Concept Learning and Heuristic Classification in Weak-Theory Domains[1]

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

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

Robert C. Holte
Computer Science Department,
University of Ottawa,
Ottawa, Ontario, Canada K1N 6N5

[1]Support for this research was provided by grants from the Army Research Office (ARO-DAAG29-84-K-0060), and the National Science Foundation (IRI-8620052), and by contributions from Apple Corporation, Texas Instruments, and the Cray Foundation. This research was conducted in the Department of Computer Sciences, University of Texas at Austin.

Abstract

This paper describes a successful approach to concept learning 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 other machine learning 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 concept learning for heuristic classification. This task differs from the usual concept learning task in three ways. 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 many domains.

An alternative approach relies on exemplars. Concepts are learned by retaining 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" [14] 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 shortcomings of Protos that became evident in these studies.

2 The Task: Concept Learning for Classification and Explanation

Our research addresses the task of improving competence at classification and explanation, starting at a level of utter incompetence and aiming for a level of expert competence. Improvement results from learning concepts and acquiring knowledge from cases which are classified and explained by a human expert. This section specifies this task in detail by defining classification, explanation, and knowledge acquisition, and by giving measures of competence for each. Figure 1 illustrates this task with a modestly abridged version of an actual dialogue between Protos and a domain expert (Section 4.4 is a detailed description of this scenario).

Expert: *NewCase* has the features ...
Protos: *NewCase*'s classification appears to be *age-induced cochlear with otitis-media* because of its similarity with *Patient*163; some of the features of the cases are identical.
Expert: Wrong diagnosis.
Protos: *NewCase* is missing some of the features of *Patient*163 which I thought were unimportant. Are any of *bone(mild), acoustic_ref_u(absent),...* more important than I thought?
Expert: Yes. *Acoustic_ref_u(absent)* is very important for membership in the category *age-induced cochlear with otitis-media.*
Protos: I'll remember that. Now *NewCase*'s classification appears to be *age and noise-induced cochlear* because of its similarity with *Patient*054; 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)*.
Expert: The category and explanation are correct.
Protos: There are two weak points in the explanation I'd like to clear up. First, the feature *bone(abnormal)* is important in *Patient*054, but is not present in *NewCase*.
Expert: *Bone(abnormal)* is sometimes equivalent to *bone(unmeasured)*, which is present in *NewCase*.
Protos: OK. The other weak point is that feature *s_neural(severe,1k)* is important in *Patient*054, but is not present in *NewCase*.
Expert: Yes, that is a genuine difference between the two cases.
Protos: Then I'll retain *NewCase* as a separate exemplar of this category. Is there anything else I should know about this case?
Expert: Yes, *acoustic_ref_c(elevated)* is spurious to *age and noise-induced cochlear*; ...

Figure 1: **An Abridged Dialogue between Protos and a Domain Expert in Clinical Audiology.** The expert presents a new case to Protos. Protos classifies the case by matching it with similar exemplars, and explains each classification by presenting the match. Protos acquires knowledge from the expert to improve the correct match, and retains the case as a new exemplar.

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 *win-for-white-in-9-ply* and *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 [11] 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 nonhierarchical, uncertain inferences"[11, p. 290]. Protos's classification method, although it differs from Clancey's, is also appropriate for 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 [59]. However, we have adopted a simplified notion of explanation in order to concentrate on other aspects of the task. 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. Explanations are not used to teach domain knowledge or to elaborate previous explanations (see [10, 38] for recent research on these tasks).

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.,* "*bone(abnormal)* is sometimes equivalent to *bone(unmeasured)*", in Figure 1.)

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 particu-

[2]Terms appear in boldface when they are defined.

lar conclusion was reached, Mycin [53] lists only the satisfied rules leading to the conclusion; 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 complex processes such as natural language generation or adaptation to a model of the person to whom explanations are targeted.

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, knowledge is elicited in the form of **training cases** which are classified and explained by the expert.

Knowledge acquisition is mixed initiative. The program may request particular knowledge from the expert, or the expert may take the initiative and volunteer knowledge. The knowledge provided by the expert and the explanations produced by the program are expressed in the same language. New vocabulary (*i.e.,* categories and features) can be introduced by the expert 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 program's ability to gain autonomy is important because greater autonomy is required in the later stages of knowledge base development than in the early stages. Most existing concept learning and knowledge acquisition programs do not gain autonomy, and so are appropriate for only a single stage of a knowledge base's development [4]. For example, ETS [7] and ROGET [6] create a knowledge base by eliciting the basic terminology and conceptual structure of a domain. However, they are inappropriate for later stages of development such as refinement and reformulation. A full discussion of this issue and a survey of existing programs, including Protos, is given in [5].

3 Weaknesses of Generalization-Based Methods for Learning and Classification

Almost all programs that learn to classify 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 [44], CN2 [13], and connectionist programs (*e.g.,* [49]). Examples of theory-based programs are explanation-based programs (*e.g.,* [16, 36]) and similarity-based programs that use background knowledge (*e.g.,* [37, 35, 26, 21]). 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 nonintrinsic 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* [61] and *hammer* [15] are defined in terms of function, not form. Categories such as *infected by pseudomonas* [53] and *seedless grape* [2] are generalizations defined in terms that are not readily perceivable in the context of classification.

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 [22, 23]. 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 [52].

The developers of theory-based programs have acknowledged this severe limitation [36], 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 [55, 51, 2, 12, 62].

Given:

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

REPEAT

Classify:

$\boxed{\text{Find}}$ an exemplar of $c_j \in C$ that $\boxed{\text{Strongly Matches}}$ *NewCase*
and classify *NewCase* as c_j.
$\boxed{\text{Explain}}$ the classification.

Learn:

If the expert disagrees with the classification or explanation then
acquire classification and explanation knowledge and $\boxed{\text{adjust}}$ C
so that *NewCase* is correctly classified and explained.

UNTIL the expert approves the classification and explanation.

Figure 2: **Exemplar-Based Learning and Classification Algorithm.** The hard problems of the exemplar-based approach are boxed.

hi

Figure 3: **A Portion of the Exemplar-Based Category** *chair*

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 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 [19].

4.1 Simple Exemplar-Based Concept Learning and Classification

Figure 2 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 3 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, 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.

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 [32]. Recent research indicates that people resist generalization and retain cases. For example, Medin[32] and Brooks[9] found that people classify previously seen cases by direct matching. Tversky and Kahneman[57] 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 [24] 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 [46, 33, 34, 55, 50].

The simple exemplar-based method uses no domain theory. However, domain theory is indispensable for solving the hard problems indicated in Figure 2. For example, determining the strength of the match between an exemplar and a case requires knowing the basis for category membership. Murphy and Medin [39] 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 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 4, there are two types of matching knowledge. The first is

hi

Figure 4: **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.)

relations among concepts, for example "*seat* enables *holds(person)*." The second type of matching knowledge is featural importances. As defined by Medin and Schaffer [33], 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 4, Protos can match *pedestal* with *legs(4)* because of the relations connecting each of them to *seat support*. These paths are an integral part of the explanation of the classification of a new case. For example, Protos could explain that *chair2* is a *chair* because it resembles *chair1*, and the resemblance is strong because all the differences between them can be explained away. 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 [31, 55]. 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 category is the amount of unexplained variability among the members of a category [47]. In domains with a strong theory, the polymorphy of most categories will be low. In domains with a weak theory, the polymorphy of categories can vary considerably (*e.g.,* see Figure 15 in Section 5.1). Because of this, Protos has been designed to cope with categories 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 not retained. Thus, the number of exemplars that Protos retains for a category is a direct indication of the category's polymorphy.

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 4) was learned when the expert classified *chair1* and explained:

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 (see Figure 5). The relations fall into three certainty classes:

1. Definitional relations denote invariant facts (*e.g.,* "*adolescent* definitionally entails *minor*").
2. Causal/functional relations denote known mechanisms. (*e.g.,* "*air pollution* causes *acid rain*").
3. Correlational relations denote experiential knowledge (*e.g.,* "*sharp teeth* suggest *carnivorous*").

Each relation can be strengthened or weakened with qualifiers, such as *always, usually, sometimes,* or *occasionally.*

Some explanations, called **conditional**, are restricted to members of a particular category, cases with particular features, or matches with particular exemplars. For example,

IF the category is *apples* THEN *color(red)* is equivalent to *color(green)*

explains that the colors red and green are equivalent for the purposes of classifying apples. Conditional explanations are essential for concepts defined functionally such as *hammer.* They permit stating that a claw is spurious for a hammer *qua* nail-inserter but is essential for a hammer *qua* nail-extractor.

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

Definitional:

definitionally entails
is equivalent to
requires
if and only if
has generalization
part of
is mutually exclusive with

Causal/Functional:

causes
has function
enables

Correlational:

co-occurs with
is consistent with
implies
suggests
spurious to

Figure 5: **Relations in Protos's Explanation Language**. The relations are in equivalence classes with respect to certainty. Each relation has an inverse which is not shown (*e.g.,* the inverse of *has generalization* is *has specialization.*)

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 4). 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 an unacceptable explanation.

The lower part of Figure 6 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.

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 6, 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 [58, 1, 42], 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 6.2 in Figure 6). 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 the sense that the features suggest the same classification.

Knowledge-based matching uses 38 domain-independent heuristics to evaluate the quality of a path ([4, 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 correlational relations such as "sometimes implies" in a causal 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* ..."

GIVEN: a case (*NewCase*) to classify, and an exemplar (*Exemplar*).

To determine the similarity of *NewCase* and *Exemplar*:

6.1 Assign a high match strength to each feature of *Exemplar* that directly matches a feature of *NewCase*.
6.2 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.
6.3 Compute the overall similarity of *Exemplar* and *NewCase* using the match strength of each matched exemplar feature and the importance 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
6.4 THEN Request new matching knowledge from the expert.
6.5 ELSE (the expert accepts the classification and explanation)
IF some features of *Exemplar* are unmatched
6.6 THEN request feature-to-feature explanations from the expert.
IF the match is very strong
6.7 THEN Discard *NewCase*.
6.8 ELSE Retain *NewCase* as an exemplar.
6.9 Construct feature-to-category explanations.
6.10 Estimate featural importances.

Figure 6: **The Protos algorithms for using and learning matching knowledge.** Steps 6.1 through 6.3 describe how Protos uses matching knowledge to determine the similarity of a case and an exemplar. Steps 6.4 through 6.10 describe how Protos acquires matching knowledge.

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 6.3) 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 of the exemplar's features (*cf.,* the similarity function of Medin and Schaffer's Context Model [33]). 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.

Using matching knowledge, the similarity of related cases is more accurately estimated. However, the matching knowledge in a weak domain theory is not perfect, and these imperfections can cause the similarity of unrelated cases to be overestimated. For example,

- Featural importances are generally ball park approximations of a quantity whose exact value is unknown. Small numerical differences in featural importance values, which ought to be negligible, can accumulate during match-strength calculation and distort the similarity estimate.

- expert-supplied explanations of relations between concepts can be overly general, and inadvertently "explain away" the significant differences between unrelated cases. (*cf.* "promiscuous theories" in [17].)

By causing the similarity of unrelated cases to be overestimated while causing the similarity of related cases to be accurately estimated, matching knowledge can result in the misclassification of cases that would be correctly classified using a simple feature-counting measure of similarity. In Protos, this problem is approached in two ways. First, matching knowledge is revised whenever it leads to a misclassification (step 6.4). Secondly, a new case is not matched against every exemplar. Protos's method for selecting the exemplars with which to match a new case is discussed next.

4.3 Indexing the Domain Theory

In Protos, a new case is matched only with those exemplars most likely to be correct. By doing this, Protos overcomes two serious problems that arise when a new case is matched with all known exemplars:

- misclassifications due to the inappropriate use of matching knowledge, as described above.

GIVEN: a case (*NewCase*) to classify.

To find an exemplar matching *NewCase*:

7.1 Collect $\boxed{\text{remindings}}$ from *NewCase*'s features to categories.
7.2 Combine remindings to related categories.
7.3 Retain the N categories with the strongest combined remindings.
7.4 Select, in order of $\boxed{\text{prototypicality}}$, several exemplars of each category.
7.5 Collect $\boxed{\text{remindings}}$ 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)
7.6 Let *Exemplar1* be the exemplar with the next highest reminding strength.
7.7 Determine the similarity of *NewCase* and *Exemplar1*.
UNTIL a sufficiently strong match is found.
7.8 Use $\boxed{\text{exemplar differences}}$ from *Exemplar1* to locate a better match (*Exemplar2*).

Exemplar2's category is used to classify *NewCase* and the match between *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
7.9 THEN Reassess the $\boxed{\text{remindings}}$ from *NewCase*'s features.
ELSE (the expert accepts the classification and explanation)
7.10 Increase $\boxed{\text{prototypicality}}$ of *Exemplar2*.
IF *NewCase* is retained as an exemplar
7.11 THEN Learn $\boxed{\text{remindings}}$ for *NewCase*.
IF *NewCase* was initially classified or explained incorrectly
7.12 THEN record $\boxed{\text{exemplar differences}}$.

Figure 7: **The Protos algorithms for using and learning indexing knowledge.** Steps 7.1 through 7.8 describe how Protos uses indexing knowledge to find an exemplar matching a given case. Steps 7.9 through 7.12 describe how Protos acquires indexing knowledge. Boxes highlight the different types of indexing knowledge. The process of matching a case with an exemplar (step 7.7) is described in Figure 6.

hi

Figure 8: **The Indexing Knowledge for** *chair*. In Protos, this knowledge overlays the network of matching knowledge (Figure 4).

- the inefficiency of applying the computationally expensive process of knowledge-based matching to many exemplars.

To select and order the exemplars to match with a given case, Protos uses three types of **indexing knowledge**: remindings, prototypicality, and exemplar differences. This section describes this knowledge, how it is used (see the upper part of Figure 7) and how it is learned (see the lower part of Figure 7).

The first type of indexing knowledge, **remindings**, indexes categories and exemplars by a new case's features (*cf.,* [50, 27] and "cue validity" in [45]). Protos uses remindings as cues to the case's classification. A reminding from a feature to a category, such as *backrest* indexing *chair* (Figure 8), 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 *chair1* (Figure 8), suggests that the exemplar will match cases described with the feature (*cf.,* "idiosyncratic information" in [32]). 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 remindings to categories (step 7.1 in Figure 7). Related categories are combined by summing the strengths of duplicate remindings and by inheriting remindings from general categories to subcategories (step 7.2). Only the N strongest combined remindings are returned (step 7.3).[3] Protos then selects several of the most prototypical exemplars (defined below) to represent each category (step 7.4). Finally, Protos collects remindings from case features to particular exemplars (step 7.5). The result is an ordered list of exemplars to try matching with the new case.

Protos learns a reminding by compiling an expert-supplied explanation of a case feature (step 7.11). For example, Protos derives a reminding from *seat* to *chair* (Figure 8) 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 remindings are derived from a portion of the explanation. For example, one heuristic applies to explanations of the form:

[3] In the current implementation, $N = 5$.

$$\langle \text{case feature} \rangle \cdots \textit{category}_1 \text{ has specialization } \textit{category}_2 \cdots$$

and derives a reminding from $\langle$ case feature $\rangle$ to $category_1$, 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:

fur is usually required by *mammal* which has specialization *cat*

Protos derives a moderate strength reminding from *fur* to *mammal*.

Some remindings, called **censors**, are negative associations between case features and categories or exemplars. 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 *mutual exclusion* relations used in explanations.

Remindings are not foolproof. The explanation from which a reminding is derived might change. For example, a reminding from a feature f to a category C might be derived from the explanation that "f causes C". Later, the domain expert might change the explanation to "f sometimes causes C", or "f causes C_2 which has specialization C". Protos does not check remindings when matching knowledge is altered. Rather, a reminding is reassessed only when it suggests an incorrect classification (step 7.9). Protos uses the current matching knowledge to explain the relationship between the classification and the case feature that evokes the reminding. From this explanation Protos derives a replacement for the faulty reminding.

The second type of indexing knowledge, **prototypicality**, orders the exemplars within a category according to their record of success in previous classifications. When a new case reminds Protos of a category, the exemplars of that category are matched with the case in order of decreasing prototypicality (step 7.4). For example, given a case with remindings to *chair* but no remindings to particular exemplars, Protos attempts to match the case with *chair1* before *chair2* (Figure 8). Prototypicality is a heuristic estimate of the psychological notion of "family resemblance" [47], which is the degree to which an exemplar is similar to other category members. Prototypicality is incrementally learned: if a match between a case and an exemplar is accepted by the expert then the exemplar's prototypicality is increased by an amount proportional to the strength of the match (step 7.10).

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 [54, 28, 29]). After finding an exemplar that matches a new case, Protos hillclimbs to the best matching exemplar (step 7.8). 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 8).

Protos learns an exemplar difference by matching a new case to a "near miss" [60] before matching the case to an exemplar preferred by the expert (step 7.12). 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 exemplars that were actually 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 9) 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. The dialogue in Figure 1 provides an outline of this section. There, the interaction between Protos and the expert on this case is given in an abridged form. This section presents and discusses the unabridged form of the interaction – *e.g.,* Figures 10 through 13 are screen-dumps of actual Protos output – and Protos's internal processing.

After training Protos with 175 cases, the expert asks Protos to classify *NewCase*, which has the features (symptoms and test results) listed in Figure 10.

There are remindings from some of these features to diagnostic categories, but no remindings to individual exemplars (Figure 11). Protos combines these remindings to produce an ordered list of possible classifications for *NewCase* (Figure 9, step 1). Duplicate remindings (such as the four remindings to *cochlear*) are summed, and remindings to general categories are inherited by subcategories. As a result, remindings 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 combined 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 12). 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 reassesses the indexing and matching knowledge that led to the misclassification. First, Protos verifies the remindings 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 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 13. Most of the features match directly, and the *speech* feature is matched using the knowledge:

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

```
GIVEN: a case (NewCase) to classify.
1     Collect and combine remindings from NewCase's features.
      Retain only the N strongest combined remindings.
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
7           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 Discard NewCase.
13                ELSE Retain NewCase as an exemplar.
14                      Construct feature-to-category explanations.
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 9: **The Protos Classification and Learning Algorithm**, combining the algorithms in Figure 6 and Figure 7.

hi

Figure 10: **The Features of** *NewCase*

hi

Figure 11: **Remindings from** *NewCase*'**s Features to Diagnostic Categories**

hi

Figure 12: **Match between** *NewCase* **and an Exemplar of** *Age-Induced Cochlear with Otitis Media*

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).

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

hi

Figure 13: **Match Between** *NewCase* **and an Exemplar of** *Age & Noise-Induced Cochlear*

qualified by "sometimes," it infers that *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)* is explained to occur 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. Before concluding, Protos asks the expert to provide any additional information he wishes, and he declines.

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. We then 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's 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 and tested in clinical audiology by Dr. Craig Wier, an expert audiologist and a professor of Speech Communication at the University of Texas at Austin. Cases were chosen at random from the Speech and Hearing Laboratory of the Methodist Hospital in Houston, Texas.[4] A case was considered **correctly diagnosed** if Dr. Wier and

[4]Professor James Jerger graciously provided access to the patient records.

Cases	*First Class Correct*	*Preferred Class Correct*
training	57.7	81.7
test	92.3	100.0

Table 1: **Percentage of Correct Classifications.** "First class correct" is the percentage of cases in which the strongest combined reminding which produced a matching exemplar 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.

the laboratory clinicians agreed. (Of an original set of 230 cases, they disagreed on 4, which were not used for Protos's training or testing.) Protos's **classification accuracy** is the percentage of cases which it correctly diagnosed.

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, efficiency, and autonomy gain.

Table 1 reports Protos's classification accuracy during training and testing. Accuracy improved during training, culminating in 100% agreement with the expert 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 14 illustrates the growth of Protos's storage requirements, as measured by the number of categories, indices, and exemplars. Overall, Protos retained 120 exemplars of 24 diagnostic categories. Figure 15 reports the number 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.[5] 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.

[5]The experiment was not restarted because the expert's time was limited.

hi

Figure 14: **Protos's Acquisition of Categories, Exemplars and Indexing Knowledge**

hi

Figure 15: **Exemplar Retention for a Sample of the 24 Diagnostic Categories**

The disproportionate increases that occurred with cases 151–200 were due to the large number of atypical cases in that group.[6]

Protos discussed every successful match with the expert, not just the strongest one. The classification could be accepted or rejected. If accepted, the expert would assess the adequacy of the explanation. The expert rarely disagreed with Protos's explanations.

The number of matches Protos discussed with the expert is a measure of Protos's autonomy. 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.

[6]Dr. Wier counted seven atypical cases in this group, compared with two in the previous group of 50 cases.

Cases	*Classifications*	*Exemplars*
1-50	2.7	*
51-100	2.8	*
101-150	2.5	4.6
151-200	4.0	7.4
test	3.7	5.3

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

Cases	*Matches Discussed (per case)*
1-50	1.7
51-100	1.6
101-150	1.5
151-200	1.9
test	1.1

Table 3: **Matches Discussed With the Expert**

Subjects	*Preferred Class Correct*	*Any Class Correct*
Supervisor 1	85%	92%
Supervisor 2	69%	81%
Student (mean)	69%	73%

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 case descriptions in the audiology data are extremely incomplete, supplying, on average, less than half the possible features. However, the clinical supervisors found the case descriptions adequate. [7]

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 [44], 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 [18]. 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 [44], 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" [44, p.

[7] From years of teaching, our domain expert believes that novice audiologists frequently rely on unnecessary data. The reliance of novice diagnosticians on unnecessary data has also been reported in formal studies, *e.g.*, [20].

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 exceeding 50%.[8] Furthermore, missing features are not distributed randomly throughout the audiology data set. 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.

Protos 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 [25] describe three similarity-based exemplar learning programs, called Proximity, Growth, and Shrink. Like Protos, 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 Protos, 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.

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% [30]. 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% [30]. This suggests that the low-level details of Protos's match strength calculation could be improved.

[8] Very high ignorance may be quite common in practice. For example, [48] reports a 50% ignorance level.

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 [14] 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, notably that 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 [30].

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 of 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. 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. Lack of indexing knowledge resulted in the inappropriate use of matching knowledge and a concomitant increase in misclassifications (as discussed at the end of section 4.2.2.).

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. 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 by the expert.

6 Summary

Our research developed a successful approach to the task of concept learning for heuristic classification. This task differs from the usual concept learning task in three ways. A program for this task must explain its classifications, accommodate incomplete case descriptions, and 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.

The evaluation of Protos also revealed several shortcomings, which we are addressing with our current research. The first shortcoming is that Protos adds new knowledge without considering its relation to existing knowledge. This allows inconsistencies to go undetected until they cause a classification or explanation failure, which is sometimes undesirable. Ken Murray's research [41, 40] explores the task of integrating new information into existing knowledge. The second shortcoming of Protos is that it does not adapt its explanations to different users. The research of Liane Acker, James Lester, and Art Souther [56] explores the generation of coherent explanations that can vary in viewpoint and level of abstraction. This research and Ken Murray's both use a large-scale, multifunctional knowledge base for the domain of plant anatomy, physiology, and development [43]. The third shortcoming of

Protos is its very restricted explanation language. Karl Branting's research [8] explores the representation and reuse of complex explanations for automated reasoning in weak-theory domains, using as a testbed the domain of Worker's Compensation case law.

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

[1] J.S. Aikins. A representation scheme using both frames and rules. In B.G. Buchanan and E.H. Shortliffe, editors, *Rule Based Expert Systems*. Addison-Wesley, 1984.

[2] J. Amsterdam. Some philosophical problems with formal learning theory. *Proceedings of AAAI-88*, pages 580–584, 1988.

[3] E.R. Bareiss. *Exemplar-Based Knowledge Acquisition*. Academic Press, 1989. (Also available as a PhD dissertation from the Department of Computer Science, University of Texas at Austin).

[4] E.R. Bareiss, B.W. Porter, and K.S. Murray. Gaining autonomy during knowledge acquisition. Technical Report AI89-97, University of Texas at Austin, Computer Sciences Department, 1989.

[5] E.R. Bareiss, B.W. Porter, and K.S. Murray. Supporting start-to-finish development of knowledge bases. *Machine Learning*, 1989. (to appear in Volume 3, Number 6).

[6] J.S. Bennett. ROGET: A knowledge-based system for acquiring the conceptual structure of a diagnostic expert system. *Automated Reasoning*, 1(1):49–74, 1985.

[7] J. Boose. Personal construct theory and the transfer of expertise. In *Proceedings of the National Conference on Artificial Intelligence*, pages 27–33, 1984.

[8] L.K. Branting. Integrating generalizations with exemplar-based reasoning. In *Proceedings of the Eleventh Cognitive Science Society Conference*, 1989.

[9] L. Brooks. Non-analytic concept formation and the memory for instances. In E. Rosch and B.B. Lloyd, editors, *Cognition and Categorization*, pages 169–211. Erlbaum, Hillsdale, NJ, 1978.

[10] W.J. Clancey. Extensions to rules for explanation and tutoring. In B.G. Buchanan and E.H. Shortliffe, editors, *Rule Based Expert Systems*, pages 531–568. Addison-Wesley, 1984.

[11] W.J. Clancey. Heuristic classification. *Artificial Intelligence*, 27:289–350, 1985.

[12] P. Clark. A comparison of rule and exemplar-based learning systems. In Pavel Brazdil, editor, *Proceedings of the International Workshop on Machine Learning, Meta-Reasoning, and Logic*. University of Porto, Portugal, 1988.

[13] P. Clark and T. Niblett. The CN2 induction algorithm. *Machine Learning*, 3(4):261–283, 1989.

[14] P. Cohen and A. Howe. How evaluation guides AI research. *The AI Magazine*, pages 35–43, Winter 1988.

[15] J.H. Connell and M. Brady. Generating and generalizing models of visual objects. *Artificial Intelligence*, 31:159–183, 1987.

[16] G. DeJong and R. Mooney. Explanation-based learning: An alternative view. *Machine Learning*, 1(2), 1986.

[17] T.G. Dietterich. Annual review of computer science. *Machine Learning*, 4, 1990. (to appear).

[18] R.T. Duran. Concept learning with incomplete data sets. Master's thesis, Department of Computer Science, University of Texas at Austin, 1988. (Available as technical report AI88-82).

[19] D.L. Dvorak. A guide to CL-Protos: An exemplar-based learning apprentice. Technical Report AI88-87, Department of Computer Science, University of Texas at Austin, 1988.

[20] P.J. Feltovich, P.E. Johnson, J.H. Moller, and D.B. Swanson. LCS: the role and development of medical knowlege in diagnostic expertise. In W.J. Clancey and E.H. Shortliffe, editors, *Readings in Medical Artificial Intelligence*, pages 275–319. Addison-Wesley, Reading, MA, 1984.

[21] N.S. Flann and T.G. Dietterich. Selecting appropriate representations for learning from examples. *Proceedings of AAAI-87*, 1987.

[22] A.L. Gardner. *An Artificial Intelligence Approach to Legal Reasoning*. PhD thesis, Stanford University, 1984.

[23] H.L.A. Hart. Positivism and the separation of law and morals. *Harvard Law Review*, 71:593–629, 1958.

[24] K.J. Holyoak and A.L. Glass. The role of contradictions and counterexamples in the rejection of false sentences. *Journal of Verbal Learning and Verbal Behavior*, 14:215–239, 1975.

[25] D. Kibler and D. Aha. Learning representative exemplars of concepts: An initial case study. In Pat Langley, editor, *Proceedings of the Fourth International Workshop on Machine Learning*, pages 24–30. Los Altos: Morgan Kaufmann, 1987.

[26] Y. Kodratoff and J.G. Ganascia. Improving the generalization step in learning. In R.S. Michalski, J.G. Carbonell, and T.M. Mitchell, editors, *Machine Learning: An Artificial Intelligence Approach*, volume 2, pages 215–244. Los Altos: Morgan Kaufmann, 1986.

[27] J. L. Kolodner. Maintaining organization in a dynamic long-term memory. *Cognitive Science*, 7(4):243–280, 1983.

[28] J.L. Kolodner. Extending problem solver capabilities through case-based inference. In *Proceedings of the Fourth International Workshop on Machine Learning*, pages 167–178, 1987.

[29] J.L. Kolodner and R.M. Kolodner. Using experience in clinical problem solving. Technical Report GIT-ICS-85/21, School of Info. and Computer Science, Georgia Institute of Technology, 1985.

[30] R.S. Mallory. Sources of classification accuracy in Protos. Technical Report AI89-XX, Department of Computer Science, University of Texas at Austin, 1989. (forthcoming).

[31] L.T. McCarty and N.S. Sridharan. A computational theory of legal argument. Technical Report LRP-TR-13, Laboratory for Computer Science Research, Rutgers University, 1982.

[32] D.L. Medin, G.I. Dewey, and T.D. Murphy. Relationships between item and category learning: Evidence that abstraction is not automatic. *Journal of Experimental Pyschology: Learning, Memory and Cognition*, 9(4):607–625, 1983.

[33] D.L. Medin and M.M. Schaffer. Context theory of classification learning. *Psychological Review*, 85:207–238, 1978.

[34] D.L. Medin and E.E. Smith. Concepts and concept formation. *Annual Review of Psychology*, 35:113–138, 1984.

[35] R.S. Michalski. A theory and methodology of inductive learning. In R.S. Michalski, J.G. Carbonell, and T.M. Mitchell, editors, *Machine Learning: An Artificial Intelligence Approach*, volume 1. Palo Alto: Tioga Publishing, 1983.

[36] T.M. Mitchell, R.M. Keller, and S.T. Kedar-Cabelli. Explanation-based generalization: A unifying view. *Machine Learning*, 1(1):47–80, 1986.

[37] T.M. Mitchell, P.E. Utgoff, and R. Banerji. Learning by experimentation: Acquiring and refining problem solving heuristics. In R.S. Michalski, J.G. Carbonell, and T.M. Mitchell, editors, *Machine Learning: An Artificial Intelligence Approach*, volume 1, pages 163–190. Palo Alto: Tioga Publishing, 1983.

[38] J. D. Moore and W.R. Swartout. A reactive approach to explanation. In *Fourth International Workshop on Natural Language Generation*, 1989.

[39] G.L. Murphy and D.L. Medin. The role of theories in conceptual coherence. *Psychological Review*, 92:289–316, 1985.

[40] K.S. Murray. KI: An experiment in automating knowledge acquisition. Technical Report AI88-90, Department of Computer Science, University of Texas at Austin, 1988.

[41] K.S. Murray and B.W. Porter. Developing a tool for knowledge integration: Initial results. In *Proceedings of the Third Knowledge Acquisition for Knowledge-based Systems Workshop*, 1988.

[42] H.P. Nii, E.A. Feigenbaum, J.J. Anton, and A.J. Rockmore. Signal-to-symbol transformation: HASP/SIAP case study. *The AI Magazine*, pages 23–35, Spring 1982.

[43] B. Porter, J. Lester, K. Murray, K. Pittman, A. Souther, L. Acker, and T. Jones. AI research in the context of a multifunctional knowledge base: The Botany Knowledge Base project. Technical Report AI88-88, Department of Computer Science, University of Texas at Austin, 1988.

[44] J.R. Quinlan. Induction of decision trees. *Machine Learning*, 1(1):81–106, 1986.

[45] E. Rosch. Principles of categorization. In E. Rosch and B.B. Lloyd, editors, *Cognition and Categorization*. Hillsdale, NJ:Erlbaum, 1978.

[46] E. Rosch. Prototype classification and logical classification: The two systems. In E.K. Scholnick, editor, *New Trends in Conceptual Representation: Challenges to Piaget's Theory?* Hillsdale, NJ:Erlbaum, 1983.

[47] E. Rosch and C.B. Mervis. Family resemblance: Studies in the internal structure of categories. *Cognitive Psychology*, 7:573–605, 1975.

[48] K. Ruberg, S.M. Cornick, and K.A. James. House calls: Building and maintaining a diagnostic rule-base. In J.H. Boose and B.R. Gaines, editors, *Proceedings of the 3rd Knowledge Acquisition for Knowledge-Based Systems Workshop*. Computer Science Department, University of Calgary, Canada, 1988.

[49] D. Rumelhart and J. McClelland, editors. *Parallel Distributed Processing*, volume 1. Cambridge, MA: MIT Press, 1986.

[50] R.C. Schank. *Dynamic Memory: A Theory of Reminding and Learning in Computers and People*. Cambridge University Press, 1982.

[51] R.C. Schank, G.C. Collins, and L.E. Hunter. Transcending inductive category formation in learning. *The Behavioral and Brain Sciences*, 1986.

[52] A.D. Shapiro. *Structured Induction in Expert Systems*. Addison-Wesley, 1987.

[53] E.H. Shortliffe. *MYCIN: Computer-Based Medical Consultations*. New York: American Elsevier, 1976.

[54] R.L. Simpson. *A Computer Model of Case-Based Reasoning in Problem Solving: An Investigation in the Domain of Dispute Mediation*. PhD thesis, School of Information and Computer Science, Georgia Institute of Technology, 1985.

[55] E. Smith and D. Medin. *Categories and Concepts*. Cambridge: Harvard University Press, 1981.

[56] A. Souther, L. Acker, J. Lester, and B. Porter. Using view types to generate explanations in intelligent tutoring systems. In *Proceedings of the Eleventh Cognitive Science Society Conference*, 1989.

[57] A. Tversky and D. Kahneman. Judgment under uncertainty: Heuristics and biases. *Science*, 185:1124–1131, 1974.

[58] S.M. Weiss, A.K. Casimir, and S. Amarel. A model-based method for computer-aided medical decision-making. *Artificial Intelligence*, 11:145–172, 1978.

[59] M.R. Wick, C.L. Paris, W.R. Swartout, and W.B. Thompson, editors. *Proceedings of the AAAI'88 Workshop on Explanation*. American Association for Artificial Intelligence, 1988.

[60] P.H. Winston. Learning structural descriptions from examples. In P.H. Winston, editor, *The Psychology of Computer Vision*, pages 157–209. McGraw-Hill, 1975.

[61] P.H. Winston, T.O. Binford, B. Katz, and M. Lowry. Learning physical descriptions from functional definitions, examples, and precedents. *Proceedings of AAAI-83*, pages 433–439, 1983.

[62] L. Wittgenstein. *Philosophical Investigations*. New York: MacMillan, 1953.