Proceedings of the Tenth International Conference on Machine Learning, 1993, Morgan Kaufmann, pages 25–32.

Using Decision Trees to Improve Case-Based Learning

Claire Cardie
Department of Computer Science
University of Massachusetts
Amherst, MA 01003
E-mail: cardie@cs.umass.edu

Abstract

This paper shows that decision trees can be used to improve the performance of case-based learning (CBL) systems. We introduce a performance task for machine learning systems called semi-flexible prediction that lies between the classification task performed by decision tree algorithms and the flexible prediction task performed by conceptual clustering systems. In semi-flexible prediction, learning should improve prediction of a specific set of features known a priori rather than a single known feature (as in classification) or an arbitrary set of features (as in conceptual clustering). We describe one such task from natural language processing and present experiments that compare solutions to the problem using decision trees, CBL, and a hybrid approach that combines the two. In the hybrid approach, decision trees are used to specify the features to be included in k-nearest neighbor case retrieval. Results from the experiments show that the hybrid approach outperforms both the decision tree and case-based approaches as well as two case-based systems that incorporate expert knowledge into their case retrieval algorithms. Results clearly indicate that decision trees can be used to improve the performance of CBL systems and do so without reliance on potentially expensive expert knowledge.

1 INTRODUCTION

The ability or inability of a natural language processing (NLP) system to handle gaps in lexicon coverage ultimately affects the system's performance on novel texts. Suppose, for example, that a natural language system processes a text with the goal of summarizing it or extracting relevant information, but unexpectedly encounters an unknown word. Rather than stop and wait for a knowledge engineer to enter the missing lexical information, or skip the offending word altogether, a robust sentence analyzer should infer the necessary syntactic and semantic knowledge for the unknown word and then continue processing the text. Consider the following sentence for which an NLP system finds no entry in its lexicon for "Malaysia:"[1]

Sanyo Electric Co. and Ford Motor Co. have agreed to set up a joint venture by the end of this year to produce car audio parts in **Malaysia**, they said Thursday.

Before the NLP system can continue beyond "Malaysia," it may need to know a specific set of features for the unknown word including its

- part of speech (e.g., noun),
- general semantic class (e.g., location),
- specific semantic class (e.g., country),
- associated concepts (e.g., "Malaysia" may activate a company-location concept in this context),
- relationship to other entities in the sentence (e.g., the joint venture company will be located in Malaysia),
- relationship to entities in a database (e.g., Ford may be a party to another joint venture in Malaysia), etc.

Although the exact types of knowledge required vary tremendously from system to system, all NLP systems are faced with the problem of inferring a number of predetermined features for each unknown word encountered in a text.

When viewed as a problem in machine learning, this lexical acquisition task does not fit neatly into existing paradigms. Because it requires classification

[1]This sentence was taken from the TIPSTER joint ventures corpus.

along multiple, sometimes related, dimensions, lexical acquisition isn't simply a *classification* problem of the type typically handled by decision tree algorithms (Quinlan 1986). However, because the features to be predicted are known beforehand, neither is it a pure example of the *flexible prediction* task (e.g., Fisher 1989; Fisher 1987) performed by conceptual clustering algorithms (Fisher 1987; Michalski & Stepp 1983). Instead, the lexical acquisition problem seems to fall naturally somewhere between the two, in a paradigm we will call *semi-flexible prediction.* In semi-flexible prediction tasks, **learning should improve prediction of a set of features known a priori** rather than a single feature (as in classification) or an arbitrary set of features (as in flexible prediction). This paper describes a hybrid learning technique for semi-flexible prediction tasks that combines case-based learning (CBL)[2] and decision trees: for each feature to be predicted, we rely on a decision tree algorithm to choose the attributes to be included in a simple k-nearest neighbor case retrieval mechanism. We evaluate the approach on the lexical acquisition task described above and show that the hybrid learning algorithm outperforms a pure decision tree solution, a k-nearest neighbors CBL algorithm, and two CBL algorithms that ostensibly encode expert knowledge in their similarity functions. Given the results of our experiments, we conclude that a combination of case-based learning and decision tree algorithms may offer a solution for semi-flexible, knowledge-based prediction. In addition, we believe that the hybrid technique offers an automated alternative to the usually time- and knowledge-intensive design of usable similarity functions for case-based reasoning systems.

In the next section, we first outline a simplified version of the lexical acquisition problem that will be used as the semi-flexible prediction task in all experiments. We then briefly describe the instance representation used across all solutions. Section 3 compares solutions to the problem using a decision tree algorithm, three case-based learning variations, and the hybrid CBL-decision tree algorithm. We conclude with a discussion of related work the contributions of this research (section 4).

2 LEARNING THE DEFINITION OF UNKNOWN WORDS

In the experiments of section 3, we use a semi-flexible prediction task that is a simplification of the the lexical acquisition task described above:

> *Given the context in which an unknown word occurs, learn just three features of the unknown word — the word's*
> 1. *part of speech,*
> 2. *general semantic class, and*
> 3. *specific semantic class.*

In addition, we focus only on learning the definitions of open class words and assume that information for all closed class words is known. Closed class words are function words like prepositions, auxiliaries, articles, and connectives, whose meanings vary little from one domain to another. All other words (e.g., nouns, verbs, adjectives) are open class words. Focusing on open class words is a legitimate simplification of the original problem because it is likely that the lexicon employed by any natural language processing system will contain entries for all closed class words.[3]

In addition, all training and test instances are lists of attribute-value pairs and are derived from sentences in the TIPSTER JV corpus. This corpus currently contains over 1300 texts that recount world-wide activity in the area of business joint ventures. The next section describes the instance representation used for this language learning problem.

2.1 THE INSTANCE REPRESENTATION

Each training instance is a list of 38 attribute-value pairs and represents the definition of a single open class word as well as the context in which it occurs. Figure 1 shows the training instance for the word "venture" in a sentence taken directly from the TIPSTER JV corpus. Features are divided into three groups: word definition features, local context features, and global context features. First, there are 5 *word definition features* that encode information about the unknown word: the word itself, its part of speech, general and specific semantic attributes, and morphology. Values for the part of speech (**p-o-s**), general attribute (**gen-att**), and specific attribute (**spec-att**) are taken from taxonomies developed for use with the corpus and contain 18, 17, and 45 entries, respectively. "Venture," for example, is a *noun modifier (nm)*[4] and has been assigned the most general semantic attribute, *entity,* but no specific semantic attribute. It has no associated morphological information.

Next, we represent the context via 20 local context features and 13 global context features. The *local context features* describe semantic and syntactic knowledge for the two words preceding (**prev1** and **prev2**) and the

[2]The term "case-based learning" is essentially equivalent to "instance-based learning" (Aha, Kibler, & Albert 1991; Aha 1989), but the former term is preferred here because it implies the possibility of a case adaptation phase.

[3]The UMass NLP systems that process texts in the domains of Latin American terrorism, business joint ventures, and microelectronics, for example, rely on the same set of approximately 130 closed class words.

[4]The *noun modifier(nm)* category covers both adjectives and nouns that act as modifiers. We reserve the *noun* category for head nouns only.

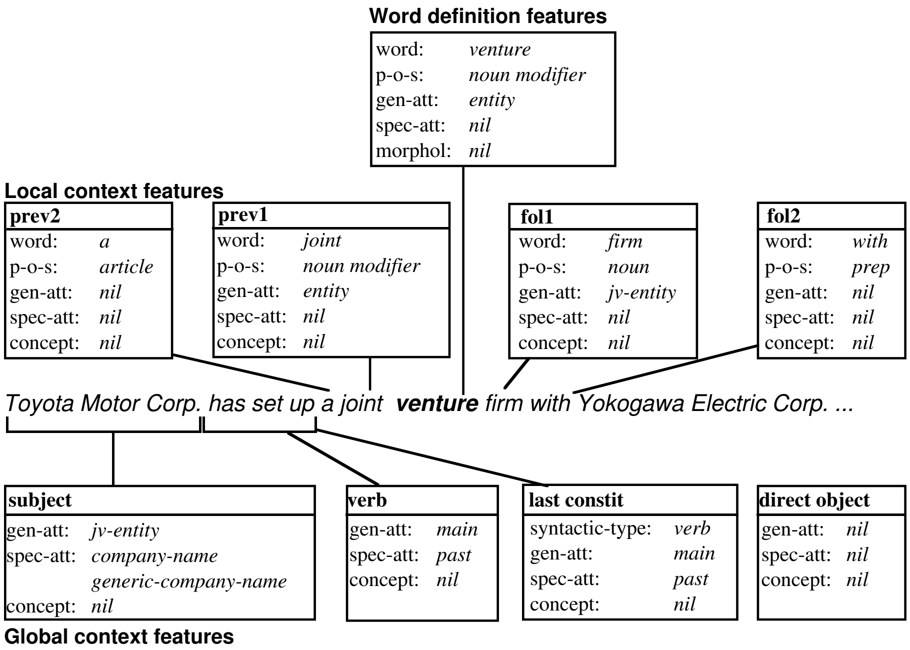

Figure 1: Case for "venture"

two words following (**fol1** and **fol2**) the current word. Again, we draw from the taxonomies to describe each word's part of speech, and general and specific attributes. We also include a feature that indicates the domain-specific **concept** activated by each word in the current context. Most knowledge-based sentence analyzers rely on such domain-specific concept activation to indicate when important information has been encountered. In the terrorism domain, for example, "killed" should activate a terrorist-murder concept in the sentence, "Terrorists killed President Gorka," but not in the sentence, "The cancer had effectively been killed." There are currently 11 possible values for concept attributes. In Figure 1, none of the words in the local context of "venture" activates a domain-specific concept.

Finally, the *global context features* represent the state of the parser at the unknown word and include semantic information for each major syntactic constituent (i.e., subject, verb, direct object) and for the most recent low-level constituent, **last constit**. **Last constit** refers to the most recent noun phrase, prepositional phrase, or verb phrase, and often overlaps with one of the major syntactic constituents. Note that all features for the direct object are empty because that constituent has not yet been recognized at the point of the unknown word.

The intent of this representation of global and local context is to include an attribute-value pair for essentially every piece of knowledge that the parser might access to determine the definition of an unknown word when one is encountered as it analyzes a text. In particular, the representation is based on the kinds of knowledge available to the CIRCUS conceptual sentence analyzer (Lehnert 1990) that was used to process the TIPSTER JV corpus. A more detailed description of the instance representation, the taxonomies, and the semi-automated method used to generate the training instances is described in (Cardie 1993).[5]

Every training instance in the experiments below is based on the the 38-attribute feature set described here. In all test instances, however, we omit the **p-o-s**, **gen-att**, and **spec-att** word definition features we are trying to predict. These features represent the part of speech and semantic classes of the unknown word and will be inferred by the learning algorithm. In the next section, we present experiments using decision tree, CBL, and hybrid solutions to the unknown word problem and compare them to each other and to additional performance baselines.

[5] In that paper we learn 4 features for each unknown word instead of 3 and focus on the task from a natural language processing perspective. As a result, there are minor differences in the instance representations described in each paper. Verbs, for example, take on semantic features in the representation used here, but do not in (Cardie 1993).

3 COMPARING THE DECISION TREE, CBL, AND HYBRID APPROACHES

In each of the following experiments, we draw the training and test instances from a base set of 2056 38-attribute instances, one for each occurrence of an open class word in 120 sentences of the TIPSTER JV corpus. In addition, all experiments use a 10-fold cross validation evaluation scheme in which we randomly choose a different, non-overlapping set of 205 test cases from this base set and use the remaining instances for training in each of 10 runs. We emphasize that the same 10 training and test set combinations were used in the 10-fold cross validation of each experiment below.

3.1 DECISION TREE APPROACH

The decision tree approach to the semi-flexible prediction problem described in section 2 consists of generating 3 decision trees, one for each class of knowledge to be learned for an unknown word — its part of speech (**p-o-s**), general semantic attribute (**gen-att**), and specific semantic attribute (**spec-att**). We will refer to these as the *missing features* of the unknown word. To generate the decision tree for feature x, we present the training instances to the C4.5 decision tree system (Quinlan 1992) after removing the 3 missing features and augmenting the training instance with the value for x as its supervisory class information. The missing features were also removed from the test instances.

Table 1 shows the average performance of C4.5 in predicting each of the missing features across 10 runs and compares it to two baselines.[6] The first baseline indicates the expected accuracy of a system that randomly guesses a legal value for each missing feature based on the distribution of values across the test set. The second baseline shows the performance of a system that always chooses the most frequent value as a default. Chi-square significance tests on the associated frequencies show that the decision tree approach performs significantly better than both of the baselines ($p = .01$).

3.2 CASE-BASED APPROACH

In CBL, the case base is effectively a set of training examples, each of which describes a single problem-solving episode. After training, when a new problem arises, a case retrieval algorithm compares the new problem to those stored in the case base, finds the most similar training case, and then uses it to solve the current problem. In the case-based solution to the NLP problem described above, we create a flat case base of training instances, each of which contains all 38 attribute-value pairs. Then, given a test case from which the 3 missing features have been removed, the case retrieval algorithm searches the case base, finds the training cases that best match it, and then uses them to predict values for the missing features of the unknown word. We use the following case retrieval algorithm for this task:

Table 1: Results for the Decision Tree Approach (% correct)

Missing Feature	Decision Tree	Random Selection	Default
p-o-s	89.0	34.3	81.5
gen-att	66.0	15.9	25.6
spec-att	69.9	24.7	45.3

1. Compare the test case to each case in the case base, counting the number of features that match (i.e., match = 1, mismatch = 0). Do not include the missing features in the comparison. Only give partial credit (.5) for matches on *nil*'s.[7]
2. Keep the k highest-scoring cases.
3. Of these, return the case(s) whose **word** matches the unknown word, if any exist (i.e., prefer instances of the unknown word seen during training). Otherwise, return all k cases.[8]
4. Let the retrieved cases vote on the values for the missing features.

The case retrieval algorithm is essentially a k-nearest neighbors (k-*nn*) matching algorithm with a bias toward examples of the unknown word encountered during training. Table 2 shows the averaged results of the case-based runs (for k= 1, 5, 10) and compares them to the decision tree results. Significant differences in performance with respect to the decision tree approach are indicated in the table by $*$'s. Generally, the decision tree performs better than the CBL approach for k=1, worse than the CBL approach for k=10, and is indistinguishable from the case-based approach for k=5.

The problem with the CBL solution as presented is the difficulty of defining a similarity function for case retrieval that can be used to accurately predict all of the missing features of the unknown word. The k-nn routine in the case retrieval algorithm described above (step1) assumes that all features are equally important for predicting each of the missing features. But intuitively it seems that accurate prediction of each class

[6] In all experiments described in this paper, we allow mismatches between the noun and noun modifier parts of speech because the parser can fix these errors.

[7] This is because a *nil* value indicates that an attribute did not apply in the current context and the matching process should focus on relevant features rather than omitted features.

[8] More than k cases will be returned if there are ties.

Table 2: Results for the Baseline Case-Based Approach (% correct). (* and ** indicate significance with respect to the decision tree results, $* \rightarrow p = .01$ and $** \rightarrow p = .05$.)

Missing Feature	Case-Based (k = 1)	Case-Based (k = 5)	Case-Based (k = 10)	Decision Tree
p-o-s	86.6*	88.8	89.4	89.0
gen-att	58.5*	66.2	69.1*	66.0
spec-att	62.9*	70.4	72.2**	69.9

of missing information for the unknown word may actually rely on very different subsets of the feature set. In fact, it is well known that k-nn algorithms perform poorly in the presence of irrelevant features (Aha, Kibler, & Albert 1991; Aha 1989).

One method for optimizing the similarity metric for each missing feature is to employ expert knowledge. We can incorporate informed intuitions about the nature of each class of missing knowledge into the case retrieval algorithm by letting an expert decide which features to include in the k-nn calculations. Some successful part of speech taggers, for example, make decisions based only on knowledge of the words in a window to either side of the unknown word. This implies that the k-nn routine should only include the local context features in its calculations. Adding the global context features may only hurt performance. On the other hand, the semantic features of an unknown word seem to depend partially on local context and partially on knowledge about the global state of the sentence. For example, the semantic class of a noun that follows a verb may depend on the semantic class of the clause's subject. Therefore, when predicting semantic features, it might be better first to find the most similar cases using the local context features and then choose from these the cases that match best along both the local and global context dimensions.

We incorporated these observations into two variations of the baseline CBL system. The first variation, referred to as the "p-o-s" CBL system, was designed to improve **p-o-s** prediction and uses only the local context features in its k-nn comparisons. The second variation, referred to as the "semantic class" CBL system, was designed to improve prediction of the **gen-att** and **spec-att** features. It submits those cases initially selected using just local context features to an additional k-nn filter that includes the global context features as well. Like the baseline CBL system, both intuitive variations also prefer cases whose **word** feature matches the unknown word.

Table 3 shows the results of the intuitive CBL variations and compares them to the baseline CBL algorithm and the decision tree results. Only the results for k=10 are shown, but runs using k=1, 5 exhibited similar behavior. All results are averaged over 10 runs. Also shown in the table are annotations for statistical significance. (*'s indicate the performance of all case-based systems as compared to the decision tree results and ◇'s indicate performance of the intuitive CBL variations as compared to the CBL baseline.) As expected, focusing on local features improved part of speech prediction and the semantic class CBL variation improved performance across the general and specific semantic attributes. However, the p-o-s CBL method also unexpectedly improved the prediction of both semantic class attributes.

3.3 HYBRID APPROACH

The preceding experiments show that it is possible to use informed intuitions to discard irrelevant attributes from the feature set and thus improve performance of the k-nn case retrieval algorithm. Given that feature set specification is a notoriously time-consuming and knowledge-intensive task however (Quinlan 1983), it would be better if the feature set could be chosen systematically and automatically. This problem is addressed in the hybrid approach to semi-flexible prediction in which decision trees aid in the definition of a similarity metric that focuses on an appropriate subset of features by isolating the attributes most important for accurate prediction of each class of missing knowledge. In the hybrid approach, we let C4.5 select the features to be included for k-nn case retrieval:

1. *For each training set used in the decision tree experiments (section 3.1), note the features that occurred in the corresponding C4.5 decision tree.*[9] This essentially produces, for each of the missing attributes, a list of all features that C4.5 found useful for predicting its value.
2. *Run the baseline case retrieval algorithm (section 3.2) with the following modification: instead of invoking the case retrieval algorithm once for each test case, run it three times, once for each missing attribute to be predicted.* In the retrieval for attribute a, however, include only the features C4.5 found to be important for predicting a in the k-nn calculations.[10]

[9] We use the pruned decision trees produced by C4.5 for this experiment as well as for the original decision tree experiments. Note also that as part of the 10-fold cross validation scheme, we created 10 decision trees for each missing feature — one for each training set. We use these same decision trees for the current experiment.

[10] We actually only compare each test case to the entire case base once (not three times) and use the results of that

Table 3: Results for Intuitive CBL Variations (% correct, $k = 10$). (* and ** indicate significance with respect to the decision tree results, $* \rightarrow p = .01$ and $** \rightarrow p = .05$. $\diamond$ and $\diamond\diamond$ indicate significance with respect to the baseline CBL system, $\diamond \rightarrow p = .01$ and $\diamond\diamond \rightarrow p = .05$.)

Missing Feature	Case-Based (p-o-s)	Case-Based (semantic class)	Case-Based (baseline)	Decision Tree
p-o-s	91.4*◇	90.4**	89.4	89.0
gen-att	73.9*◇	72.1*◇	69.1*	66.0
spec-att	75.0*◇	74.2*◇◇	72.2**	69.9

When predicting the part of speech of the unknown word, for example, only those features C4.5 found to be important for **p-o-s** prediction are included in the k-nn matching routine (step 1 of the case retrieval algorithm). In contrast to the expert knowledge required to devise the intuitive CBL approaches, case retrieval is automatically tuned in the hybrid system by using C4.5 for feature specification. The feature sets proposed by C4.5 reduce the number of attributes used in the case retrieval algorithm from 35 to an average of 14 (**p-o-s**), 11 (**gen-att**), and 15 (**spec-att**) features.[11]

Table 4 shows the average performance of the hybrid approach across 10 runs and compares it to identical runs for the baseline CBL system and the best of the intuitive T approaches, i.e., the approach that relied only on local context features. Again, only results for k=10 are shown although results for k=1, 5 were much the same. The table also compares the results to a system that randomly chooses the features to be used in the k-nn calculations while controlling for feature set size (i.e., we use the same number of features that were used in the corresponding run for the hybrid approach). In all but one case, the hybrid approach significantly outperforms the other approaches ($p = .05$). The only exception was prediction of the **gen-att** feature, for which the p-o-s CBL system did as well as the hybrid approach. As noted above, however, the p-o-s CBL system that focused on local context features was designed to improve prediction of part of speech, not general semantic class.

In spite of the promising performance demonstrated by the hybrid learning system, there are problems with our current approach. The speed of the algorithm degrades linearly with the size of the case base and modifications would be required before the approach could be tested using a hierarchical case base. Methods described in (Aha, Kibler, & Albert 1991) to reduce the storage requirements of T algorithms provide an alternative to construction of a hierarchical case base, however. In addition, it is not feasible to tune the case retrieval mechanism (i.e., to determine the relevant attributes associated with each missing feature) after every incoming instance because the costs associated with running a decision tree algorithm are too great. Instead, one might wait until the case base was relatively stable before employing the hybrid CBL approach or tune the similarity metrics only occasionally. In both solutions, however, we lose some of the inherent advantages associated with the incremental nature of CBL algorithms. Finally, we should test the approach on additional data sets, or find a method for automatically recognizing problems that will respond favorably to this hybrid technique.

comparison for each of the three k-nn calculations.

[11]These are averages across the 10 experiments run for each missing feature as part of the 10-fold cross validation evaluation.

4 RELATED WORK AND CONCLUSIONS

4.1 RELATED WORK IN LEXICAL ACQUISITION

Although the problem of automating lexical acquisition has been addressed before, previous approaches often focus on learning either syntactic or limited semantic knowledge but not both (e.g., (Brent 1990; Grefenstette 1992; Resnik 1992; and Zernik 1991)). Moreover, the approaches tend to fall into one of two categories: statistically-based methods that acquire (usually syntactic) lexical knowledge (e.g., (Brent 1991; Church & Hanks 1990; Hindle 1990; Resnik 1992; Yarowsky 1992; and Zernik 1991)), or knowledge-intensive methods that acquire syntactic and/or semantic lexical knowledge, but rely heavily on hand-coded world knowledge (e.g., (Berwick 1983; Granger 1977; Hastings et al. 1991; Lytinen & Roberts 1989; and Selfridge 1986)) or hand-coded heuristics that describe how and when to acquire new word definitions (e.g., Jacobs & Zernik 1988 and Wilensky 1991). Our approach differs from all of these in that

- it uses a novel combination of two existing machine learning paradigms
- the same learning algorithm and instance representation are used to simultaneously learn both syntactic and semantic lexical knowledge
- the approach does not rely on hand-coded heuristics, and
- relatively little training is needed.[12]

[12]For a more detailed description of this work from an NLP perspective, see (Cardie, 1993). In that paper, we

Table 4: Results for Hybrid Approach (% correct). (ˆ indicates results not significantly different than the hybrid system. The hybrid system significantly outperforms all other variations, $p = .05$.)

Missing Feature	Hybrid (DT + CBL) $k = 10$	Case-Based (baseline) $k = 10$	Case-Based (p-o-s) $k = 10$	Random Features $k = 10$	Decision Tree
p-o-s	92.5	89.4	91.4	89.7	89.0
gen-att	73.4	69.1	73.9ˆ	62.9	66.0
spec-att	76.7	72.2	75.0	71.1	69.9

4.2 ADDITIONAL RELATED WORK AND CONCLUSIONS

In this paper, we have compared three approaches to problems in semi-flexible prediction: decision trees, case-based learning, and a hybrid technique that combines the two. In the hybrid approach, decision trees specify the features to be included in k-nearest neighbor case retrieval. In related work, (Aha 1989) presents a method for learning concept-dependent attribute relevancies in a case-based paradigm. He dynamically updates the similarity function for each concept by modifying an attribute weight vector associated with the concept in response to classification performance. Here we use decision trees essentially to create an attribute weight vector for each concept where the weights are either 0 or 1. However, one possibility which we have not yet explored is to use the position of an attribute in the decision tree to derive attribute weights between 0 and 1. This would make our weight vector more similar to Aha's real-valued weights that range between 0 and 0.5. In addition, Aha's method differs from ours in that (1) it is completely incremental, i.e., the similarity function for each concept must be updated for every incoming instance, and (2) it is designed for boolean-valued concepts rather than the multi-valued concepts used here. Although, in theory, the incremental method seems ultimately more appropriate, it may not be feasible when the number of concepts to be learned is large and/or there are multi-valued concepts involved.

Given the results of the experiments outlined in section 3 that compare the decision tree, cased-based, and hybrid approaches to semi-flexible prediction, we conclude that the hybrid technique performs significantly better than the pure decision tree and CBL algorithms for a language learning task. It also performed better than two CBL systems that incorporated expert knowledge for the feature specification task. This result has important implications for work in case-based paradigms because it clearly indicates that decision tree algorithms can be used to improve the performance of some CBL systems without reliance on potentially expensive expert knowledge. On one hand, these results may not seem surprising since previous research has found the converse to be true — (Skalak & Rissland 1990) show that a case-based reasoning system can successfully perform the feature specification task for a decision tree classification system. However, (Almuallim & T 1991) show that ID3 (Quinlan 1986) is not particularly good at selecting a minimum set of features from an original set containing possibly many irrelevant attributes. While their results may hold in general, we claim that there is at least one important class of problem for which decision tree algorithms can perform feature specification reasonably well.

incorporate the hybrid learning algorithm described here into a working sentence analyzer that processes text from a variety of corpora. We then evaluate the approach for learning the definitions of unknown words in two practical language processing applications.

Acknowledgments

Many thanks to Professor J. Ross Quinlan for supplying the C4.5 decision tree system. Thanks also to Carla Brodley, Ellen Riloff, Wendy Lehnert, and David Skalak for helpful comments and discussions. This research was supported by the Office of Naval Research Contract N00014-92-J-1427 and NSF Grant no. EEC-9209623, State/Industry/University Cooperative Research on Intelligent Information Retrieval.

References

Aha, D., Kibler, D., & Albert, M. (1991). Instance-Based Learning Algorithms. *Machine Learning 6* (1): pp. 37-66.

Aha, D. (1989). Incremental, Instance-Based Learning of Independent and Graded Concept Descriptions. *Proceedings, Sixth International Workshop on Machine Learning*, pp. 387-391. Cornell University, Ithaca, NY. Morgan Kaufmann.

Almuallim, H., & Dietterich, T. G. (1991). Learning With Many Irrelevant Features. *Proceedings, Ninth National Conference on Artificial Intelligence*, pp. 547-552. Anaheim, CA. AAAI Press / The MIT Press.

Berwick, R. (1983). Learning word meanings from examples. *Proceedings, Eighth International Joint Conference on Artificial Intelligence*, pp. 459-461. Karlsruhe, Germany.

Brent, M. (1991). Automatic acquisition of subcategorization frames from untagged text. *Proceedings, 29th Annual Meeting of the Association for Computational Linguistics*, pp. 209-214. University of California, Berkeley. Association for Computational Linguistics.

Brent, M. (1990). Semantic classification of verbs from their syntactic contexts: automated lexicography with implications for child language acquisition. *Proceedings, Twelfth Annual Conference of the Cognitive Science Society*, pp. 428-437. Cambridge MA. The Cognitive Science Society.

Cardie, C. (1993). A Case-Based Approach to Knowledge Acquisition for Domain-Specific Sentence Analysis. To appear in *Proceedings, Eleventh National Conference on Artificial Intelligence.* Washington, DC.

Church, K., & Hanks, P. (1990). Word association norms, mutual information, and lexicography. *Computational Linguistics, 16.*

Fisher, D. (1989). Noise-Tolerant Conceptual Clustering. *Proceedings, Eleventh International Joint Conference on Artificial Intelligence*, pp. 630-635. Detroit MI. Morgan Kaufmann.

Fisher, D. H. (1987). Knowledge Acquisition Via Incremental Conceptual Clustering. *Machine Learning 2*: 139-172.

Granger, R. (1977). Foulup: A program that figures out meanings of words from context. *Proceedings, Fifth International Joint Conference on Artificial Intelligence*, pp. 172-178. Morgan Kaufmann.

Grefenstette, G. (1992). SEXTANT: Exploring unexplored contexts for semantic extraction from syntactic analysis. *Proceedings, 30th Annual Meeting of the Association for Computational Linguistics*, pp. 324-326. University of Delaware, Newark, DE. Association for Computational Linguistics.

Hastings, P., Lytinen, S., & Lindsay, R. (1991). Learning Words from Context. *Proceedings, Eighth International Conference on Machine Learning*, pp. 55-59. Northwestern University, Chicago, IL. Morgan Kaufmann.

Hindle, D. (1990). Noun classification from predicate-argument structures. *Proceedings, 28th Annual Meeting of the Association for Computational Linguistics*, pp. 268-275. University of Pittsburgh. Association for Computational Linguistics.

Jacobs, P., & Zernik, U. (1988). Acquiring Lexical Knowledge from Text: A Case Study. *Proceedings, Seventh National Conference on Artificial Intelligence*, pp. 739-744. St. Paul, MN. Morgan Kaufmann.

Lehnert, W. (1990). Symbolic/Subsymbolic Sentence Analysis: Exploiting the Best of Two Worlds. In J. Barnden, & J. Pollack (Eds.), *Advances in Connectionist and Neural Computation Theory*, pp. 135-164. Norwood, NJ: Ablex Publishers.

Lytinen, S., & Roberts, S. (1989). Lexical Acquisition as a By-Product of Natural Language Processing. *Proceedings, IJCAI-89 Workshop on Lexical Acquisition.*

Michalski, R. S., & Stepp, R. (1983). Learning from observation: conceptual clustering. In R. S. Michalski, J. G. Carbonell, & T. M. Mitchell (Eds.), *Machine Learning: An Artificial Intelligence Approach.* Morgan Kaufmann.

Quinlan, J. R. (1992). *C4.5: Programs for Machine Learning.* Morgan Kaufmann.

Quinlan, J. R. (1986). Induction of decision trees. *Machine Learning, 1*: 81-106.

Quinlan, J. R. (1983). Learning Efficient Classification Procedures and Their Application to Chess End Games. In R. S. Michalski, J. G. Carbonell, & T. M. Mitchell (Eds.), *Machine Learning: An Artificial Intelligence Approach.* Palo Alto: Morgan Kaufmann.

Resnik, P. (1992). A class-based approach to lexical discovery. *Proceedings, 30th Annual Meeting of the Association for Computational Linguistics*, pp. 327-329. University of Delaware, Newark, DE. Association for Computational Linguistics.

Selfridge, M. (1986). A computer model of child language learning. *Artificial Intelligence, 29*: 171-216.

Skalak, D. and Rissland, E. (1990). Inductive Learning in a Mixed Paradigm Setting. *Proceedings, Eighth National Conference on Artificial Intelligence*, pp. 840-847. Boston, MA. AAAI Press/MIT Press.

Wilensky, R. (1991). Extending the Lexicon by Exploiting Subregularities. *Tech. Report No. UCB/CSD 91/618.* Computer Science Division (EECS), Univ. of California, Berkeley.

Yarowsky, D. (1992). Word-Sense Disambiguation Using Statistical Models of Roget's Categories Trained on Large Corpora. *Proceedings, COLING-92.*

Zernik, U. (1991). Train1 vs. Train 2: Tagging Word Senses in Corpus. In U. Zernik (Ed.), *Lexical Acquisition: Exploiting On-Line Resources to Build a Lexicon*, pp. 91-112. Hillsdale, NJ: Lawrence Erlbaum Associates, Inc.