge acquisition for heuristic classification. Although similar to concept learning, which is widely studied in machine learning, this task has three additional requirements. First, classifications must be explained, not simply reported. Second, a program for this task must accommodate incomplete case descriptions. Third, the program must learn domain-specific knowledge for inferring case features needed for classification. Section 2 summarizes this learning and classification task.

The traditional approach to concept learning and classification relies on generalizations. It requires a strong domain theory both to summarize training cases with concept descriptions and to classify new cases using these descriptions. Section 3 argues that this approach is ineffective for most domains.

An alternative approach relies on exemplars. Concepts are learned by retaining category exemplars and new cases are classified by matching them with exemplars. Our learning and classification program, Protos, uses the exemplar-based approach. Section 4 describes Protos’s design and its appropriateness for weak-theory domains.

To evaluate the design, Protos was applied to the task in clinical audiology of identifying a patient’s hearing disorder from symptoms, test results, and history. An expert clinician instructed Protos with 200 cases – a level of training comparable to that received by student clinicians. After this training, Protos’s classification accuracy was compared with that of clinicians and several learning programs. Protos compared favorably with the best clinician and was significantly better than the other programs. Finally, an “ablation study” [10] identified the aspects of Protos that are primarily responsible for its success. Section 5 describes the evaluation.

Section 6 summarizes the research. We conclude that exemplar-based learning and classification is appropriate and effective for domains lacking a strong domain theory. Our current research focuses on explanation, Protos’s least developed capability.

2 The Task: Knowledge Acquisition for Classification and Explanation

Our research addresses the task of acquiring knowledge to improve competence at classification and explanation, starting at a level of utter incompetence and aiming for a level of expert competence. This section specifies this task in detail by defining classification, explanation, and knowledge acquisition, and by giving measures of competence for each.
2.1 Classification

Classification\(^2\) is assigning a given input, called a case, to one of the categories in a pre-enumerated list. Competence at classification is defined in terms of accuracy and efficiency. A case is described by a collection of features. However, case descriptions differ in two significant ways from the feature-vector descriptions common in machine learning. First, a case description may be incomplete, in the sense that it does not include some of the features present in other case descriptions. Second, the features with which cases are described may not directly indicate category membership. Instead, inference using domain-specific knowledge may be necessary to determine category membership from a case description. For example, suppose a case is a configuration of pieces on a chessboard, described in terms of pieces and their positions, and the categories are \textit{win-for-white-in-9-ply} and \textit{no-win-for-white-in-9-ply}. There is no known way to determine membership in these categories directly from a case description, but it can be determined using knowledge-based inference, such as an exhaustive 9-ply look-ahead based on the knowledge of the rules of chess.

The “heuristic classification method” described by Clancey [7] is tailored to domains in which cases are described with features that do not directly indicate category membership. It explicitly includes “the important twist of relating concepts in different classification hierarchies by non-hierarchical, uncertain inferences”[7, p. 290]. However, it has no associated learning method. Furthermore, current methods for the joint task of learning and classification are not well-suited to domains requiring this “twist” (see Section 3).

2.2 Explanation

In this paper, classification is combined with explanation: an input case must be classified and the classification must be explained. Explanation, in the broadest sense, includes a variety of inference methods for reasoning, learning, and communicating [52]. It is the focus of our current research (see Section 6), but knowledge acquisition, not explanation, is the central subject of the research reported in this paper. In order to concentrate on knowledge acquisition issues, we adopted a simplified notion of explanation. The main simplifying assumptions, similar to those made in first-generation expert systems, are as follows.

First, explanations are used for only two purposes: to justify classifying a case in a particular way, and to establish the degree of similarity of two cases.

Second, explanations are all of the same form: domain-specific terms (e.g., features, feature-values, categories) are related to one another in domain-independent ways (e.g., “X causes Y which enables Z” where X, Y, and Z are domain-specific).

Third, explaining the classification of a case only requires mentioning the evidence supporting the classification. It is not necessary to mention evidence against the classification or evidence pertaining to other possible classifications. For example, to explain how a particular conclusion was reached, Mycin [47] lists only the satisfied rules leading to the conclusion;

\(^2\)Terms appear in boldface when they are defined.
it does not list the rules leading, with less confidence, to different conclusions.

Finally, an explanation can be constructed by a simple transformation of the inferential path that leads to a classification. The transformation may involve suppressing certain details and rephrasing or reorganizing the remaining details, but it does not involve natural language generation, adaptation to a model of the person to whom explanations are targeted, etc.

In our study, competence at explanation is defined in terms of the quality of the explanation, as assessed by a human expert.

2.3 Knowledge Acquisition

Knowledge acquisition is the elicitation of knowledge from a human expert and the incorporation of this knowledge into an existing body of knowledge. In the present task, the purpose of knowledge acquisition is to improve competence at classification and explanation by modifying the underlying knowledge base.

Knowledge acquisition is mixed initiative. The program may request particular knowledge from the expert, or the expert may take the initiative and volunteer knowledge. Typically, knowledge is provided in the form of training cases which are classified and explained by the expert. The knowledge provided by the expert and the explanations produced by the program are expressed in the same language. The expert may introduce new vocabulary (categories, features) at any time.

Competence at knowledge acquisition is defined in terms of two factors. The first is the competence at classification and explanation attained after a realistic amount of training. The second factor is the degree to which the knowledge acquisition program gains autonomy as the knowledge base develops. Autonomy may be measured by the number and nature of program-issued requests for knowledge. For example, a program exhibits low autonomy if it asks questions of the expert that it could answer by consulting the existing knowledge base. A full discussion of autonomy gain is given in [4].

3 Weaknesses of Generalization-based Methods for Learning and Classification

Almost all current learning and classification programs are generalization-based, in the sense that they create or use explicit abstract generalizations. Generalization-based programs are either simple or theory-based, depending on whether the language with which cases are described (called the case language) and the language with which generalizations are expressed (called the generalization language) are closely related or entirely different. Examples of simple programs are ID3 [39] and CN2 [9]. Examples of theory-based programs are explanation-based programs (e.g., [12, 31]) and similarity-based programs that use background knowledge (e.g., [32, 30, 21, 16]). Both types of generalization-based programs have
been applied successfully.

A case language usually consists of intrinsic, readily perceivable features. Such features are called superficial. An intrinsic feature of a case is one that is defined without reference to the case’s context, e.g., the role of the case in a particular task, or the situation in which the case occurs. For example, with intrinsic features, a desk would be described as an arrangement of structural parts (e.g., drawers, legs, top) with physical properties (weight, size, strength). With non-intrinsic features it might be described as an arrangement of functional parts (e.g., work surface, storage areas) with use-related properties (e.g., ample, ergonomic, easy to clean). Features that are not superficial are called abstract.

Simple programs for learning and classification are applicable when superficial features suffice to define the generalizations of interest. For example, in a simulated blocks world, the superficial features shape, size, color, and relative position suffice to define generalizations such as arch, stack, and large, red block.

In most domains, superficial features do not suffice to define generalizations. Generalizations such as cup [54] and hammer [11] are defined in terms of function, not form. Categories such as infected by pseudomonas [47] and seedless grape [2] are generalizations defined in terms that are not readily perceivable in the context of non-destructive diagnosis.

When abstract features are required to define generalizations, a gap exists between the case language and the generalization language. This gap may be bridged in two ways. The first is to preprocess the case descriptions to add the required abstract features. It is usually necessary for human experts to do the preprocessing, because evaluating the abstract terms (e.g., “abnormal”) requires expert judgement. In this way, simple programs can be used in domains in which the relationship between abstract and superficial features is not well understood. The second way to bridge the language gap is to construct a domain theory describing the relationships between terms in the two languages and use theory-based programs. Note that theory-based programs provide no assistance in the difficult task of constructing a domain theory.

Theory-based programs are applicable only if the domain theory is both tractable and strong. A tractable domain theory is one in which the definitions of all terms can be computed efficiently from superficial features. The strength of a domain theory depends on the certainty associated with the relationships between terms. The strongest theories, called perfect, consist entirely of relationships of perfect certainty, such as standard logical and taxonomic relationships. The weakest theories consist of correlational relationships, such as “X and Y often co-occur.”

Very few domains have the tractable, perfect domain theories required by current theory-based programs. Indeed, few domains have perfect domain theories, tractable or otherwise. Legal reasoning, for example, almost always involves open-textured concepts, i.e., concepts having only a weak domain theory [17, 18]. The fact that many fields of diagnostic expertise lack a perfect domain theory is indicated by the widespread use of certainty factors in expert systems. When a perfect domain theory does exist, it is often intractable. For example, the rules of chess constitute a perfect, but intractable, theory of “winning position.” Even for
chess endgames involving very few pieces and analyzed extensively in textbooks, the rules constitute an intractable theory, and the existing tractable theories are far from perfect [46].

The developers of theory-based programs have acknowledged this severe limitation [31], and have recently begun trying to adapt their programs to work with weak theories. We anticipate that generalization-based programs will not adapt well to weak theories.

With a weak theory, the most reliable and efficiently found chains of inference are those that are short and involve individual steps of low uncertainty. Chains of inference bridging the language gap, which are necessary whenever a generalization is created or used, are usually long and involve steps of high uncertainty. Consequently, generalization-based programs, when used with a weak theory, can be expected to be unreliable and inefficient.

By contrast, chains of inference consisting entirely of direct matches between features are short and reliable. For example, a case can be classified with certainty if it is identical to a training case. The exemplar approach to learning and classification, described in the next section, attempts to achieve reliability and efficiency by maximizing the use of direct match. Training cases are recorded, and a case is classified by comparing it, feature by feature, with the training cases. Domain theory is used only for those features that have no direct match. The number of such features, and therefore the use of domain theory, can be minimized by retaining all training cases. However, this is only necessary when the domain theory is very weak. With a strong domain theory, very few training cases need to be retained to achieve reliable, efficient inference.

In summary, generalization-based methods are not likely to perform as well as exemplar-based methods in domains with weak theories. Furthermore, domains with weak theories are far more common, in practical applications, than domains with strong, tractable theories. Other weaknesses of generalization-based methods are given in [49, 45, 2, 8, 55].

4 Protos: The Exemplar-based Alternative

This section describes the design of Protos, an exemplar-based program for concept learning and classification. Simple exemplar-based programs, although strongly supported by psychological studies, suffer from two fundamental problems: determining similarity and indexing exemplars. Protos’s design includes solutions to these two problems. The design principles are introduced with simple, familiar examples and demonstrated with a large-scale application of Protos to clinical audiology. Complete details of Protos are given in [3], and a Common Lisp reconstruction, available for distribution, is documented in [14].

4.1 Simple Exemplar-based Concept Learning and Classification

\(^3\)except, of course, for the theories painstakingly developed by Shapiro[46], Quinlan [38], and others in order to apply generalization-based programs to these domains.
Given:

- a set of exemplar-based categories \( C = \{c_1, c_2, \ldots, c_n\} \)
- and a case (\textit{NewCase}) to classify.

**REPEAT**

**Classify:**
- **Find** an exemplar of \( c_j \in C \) that **Strongly Matches** \textit{NewCase}
- and classify \textit{NewCase} as \( c_j \).
- **Explain** the classification.

**Learn:**

- If the expert disagrees with the classification or explanation then acquire classification and explanation knowledge and **adjust** \( C \)
  - so that \textit{NewCase} is correctly classified and explained.

**UNTIL** the expert approves the classification and explanation.

**Figure 1:** Exemplar-based Learning and Classification Algorithm. The hard problems of the exemplar-based approach are boxed.

![Exemplar-based Learning and Classification Algorithm](image)

**Figure 2:** A Portion of the Exemplar-Based Category \textit{chair}
Figure 1 describes the exemplar-based approach to concept learning and classification and identifies the hard problems. Concepts are represented extensionally with a collection of exemplars described with features in the case language. For example, the concept chair is represented in Figure 2 by two exemplars, chair1, a metal chair with a pedestal and wheels, and chair2, a wooden chair with four legs. Classifying a new case involves searching for an exemplar that strongly matches the case. The simplest method is an exhaustive search for a direct match. Explaining the classification involves showing the line of reasoning used during match. The simplest explanation is a list of the common features of the case and the exemplar. Learning from a case involves adjusting the categories so that the case will be properly classified and explained. The simplest adjustment adds the case to the correct category as a new exemplar and ensures that it will be found before other matching exemplars, should this case be classified again.

In some theories of exemplar-based categories, including Protos’s theory, abstract features may be defined by exemplars, just as categories are. Determining whether such a feature is present or absent in a given case involves matching the case with the exemplars of the feature. Concept is the general term used to refer to categories and features.

Psychological experiments, devised to distinguish between the generalization-based approach and the exemplar-based approach, support the exemplar-based approach. As in machine learning, early psychological research assumed that generalization was automatic; researchers focused on what is abstracted and how generalization is performed, rather than whether cases are generalized [27]. Recent research indicates that people resist generalization and retain cases. For example, Medin[27] and Brooks[6] found that people classify previously seen cases by direct matching. Tversky and Kahneman[50] found that people estimate the frequency of a class or the probability of an event by their ability to recall instances of the class or event. Holyoak and Glass [19] found that people reject false statements by recalling category exemplars for which the statement is untrue. For a range of cognitive tasks, these studies emphasize retaining, recalling, and matching category exemplars, rather than reasoning with category-wide abstractions. To account for this data, psychological theories propose models involving exemplar-based concept learning and classification [41, 28, 29, 49, 44].

The simple exemplar-based method uses no domain theory. However, domain theory is indispensable for solving the hard problems indicated in Figure 1. For example, determining the strength of the match between an exemplar and a case requires knowing the basis for category membership. Murphy and Medin [33] argue that domain theory provides this basis and adds coherence to a collection of otherwise dissimilar exemplars. The following sections describe how Protos uses domain theory to determine the similarity of a case and an exemplar and to index exemplars.
4.2 Problem 1: Determining the Similarity of a Case and an Exemplar

The simple exemplar-based method uses only direct match, and treats identically all unmatched features. This gives a very crude estimate of the quality of an imperfect match between an exemplar and a new case. The problem of estimating the quality of an imperfect match is called the matching problem. To solve this problem Protos acquires and uses matching knowledge, a form of domain theory.

As illustrated in Figure 3, there are two types of matching knowledge. The first is relations among concepts, for example "seat enables holds(person)." The second type of matching knowledge is featural importances. As defined by Medin and Schaffer [28], this is knowledge of the "differential salience" of an exemplar's features to its category. For example, the wheels feature of chair1 is spurious to the category chair, and the seat feature is essential. Section 4.2.1 describes Protos's acquisition of matching knowledge.

Given matching knowledge, Protos can match dissimilar features by finding a path of relations connecting them. For example, in Figure 3, Protos can match pedestal with legs(4) because of the specialization-of relations connecting each of them to seat support. Unlike simple, feature-counting similarity functions [20], explanations of similarity can be heuristically evaluated and discussed with the expert. Section 4.2.2 describes Protos's use of matching knowledge to determine similarity.

Protos is equally applicable to all domains, regardless of the strength of the domain theory. A perfect domain theory includes matching knowledge sufficient to match all the members of a category with each other. When this is available, the category can be represented with a single exemplar, called a prototype [26, 49]. When a perfect theory is not available, the category can still be represented, fairly accurately, by using several exemplars. For example, the category strike in baseball can be represented with two exemplars, one in which the batter swings and fails to hit the ball into fair play, and another in which the ball crosses the plate through the strike zone. With a very weak domain theory, an accurate representation of a category may require many exemplars.

The polymorphy of a concept is the amount of unexplained variability among the instances of a concept [42]. In domains with a strong theory, the polymorphy of most concepts will be low. In domains with a weak theory, the polymorphy of concepts can vary considerably (e.g., see Figure 14 in Section 5.1). Because of this, Protos has been designed to cope with concepts of any degree of polymorphy. Protos retains a case as an exemplar only if the case differs from existing exemplars in significant ways that cannot be explained using existing matching knowledge. If a case does match an exemplar strongly, it is "merged" with the exemplar and not retained. Thus, the number of exemplars that Protos retains for a concept is a direct indication of the concept's polymorphy.
Figure 3: Protos’s Matching Knowledge for chair. The upper diagram indicates relations among concepts. The lower diagram indicates featural importances (dotted line means spurious, medium line means moderately important, and thick line means essential.)
4.2.1 Acquiring Matching Knowledge from Explained Cases

Protos acquires matching knowledge from explained cases. When Protos fails on a new case, the expert provides the classification and explanation. Protos installs this information in its network of matching knowledge. For example, part of the knowledge of chairs (Figure 3) was learned when the expert classified chair1 and explained:

\[
\text{pedestal is a specialization of seat support which enables holds(person) which is the function of chair.}
\]

Protos requires feature-to-category explanations relating each case feature to the case's classification. To explain the relationship, the expert typically introduces new concepts and relations. For example, the previous explanation introduces the category seat support, the function holds(person), and three relations. Protos adds these concepts and relations to its current network of matching knowledge.

Protos and the expert work together to explain the relationship between case features and categories. Often the expert provides feature-to-feature explanations and Protos completes the explanation of the classification. For example, after the expert explains the relationship between pedestal and chair, Protos explains the relevance of legs(4) for chair2 given only that "legs(4) is a specialization of seat support."

Explanations are expressed in a predefined language of relations. Some relations are "strong," as in "adolescent definitionally entails minor," while others are "weak," as in "sharp teeth suggest carnivorous." Figure 4.2.1 shows all the relations in decreasing order of strength. Each relation can be strengthened or weakened with qualifiers, such as always, usually, sometimes, or occasionally.

Conditional explanations are restricted to particular categories or exemplars. They have the form: "IF condition THEN explanation." For example:

\[
\text{IF the category is apples THEN color(green) is sometimes equivalent to color(red)}
\]

The condition can restrict the explanation's applicability to members of a particular category, cases with particular features, or matches with particular exemplars.

Protos heuristically estimates the importance of a feature to a category by analyzing a feature-to-category explanation. For example, from the explanations relating chair1's features to the category chair, Protos estimates that seat is an essential feature of a chair, that pedestal is a moderately important feature of a chair, and that wheels is a spurious feature of chair1 (see Figure 3). Internally, featural importances are represented as numbers between 0 (spurious) and 1 (essential). Protos's estimates of featural importances may be revised by the expert if they result in a misclassification or unacceptable explanation.

The lower part of Figure 5 summarizes Protos's algorithm for learning matching knowledge. As this figure indicates, the matching knowledge that is acquired depends on Protos's assessment of the similarity of a case and an exemplar, and whether the expert agrees. Protos's algorithm for assessing similarity is discussed next.
causes
has function
is mutually exclusive with
is equivalent to
requires
if and only if
definitionally entails
has generalization
enables
c o-occurs with
is consistent with
implies
suggests
part of
spurious with

Figure 4: Relations in Protos's Explanation Language (in order of decreasing strength). Each relation has an inverse which is not shown (e.g., the inverse of has generalization is has specialization.)
4.2.2 Using Matching Knowledge to Determine Similarity

Protos classifies a new case by explaining its similarity to an exemplar. This method, summarized in the upper part of Figure 5, is called knowledge-based matching. It uses matching knowledge and a collection of heuristics to evaluate explanations.

During knowledge-based matching of an exemplar and a case, the exemplar is a model for interpreting the case. It determines which features are important for a successful match. If an important feature is absent from the case description, Protos attempts to infer it from the case features using matching knowledge. Unlike the models used by other expectation-driven classifiers [51, 1, 36], exemplars are specific and numerous; usually, case features and exemplar features match directly, and the range of category exemplars provides models for both typical and atypical cases.

Knowledge-based matching is a uniform-cost, heuristic search. The search begins from each unmatched exemplar feature, and chains through the network of matching knowledge until reaching either a case feature or the depth bound (step 5.2 in Figure 5). Each step of the search extends the current path with a relation. A path connecting an exemplar feature with a case feature is an explanation of how the features are “equivalent,” in that the features suggest the same classification.

Knowledge-based matching uses 36 domain-independent heuristics to evaluate the quality of a path ([3, appendix C]). The purpose of the heuristics is to find the strongest explanation and to prune weak explanations. When selecting from a set of relations to extend a path, the heuristics evaluate the potential contribution of each relation to the developing explanation. This is a function both of:

- The individual relation and its qualifiers. For example, the heuristics penalize the inclusion of weak relations such as “sometimes implies” in an explanation.

- The overall explanation constructed thus far. For example, one heuristic prevents ascribing the function of an assembly to a particular part. This heuristic would prune an explanation that begins “steering wheel is part of car which has function transportation . . . ”

The depth bound for the search relating an exemplar feature and a case feature is a function of the importance of the exemplar feature to the exemplar’s category. If the feature is necessary, Protos will search extensively, allowing weak explanations to be found. If the feature is moderately important, the search either will find a strong explanation or fail. If the feature is spurious, Protos will not search at all.

In calculating the overall match strength between a case and an exemplar (step 5.3) each feature of the exemplar contributes a factor between 0 and 1. The contribution of an unmatched feature with importance \( i \) is \( 1 - i \) (for example, 0 for an essential feature). The contribution of a matched feature is the strength of the explanation relating the feature to a case feature (1 for a direct match; a fraction for an explained match). The overall match strength is the product of these factors for all the exemplar’s features (cf., the similarity
GIVEN: a case (NewCase) to classify, and an exemplar (Exemplar).

To determine the similarity of NewCase and Exemplar:

5.1 Assign a high match strength to each feature of Exemplar that directly matches a feature of NewCase.
5.2 For each feature of Exemplar that is not directly matched, search matching knowledge for the best explanation relating it to a feature of NewCase. Assign a match strength corresponding to the quality of the explanation.
5.3 Compute the overall similarity of Exemplar and NewCase using the match strength of each matched exemplar feature and the importances of each unmatched exemplar features.

If the match between NewCase and Exemplar is sufficiently strong, Exemplar’s category is used to classify NewCase and the match between Exemplar and NewCase is used to explain this classification. Indexing knowledge is acquired by discussing this classification and explanation with the expert, as follows.

IF the expert rejects the classification or explanation
5.4 THEN Request new matching knowledge from the expert.
5.5 ELSE (the expert accepts the classification and explanation)
5.6 IF some features of Exemplar are unmatched
5.7 THEN Merge NewCase with Exemplar.
5.8 ELSE Retain NewCase as an exemplar.
5.9 Construct feature-to-category explanations.
5.10 Estimate featural importances.

Figure 5: The Protos algorithms for using and learning matching knowledge. Steps 5.1 through 5.3 describe how Protos uses matching knowledge to determine the similarity of a case and an exemplar. Steps 5.4 through 5.10 describe how Protos acquires matching knowledge.
function of Medin and Schaffer’s Context Model [28]). An important property of this definition of match strength is that matching numerous unimportant features does not compensate for failing to match an important one.

4.3 Problem 2: Indexing the Domain Theory for Efficient Classification

The simple exemplar-based method is inefficient in finding exemplars to match a new case. This is called the indexing problem. Solving this problem is particularly important for Protos. Because knowledge-based matching is computationally expensive, it must be restricted to likely matches. Furthermore, because classifications can be based on imperfect matches, “correct” exemplars must be matched before “incorrect” exemplars to avoid false-positive classifications. This section describes three types of indexing knowledge in Protos’s domain theory and the methods used to learn them. The upper part of Figure 6 describes how this knowledge is used to find an exemplar that matches a given case. The lower part of Figure 6 describes how indexing knowledge is acquired.

The first type of indexing knowledge, reminders, indexes categories and exemplars by a new case’s features (cf., [44, 22] and “cue validity” in [40]). Protos uses reminders as cues to the case’s classification. A reminding from a feature to a category, such as backrest indexing chair (Figure 7), suggests that the category is the most general classification for cases described with the feature. A reminding from a feature to an exemplar, such as pedestal indexing chairl (Figure 7), suggests that the exemplar will match cases described with the feature (cf., “idiosyncratic information” in [27]). Each reminding has an associated strength, which is used to order the list of candidate exemplars.

When searching for an exemplar that matches a new case, Protos first collects reminders to categories (step 6.1 in Figure 6). Related categories are combined by summing the strengths of duplicate reminders, and by inheriting reminders from general categories to subcategories (step 6.2). Then, Protos selects several of the most prototypical exemplars (defined below) to represent each category (step 6.3). Finally, Protos collects reminders from case features to particular exemplars (step 6.4). The result is an ordered list of exemplars to try matching with the new case.

Protos learns a reminding by compressing an expert-supplied explanation of a case feature (step 6.10). For example, Protos derives a reminding from seat to chair (Figure 7) from the explanation:

seat enables holds(person) which is the function of chair

Protos heuristically analyzes each explanation to determine the category or exemplar to which the reminding should refer and the strength of the reminding. As in the previous example, a reminding often refers to the last term in an explanation. However, some reminders are derived from a portion of the explanation. For example, one heuristic applies
GIVEN: a case \((NewCase)\) to classify.

To find an exemplar matching \(NewCase\):

6.1 Collect \textbf{remindings} from \(NewCase\)'s features to categories.
6.2 Combine remindings to related categories.
6.3 Select, in order of \textbf{prototypicality}, several exemplars of each category.
6.4 Collect \textbf{remindings} from \(NewCase\)'s features to exemplars, and add these to the list of exemplars. Order this list by reminding strength.

REPEAT (consider the exemplars in decreasing order)

6.5 Let \textit{Exemplar1} be the exemplar with the next highest reminding strength.
6.6 Determine the similarity of \(NewCase\) and \textit{Exemplar1}.

UNTIL a sufficiently strong match is found.

6.7 Use \textbf{exemplar differences} from \textit{Exemplar1} to locate a better match (\textit{Exemplar2}).

\textit{Exemplar2}'s category is used to classify \(NewCase\) and the match between \textit{Exemplar2} and \(NewCase\) is used to explain this classification. Indexing knowledge is acquired by discussing this classification and explanation with the expert, as follows.

IF the expert rejects the classification or explanation

6.8 THEN Reassess the \textbf{remindings} from \(NewCase\)'s features.

ELSE (the expert accepts the classification and explanation)

6.9 Increase \textbf{prototypicality} of \textit{Exemplar2}.

IF \(NewCase\) is retained as an exemplar

6.10 THEN Learn \textbf{remindings} for \(NewCase\).

IF \(NewCase\) was initially classified or explained incorrectly

6.11 THEN record \textbf{exemplar differences}.

Figure 6: The Protos algorithms for using and learning indexing knowledge. Steps 6.1 through 6.7 describe how Protos uses indexing knowledge to find an exemplar matching a given case. Steps 6.8 through 6.11 describe how Protos acquires indexing knowledge. Boxes highlight the different types of indexing knowledge. The process of matching a case with an exemplar (step 6.6) is described in Figure 5.
Figure 7: The Indexing Knowledge for chair. In Protos, this knowledge overlays the network of matching knowledge (Figure 3).

to explanations of the form:

\[(\text{case feature}) \cdots \text{category}_1 \text{ has specialization } \text{category}_2 \cdots\]

and derives a reminding from \(\text{case feature}\) to \(\text{category}_1\), the most general category named. The strength of a reminding is a function of the relations and qualifiers used in the explanation. For example, from the explanation:

\textit{fur is usually required by mammal which has specialization cat}

Protos derives a moderate strength reminding from \textit{fur} to \textit{mammal}.

Some remindings, called censors, are negative cues from a case feature to a category or exemplar. A censor from a feature to a category suggests that cases described with the feature should not be classified in the category. Protos uses censors to remove entries from the set of candidate classifications. Censors are derived from \textit{mutual exclusion} relations used in explanations.

Reminding are not foolproof. If a reminding suggests a classification for a new case that cannot be confirmed by matching the case with an exemplar, then Protos re-assesses the reminding (step 6.8). Matching knowledge, which is acquired and revised incrementally, may no longer justify the reminding. Protos uses the current knowledge to explain the relationship between the case feature that evokes the reminding and the classification. Protos derives a reminding from the explanation and replaces the original reminding, if they differ.

The second type of indexing knowledge, \textit{prototypicality}, indexes exemplars by the degree to which their features overlap with other exemplars in the same category \(c.f.\ “\text{family resemblance}”\) in [42] and “representativeness” in [41]). When category remindings suggest a new case’s classification, exemplars are matched with the case in decreasing order of prototypicality (step 6.3). For example, given a case with remindings to \textit{chair} but no remindings to particular exemplars, Protos attempts to match \textit{chair1} before \textit{chair2} (Figure 7).
Protos heuristically estimates an exemplar's prototypicality by counting its successful matches with new cases. Because matches typically require overlapping features, the more new cases an exemplar matches the more prototypical it becomes (step 6.9). The strength of each match determines the amount of increase.

The third type of indexing knowledge, exemplar differences, indexes exemplars by the features that distinguish them from exemplars with similar descriptions (cf., "indexing of failures" in [48, 23, 24]). After finding an exemplar that matches a new case, Protos hillclimbs to the best matching exemplar (step 6.7). For example, if the case partially matches chair2, but has the unmatched feature armrests, chair1 is suggested by the exemplar difference relating chair1 to chair2 (Figure 7).

Protos learns an exemplar difference by matching a new case to a "near miss" [53] before matching the case to an exemplar preferred by the expert (step 6.11). The near miss and the preferred exemplar may be members of the same category or different categories. Protos relates them by their differences to improve classification accuracy on subsequent cases. Relating only those pairs of exemplar that were confused during classification avoids the problem of recording a plethora of exemplar differences.

4.4 An Example of Protos in Clinical Audiology

The Protos classification and learning algorithm (Figure 8) combines the algorithms for matching and selecting exemplars. This section applies the algorithm to a typical case that Protos processed in the clinical audiology domain. After training Protos with 175 cases, the expert asks Protos to classify NewCase, which has the symptoms and test results listed in Figure 9.

There are reminders to diagnostic categories, but not to individual exemplars, from some of NewCase's features (Figure 10). Protos combines these reminders to produce an ordered list of possible classifications for NewCase (Figure 8, step 1). Duplicate reminders (such as the four reminders to cochlear) are summed, and reminders to general categories are inherited by subcategories. As a result, reminders to the general categories cochlear, age-induced cochlear, and otitis media are inherited by their shared subcategory age-induced cochlear with otitis-media.

The strongest reminding is to the category age-induced cochlear with otitis-media. Protos attempts to confirm this classification (steps 3 and 4) by explaining the similarity of NewCase and a prototypical exemplar, Patient163 (Figure 11). The match is strong, and there is no exemplar-difference knowledge that indexes a better match (step 5). The match is presented to the expert, who rejects it as incorrect (step 6).

In response to the expert's rejection, Protos re-assesses the indexing and matching knowledge that led to the misclassification. First, Protos verifies the reminders by searching the domain theory for feature-to-category explanations (step 7). Then, Protos discusses with the expert the matching knowledge that overestimated the similarity of NewCase and Patient163 (step 8). In the current match, there are no feature-to-feature explanations to be
GIVEN: a case (*NewCase*) to classify.

1. Collect and combine remindings from *NewCase*’s features.
2. Create a list of exemplars ordered by reminding strength.
   REPEAT
   
   REPEAT (consider the exemplars in decreasing order)
   3. Let *Exemplar1* be the exemplar with the next highest reminding strength.
   4. Determine the similarity of *NewCase* and *Exemplar1*.
   UNTIL an sufficiently strong match is found.
   5. Use exemplar differences from *Exemplar1* to locate a better match (*Exemplar2*).
   6. Use *Exemplar2*’s category to classify *NewCase* and use the match
      between *Exemplar2* and *NewCase* to explain this classification.
      Discuss this classification and explanation with the expert:
      IF the expert rejects the classification or explanation
      THEN Reassess the remindings from *NewCase*’s features.
      8. Request new matching knowledge from the expert.
      9. ELSE (the expert approves the classification and explanation)
      10. Increase prototypicality of *Exemplar2*.
      IF some features of *Exemplar2* are unmatched
      11. THEN request feature-to-feature explanations from the expert.
      IF the match is very strong
      12. THEN Merge *NewCase* with *Exemplar2*.
      13. ELSE Retain *NewCase* as an exemplar.
      15. Estimate featural importances.
      16. Learn remindings for *NewCase*.
      17. IF *NewCase* was initially classified or explained incorrectly
      THEN record exemplar differences.
   UNTIL the expert approves the classification and explanation.

Figure 8: The Protos Classification and Learning Algorithm, combining the algorithms in Figure 5 and Figure 6.
age_gt_60
air(mild)
bone(unmeasured)
speech(poor)
static(normal)
tymp(a)

history(noise)
s_neural(profound,2k)
acoustic_ref_u(elevated)
acoustic_ref_c(elevated)
o_acoustic_ref_u(normal)
o_acoustic_ref_c(elevated)

Figure 9: The Features of NewCase

NewCase Features
age_gt_60
air(mild)
history(noise)
speech(poor)
tymp(a)
acoustic_ref_c(elevated)
acoustic_ref_u(elevated)
o_acoustic_ref_c(elevated)

Category Reminders
noise-induced cochlear
age-induced cochlear
normal_ear
acoustic_neuroma
cochlear
mixed
otitis_media
possible_menieres
bells_palsy

Figure 10: Reminders from NewCase’s Features to Diagnostic Categories
**NewCase**

<table>
<thead>
<tr>
<th>Feature</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>air (mild)</td>
<td></td>
</tr>
<tr>
<td>bone (unmeasured)</td>
<td></td>
</tr>
<tr>
<td>tymp (a)</td>
<td></td>
</tr>
<tr>
<td>speech (poor)</td>
<td></td>
</tr>
<tr>
<td>static (normal)</td>
<td></td>
</tr>
<tr>
<td>age &gt; 60</td>
<td></td>
</tr>
<tr>
<td>s_neural (profound, 2k)</td>
<td></td>
</tr>
<tr>
<td>history (noise)</td>
<td></td>
</tr>
<tr>
<td>acoustic_ref_u (elevated)</td>
<td></td>
</tr>
<tr>
<td>o_acoustic_ref_u (normal)</td>
<td></td>
</tr>
<tr>
<td>o_acoustic_ref_u (elevated)</td>
<td></td>
</tr>
</tbody>
</table>

**Exemplar of Age-Induced Cochlear with Otitis Media**

<table>
<thead>
<tr>
<th>Patient #63</th>
</tr>
</thead>
<tbody>
<tr>
<td>air (mild)</td>
</tr>
<tr>
<td>bone (mild)</td>
</tr>
<tr>
<td>tymp (c)</td>
</tr>
<tr>
<td>speech (poor)</td>
</tr>
<tr>
<td>static (normal)</td>
</tr>
<tr>
<td>s_neural (moderate, 3k)</td>
</tr>
<tr>
<td>acoustic_ref_u (absent)</td>
</tr>
<tr>
<td>acoustic_ref_c (elevated)</td>
</tr>
<tr>
<td>o_acoustic_ref_u (elevated)</td>
</tr>
<tr>
<td>o_acoustic_ref_c (absent)</td>
</tr>
</tbody>
</table>

**Figure 11: Match between NewCase and an Exemplar of Age-Induced Cochlear with Otitis Media**

discussed, because there are only direct matches. There are some unmatched features, and Protos asks the expert to reconsider their importances. The expert tells Protos that one of these features, acoustic_ref_u (absent), is actually very important for category membership. Consequently, Protos's assessment of the similarity of the two cases decreases.

Protos attempts to classify NewCase using its second strongest reminding, which is to the category age and noise-induced cochlear. Protos selects a prototypical exemplar and explains its similarity to NewCase (Steps 3 and 4). The result is shown in Figure 12. Most of the features match directly, and the speech feature is matched using the knowledge:

**IF** the category is age and noise-induced cochlear

**THEN** speech (poor) is usually equivalent to speech (very_poor).

Of the four unmatched exemplar features, two are believed to be important: s_neural (severe, 1k) and bone (abnormal). The expert accepts the classification and the explanation.

Because important features are unmatched, Protos solicits explanations from the expert (step 11). He tells Protos that

bone (unmeasured) is sometimes equivalent to bone (abnormal)

but is unwilling to equate the value of feature s_neural in the exemplar with its value in NewCase. Because this important exemplar feature remains unmatched, Protos retains NewCase as an exemplar of age and noise-induced cochlear (step 13).
Figure 12: Match Between NewCase and an Exemplar of Age & Noise-Induced Cochlear

Protos acquires more matching knowledge by constructing feature-to-category explanations for the new exemplar (step 14). It constructs most of the explanations using existing knowledge, and the expert adds:

*acoustic_ref_c(elevated)* is spurious to *age and noise-induced cochlear*

* s_neural(profound,2k) is sometimes consistent with cochlear
  which has specialization *age-induced cochlear*
  which has specialization *age and noise-induced cochlear*.

NewCase introduced features into the domain theory, and Protos estimates their importance to the category *age and noise-induced cochlear* (step 15). Because *acoustic_ref_c(elevated)* was explained to be spurious, Protos considers it unimportant. Protos heuristically evaluates the explanation that relates *s_neural(profound,2k)* to the category. Because the relation is qualified by “sometimes,” it asserts *s_neural(profound,2k)* is moderately important to *age and noise-induced cochlear*.

Protos adds indexing knowledge for the new exemplar (step 16). Because *s_neural(profound,2k)* occurs with any type of cochlear hearing loss, Protos derives a reminding from *s_neural(profound,2k)* to *cochlear*. The presence of the qualifier “sometimes” causes Protos to assign only a moderate strength to the reminding.

Recall that Protos initially misclassified NewCase by matching it with Patient163, the exemplar of *age-induced cochlear with otitis-media*. Protos records this near-miss by adding exemplar-difference knowledge (step 17). Protos asks the expert which features reliably distinguish the exemplars and then annotates the relationship.
5 Evaluating Protos

In this section we empirically evaluate Protos's competence at the learning and classification task defined in Section 2. First we describe the results of training Protos in the audiology domain. Then we compare the classification accuracy achieved by Protos to that achieved by human audiologists and by several other programs, including simplified versions of Protos. Some of these programs have a potential advantage in this comparison because they are specifically designed to achieve high classification accuracy without regard for explanatory adequacy. On the other hand, a potential advantage for Protos is that it acquires, and uses for classification, a considerable amount of domain knowledge that the other programs are not designed to use. These comparisons are intended to determine the significance of Protos's results in audiology and to identify the aspects of Protos that are most important in achieving these results.

5.1 Protos's Competence at the Knowledge Acquisition Task

To evaluate Protos, we applied it to the task in clinical audiology of identifying a patient's hearing disorder. This domain was chosen for three reasons. First, it is a representative application of heuristic classification. Patients are assigned to diagnostic categories as a result of uncertain inferences involving features such as symptoms, test results, and patient history. Second, an interested expert was available to serve as Protos' teacher. Third, a graduate program in audiology at the University of Texas provided a population of expert and student diagnosticians to whom Protos could be compared.

Protos was trained in clinical audiology by Dr. Craig Wier, an expert audiologist and a professor of Speech Communication at the University of Texas at Austin. Dr. Wier interacted directly with Protos without a knowledge engineer's assistance. He trained Protos with 200 cases in 24 diagnostic categories. This is approximately the number of cases seen by a student during graduate school preparation for state certification. After this training, Protos was tested with 26 different cases. Learning was "turned-off" and the expert provided no explanations. Protos's competence is evident in the following figures, which describe its classification accuracy and efficiency, explanation adequacy, and autonomy gain.

Table 1 reports Protos's classification accuracy during training and testing. Accuracy improved during training, culminating in 100% accuracy during testing. Moreover, Protos's misclassifications were usually plausible categories that the expert judged were more specific than the evidence permitted.

Classification efficiency was measured in two ways, by the storage required for classification, and by the effort required to find a correct match. Figure 13 reports Protos's storage requirements, as measured by the number of categories, indices, and exemplars. Overall, Protos retained 120 exemplars of 24 diagnostic categories. Figure 14 reports the number

---

4The cases were chosen randomly from the Methodist Speech and Hearing Laboratory, Baylor College of Medicine, Houston, Texas. Professor James Jerger graciously provided access to the patient records.
<table>
<thead>
<tr>
<th>Cases</th>
<th>First Class Correct</th>
<th>Preferred Class Correct</th>
</tr>
</thead>
<tbody>
<tr>
<td>training</td>
<td>57.7</td>
<td>81.7</td>
</tr>
<tr>
<td>test</td>
<td>92.3</td>
<td>100.0</td>
</tr>
</tbody>
</table>

Table 1: **Percentage of Correct Classifications.** "First class correct" is the percentage of cases in which the strongest combined reminding resulted in the correct classification. "Preferred class correct" is the percentage of cases in which the strongest match Protos found was the correct classification. The 24 training cases that introduced new categories were excluded from these calculations.

<table>
<thead>
<tr>
<th>Cases</th>
<th>Classifications</th>
<th>Exemplars</th>
</tr>
</thead>
<tbody>
<tr>
<td>1-50</td>
<td>2.7</td>
<td>*</td>
</tr>
<tr>
<td>51-100</td>
<td>2.8</td>
<td>*</td>
</tr>
<tr>
<td>101-150</td>
<td>2.5</td>
<td>4.6</td>
</tr>
<tr>
<td>151-200</td>
<td>4.0</td>
<td>7.4</td>
</tr>
<tr>
<td>test</td>
<td>3.7</td>
<td>5.3</td>
</tr>
</tbody>
</table>

Table 2: **Protos's Mean Effort to Find a Correct Match** ("*" indicates data unavailable).

of exemplars retained in several common categories. Some categories, such as normal and unknown cochlear hearing loss, are highly polymorphic, whereas others, such as fixation, are not. During the first 100 training cases, exemplar retention was 86%. There are two reasons for this high rate of retention. First, Protos was learning two-thirds of its diagnostic categories from these cases, including many of the highly polymorphic categories. Second, the conditional explanation capability was added to Protos after the first hundred cases had been presented. The expert estimates 30–40 fewer exemplars would have been retained with conditional explanations. This is consistent with Protos’s lower rate of exemplar retention during the second hundred cases.

Table 2 reports the number of categories considered and the number of exemplar matches attempted by Protos, on average, before (and including) making a correct match. Both numbers increased at approximately the same rate as the number of categories and exemplars. The disproportionate increases that occurred with cases 151–200 were due to the large number of atypical cases in that group.  

Protos discussed with the expert every successful match, not just the strongest one. The classification could be accepted or rejected. If accepted, the expert would assess the adequacy of the explanation. As indicated in the rightmost column of Table 3, inadequate explanations were relatively rare, occurring roughly once every four or five cases.

---

6 The experiment was not restarted because the expert's time was limited.
6 Dr. Wier counted seven atypical cases in this group, compared with two in the previous group of 50 cases.
Figure 13: Protos's Acquisition of Categories, Exemplars and Indexing Knowledge
Figure 14: Exemplar Retention for a Sample of the 24 Diagnostic Categories
<table>
<thead>
<tr>
<th>Cases</th>
<th>Matches Discussed (per case)</th>
<th>Explanations Rejected (per case)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1-50</td>
<td>1.7</td>
<td>.24</td>
</tr>
<tr>
<td>51-100</td>
<td>1.6</td>
<td>.28</td>
</tr>
<tr>
<td>101-150</td>
<td>1.5</td>
<td>.22</td>
</tr>
<tr>
<td>151-200</td>
<td>1.9</td>
<td>.18</td>
</tr>
<tr>
<td>test</td>
<td>1.1</td>
<td>*</td>
</tr>
</tbody>
</table>

Table 3: Matches Discussed With the Expert

<table>
<thead>
<tr>
<th>Subjects</th>
<th>Preferred Class Correct</th>
<th>Any Class Correct</th>
</tr>
</thead>
<tbody>
<tr>
<td>Supervisor 1</td>
<td>85%</td>
<td>92%</td>
</tr>
<tr>
<td>Supervisor 2</td>
<td>69%</td>
<td>81%</td>
</tr>
<tr>
<td>Student (mean)</td>
<td>69%</td>
<td>73%</td>
</tr>
</tbody>
</table>

Table 4: Correct Classifications by Clinicians. Each clinician rank-ordered his diagnoses of each case. “Preferred Class Correct” is the percentage of cases in which the clinician’s top-ranked diagnosis was correct. “Any Class Correct” is the percentage of cases in which the clinician listed the correct diagnosis.

The number of matches Protos discussed with the expert is a measure of Protos’s autonomy. The middle column of Table 3 shows that most of Protos’s classification effort was independent of the expert. An indication that Protos gained autonomy is that the number of matches discussed with the expert remained constant as the number of exemplars and categories grew.

5.2 The Classification Accuracy Achieved by Audiologists

Protos’s classification performance was compared with that of 19 clinicians from the Department of Speech Communication of the University of Texas. Two were supervisors of the department’s clinical practicum. The other 17 were graduate students with more than one year of clinical experience. On average, the subjects did considerably less well than Protos, although one supervisor performed almost as well (Table 4).

The student clinicians commonly complained that some cases could not be diagnosed because information that they believed was essential was missing from the case descriptions. The case descriptions in the audiology data are extremely incomplete, supplying, on average, only 11 of the 58 possible features. However, the clinical supervisors found the case descriptions adequate; from years of teaching, our domain expert believes that novice audiologists frequently rely on unnecessary data.  

7The reliance of novice diagnosticians on unnecessary data has been reported in formal studies, e.g., [15].
5.3 The Classification Accuracy Achieved by ID3

The incompleteness of the case descriptions was also an obstacle for ID3, a well-known concept formation program that learns decision trees from examples. In [39], Quinlan describes four ways to adapt ID3 to cope with incomplete case descriptions. Each of these variants of ID3 was implemented, applied to the 200 audiology training cases, and evaluated by classifying the 26 audiology test cases with the resulting decision trees [13]. None of the variants of ID3 achieved high classification accuracy. The highest accuracy, a mere 38%, was achieved by treating “missing” as a feature-value. The other variants, which represent various methods of estimating missing features, achieved considerably lower classification accuracies.

In [39], the methods of estimating missing features perform well on data with the following properties:

- The percentage of features that are missing, called the “ignorance level,” is very small. Quinlan claims that “in practice, an ignorance level of even 10% is unlikely” [39, p. 99].

- Missing features are randomly distributed across cases.

   The methods perform poorly on the audiology data because the data exhibits neither of these properties. The data has an ignorance level of 81%, higher than any level studied in [39]. Additionally, missing features are not distributed randomly throughout the audiology dataset. For example, yes/no features are always missing when their value is “no.” More generally, features are missing when they are known to be irrelevant for classification or redundant with features already present in the case description.

   Protons performs well given data in which missing features have the properties exhibited by the audiology data. Although the ignorance level is very high, important features either were given or were inferable. Unimportant features, which may be missing in some cases and present in others, have little effect on the matching process.

5.4 The Classification Accuracy Achieved by Simple Exemplar-based Programs

Kibler and Aha [20] describe three similarity-based exemplar learning programs, called Proximity, Growth, and Shrink. Like Protons, these programs learn by retaining exemplars, and they classify a test case by assigning it to the category of the exemplar that best matches the test case. Unlike Protons, they do not use knowledge during classification: a test case is matched against all exemplars, and the strength of a match is determined by the number of features that are identical in the exemplar and the test case. The programs differ from each other only in the rule used to decide whether to retain a new training case as an exemplar.

---

*Very high ignorance may be quite common in practice. For example, [43] reports a 50% ignorance level.*
The matching algorithm in Kibler and Aha’s programs is reasonably well-suited to data whose missing features exhibit the properties of the audiology data. Consequently, these algorithms have a much better chance of achieving high classification accuracy on the audiology data than did any of the variants of ID3. And indeed, when the programs were implemented and tested, Shrink and Growth achieved a classification accuracy of 65%, and Proximity achieved a classification accuracy of 77% [25]. The method of calculating match strength in Kibler and Aha’s programs can be replaced with Protos’s method by assuming that no distinct features are equivalent and that all features have the same importance. When this was done, the performance of Shrink and Growth did not change, but Proximity achieved a classification accuracy of only 62% [25]. This suggests that the low-level details of Protos’s match strength calculation could be improved.

5.5 The Effect of Deleting Knowledge from Protos on Classification Accuracy

Protos derived indexing and matching knowledge from the explanations supplied by the expert. To determine the degree to which Protos’s high classification accuracy depended on these two types of knowledge, an ablation study [10] was conducted in which various combinations of Protos’s knowledge were used to classify the audiology test cases. Because this type of study would be very difficult to run on Protos itself, a program, called M-Protos, was constructed specifically for these experiments. M-Protos differs from Protos in many ways, for example, it cannot acquire knowledge. However, its classification process is sufficiently similar to Protos’s that the results of the experiments apply equally to M-Protos and Protos. Details of M-Protos and of the experimental results are given in [25].

Like Protos, M-Protos continues attempting matches, in the order specified by the indexing knowledge, until directed to stop by the expert. In the absence of indexing knowledge, M-Protos matches all exemplars with the given test case. In the absence of matching knowledge, M-Protos reduces the match strength a fixed amount for every feature in the exemplar that does not occur in the test case. M-Protos is considered to have correctly classified the test case if the strongest of its attempted matches is an exemplar in the correct category. Given all the audiology training cases, and no indexing or matching knowledge, M-Protos achieved a classification accuracy of 65%, similar to that achieved by Kibler and Aha’s programs. Oddly, the accuracy dropped to 58% when matching knowledge was added. With indexing knowledge, but no matching knowledge, M-Protos achieved 92%. With both indexing and matching knowledge, M-Protos, like Protos, achieved 100% classification accuracy.

For the most part, these results are consistent with the general philosophy of exemplar-based programs. Classification accuracy is primarily achieved by using direct match and a large, well-indexed set of exemplars; knowledge-based matching boosts classification accuracy, but is not its primary source.

However, the dependence of classification accuracy on matching knowledge in this experiment was surprisingly small. This may be explained by peculiarities of our audiology training
that will not necessarily arise in other domains, or even in the audiology domain given different training. With the case language used in our experiments, direct match worked well. In a typical correct match between an exemplar and a case, 8 of the 11 exemplar features matched features in the case, and 7 of these matched directly. Thus, there was limited opportunity to use matching knowledge, either to increase the strength of the correct match or to reduce the strength of incorrect matches stronger than the correct match. Furthermore, much of the matching knowledge that did exist was expressible only as conditional explanations of feature equivalence, a form of explanation that was not available during the first half of training. This prevented a significant body of matching knowledge from being entered.

6 Summary

Our research developed a successful approach to the task of knowledge acquisition for heuristic classification. This task is similar to concept learning. However, a program for this task has three additional requirements. First, the program must explain its classifications. Second, it must accommodate incomplete case descriptions. Third, the program must learn domain-specific knowledge for inferring case features needed for classification.

Our approach to this task is exemplar-based. Concepts are learned by retaining exemplars, and new cases are classified by matching them to similar exemplars. This approach has strong psychological support, but it raises two problems: measuring similarity and efficiently finding an exemplar to match. We solve these problems by augmenting exemplars with matching knowledge and indexing knowledge. A new case is classified by using indexing knowledge to find an exemplar and using matching knowledge to explain similarity. By interleaving heuristic classification and knowledge acquisition, this approach permits a program to start at a level of utter incompetence and to achieve a level of expert competence.

To evaluate our approach, we built the Protos program and applied it to the task in clinical audiology of identifying a patient’s hearing disorder from symptoms, test results, and history. This evaluation yielded many encouraging results:

- After a reasonable amount of training, Protos achieved very good classification accuracy without using excessive computational resources.

- Protos’s classification accuracy is comparable to that of experienced human experts, and is significantly better than that of the machine learning alternatives we examined. The two main reasons for this are (1) Protos’s ability to deal with incomplete case descriptions and (2) Protos’s use of indexing and matching knowledge derived from explanations provided by the expert.

- Most of the explanations Protos generated for a case’s classification were acceptable to the expert.

- Protos gained autonomy. The number of matches Protos discussed with the expert remained constant as the number of exemplars and categories grew.
Currently, our research labs are focusing on explanation, which includes a variety of inference methods for reasoning, learning, and communicating. Ray Bariess's research explores the use of a domain-independent model of diagnostic reasoning to guide the acquisition and generation of diagnostic explanations. Karl Branting's research [5] explores the representation and re-use of complex explanations for automated reasoning in weak-theory domains, using the domain of Workman's Compensation case law. Kenneth Murray's research [35, 34] explores the task of integrating new information into existing knowledge, which requires explaining the plausibility and the consequences of the information. The research of Liane Acker, James Lester, and Art Souther explores the generation of coherent explanations that can vary on viewpoint and level of abstraction. Both of the last two projects use a large-scale, multifunctional knowledge base for the domain of plant anatomy, physiology, and development [37].

Acknowledgements

Many of our colleagues have contributed to this research. Dr. Craig Wier trained Protos in clinical audiology and counseled us on evaluation methods and user interface design. Richard Mallory performed the ablation study. Claudia Porter, Joe Ross, and Todd Stock contributed to Protos's construction. Kenneth Murray and Adam Farquhar were instrumental in Protos's design. Liane Acker, Brad Blumenthal, Karl Branting, James Lester, Richard Mallory, and Kenneth Murray reviewed early drafts of this paper.

References


