Automatically Learned vs. Hand-crafted Text Analysis Rules*

Stephen Soderland
Dept. Computer Science & Engineering
University of Washington
Seattle, WA 98195-2350
soderlan@cs.washington.edu

David Fisher, Wendy Lehnert
Dept. Computer Science
University of Massachusetts
Amherst, MA 01003-4610
{dfisher lehnert}@cs.umass.edu

Abstract

As vast quantities of on-line text become available, there is an increasing need for systems that automatically analyze the conceptual content of natural language text. Systems that operate on narrowly defined domains show promise, but require a different set of domain-specific rules for each application.

This paper describes CRYSTAL, a system that learns text analysis rules automatically from examples. Rules induced by CRYSTAL achieve performance approaching that of hand-crafted rules. CRYSTAL has a particularly efficient learning algorithm that is not improved by more extensive search. This offers a practical alternative to time-consuming manual knowledge engineering for each new domain.

1 Domain-specific Text Analysis

With the increasing amounts of on-line text available, the need is growing for automated text analysis systems that go beyond keywords to extract the conceptual content of the text. This requires a system that can reliably extract both explicitly stated information and information that can be reasonably inferred. General purpose text understanding is still beyond the reach of current technology, but considerable progress has been made by restricting the problem to a predefined set of concepts in a narrowly defined domain.

A text analysis system with the appropriate domain-specific knowledge sources can identify references to information that is of interest to a particular *domain*, which consists of a corpus of texts together with a set of *concepts* to be identified in those texts.

The target concepts in a medical domain might be references to symptoms and diagnoses in patient records. In a collection of Wall Street Journal articles, the target concept might be management succession events: persons moving into top management positions in corporations and persons moving out of those positions. The ARPA-sponsored Sixth Message Understanding Conference [MUC-6 1995] used such a "Management Succession" domain. This domain is illustrated by Figure 1.

Input Text:

Who's News: Topologix Inc.

Donald E. Martella, formerly vice president, operations, was named president and chief executive officer of this maker of parallel processing subsystems. He succeeds Jack Harper, a company founder who was named chairman. ...

Succession_Event:
Person_In: Donald E. Martella
Person_Out: Jack Harper
Position: president and chief executive officer
Organization: Topologix Inc.

Succession_Event:
Person_Out: Donald E. Martella
Position: vice president, operations
Organization: Topologix Inc.

Succession_Event:
Person_In: Jack Harper
Position: chairman
Organization: Topologix Inc.

Figure 1: Output from a "Management Succession" text

This text has three succession events. Donald Martella is moving into a position that Jack Harper is leaving; Martella is moving out of his old job as vice president; Harper is moving in as chairman. These succession events can be represented as three case frames, each case frame having up to four slots: *Person_In*, *Person_Out*, *Position*, and *Organization*.[1]

*This material is based on work supported in part by the National Science Foundation, Library of Congress and Department of Commerce under cooperative agreement number EEC-9209623 and in part by NRaD Contract Number N66001-94-D-6054.

[1] We are using a somewhat simpler output representation than that used in MUC-6, for the sake of clarity.

How can an automated system start from the raw text in Figure 1 and produce the desired output representation? A key knowledge source is a set of text analysis rules that identify references to management succession events, based on local linguistic context. These rules will be specific to the way such events are typically described and are sensitive to the vocabulary, word senses, and writing style of the domain.

Rules that apply to the text in Figure 1 might look for patterns such as the following.

1. "<Person> WAS NAMED <Corporate Post> OF <Organization>"
2. "<Person> SUCCEEDS <Person>"
3. "<Person> FORMERLY <Corporate Post>"
4. "<Person> WHO WAS NAMED <Corporate Post>"

Another domain-specific knowledge source needed is a semantic lexicon used to tag individual words with semantic classes appropriate to the domain. Semantic tagging of individual words enables rules of greater generality than rules based exclusively on exact words.

Rules based on patterns such as these are highly specific to a particular domain and can be difficult and time-consuming to write by hand. Writing such rules manually requires both domain expertise and a detailed knowledge of the text analysis system.

An attractive alternative is to use machine learning to acquire the necessary rules. This paper describes CRYSTAL, a system that learns domain-specific text analysis rules from training examples. An earlier implementation of CRYSTAL was presented in [Soderland *et al.* 1995]. A fuller treatment may be found in [Soderland 1996].

With CRYSTAL, a domain expert's responsibility is to define the target concepts for a domain and to create training data by marking each reference to the target concept in a set of representative texts. This does not require any background in linguistics or computer science. CRYSTAL automatically derives a set of rules that, in effect, imitate the domain expert's annotations on previously unseen texts.

2 Concept Definitions

The rules that CRYSTAL learns, called *concept definitions*, apply a combination of lexical, semantic, and syntactic constraints on an input instance. If all the constraints are satisfied, a case frame is created with the extracted information.

Before concept definitions are applied, a sentence analyzer identifies major syntactic constituents such as subject, verb, direct object, and prepositional phrases. We used the BADGER sentence analyzer[2] [Fisher *et al.* 1995] for experiments reported here. Semantic tagging for the Management Succession domain was based on a semantic lexicon that was tailored to the domain and was supplemented by a module that recognizes person names and company names.

```
Concept type: Succession Event
Constraints:
   SUBJ::
      Classes include:  <Person>
      Extract:          Person_In
   VERB::
      Terms include:    NAMED
      Mode:             passive
   OBJ::
      Terms include:    OF
      Classes include:  <Corporate Post>,
                        <Organization>
      Extract:          Position, Organization
```

Figure 2: A concept definition that applies to "Donald E. Martella ... was named president ... of this maker of parallel processing subsystems."

Figure 2 shows a concept definition that applies to the first sentence in Figure 1. This concept definition has a constraint that requires the semantic class <Person> in the subject, which is satisfied since BADGER's name recognizer has labeled "Donald E. Martella" as the semantic class <Person Name>, a subclass of <Person>.

This concept definition also has constraints requiring the verb phrase to include the word "named" in the passive voice. Constraints on the direct object require the semantic class <Corporate Post>, the word "of", and the class <Organization>. Each of these constraints is met by the direct object "president and chief executive officer of this maker of parallel processing subsystems". Both "president" and "chief executive officer" are tagged as <Corporate Post> in this domain. The word "maker" has the semantic class <Generic Organization>, which is an <Organization>.

Since all these constraints are satisfied, CRYSTAL creates a case frame with the *Person_In* slot filled by the subject and the *Position* slot and *Organization* slot filled by the direct object. Later processing in a full text analysis system is needed to trim away extraneous words from the extracted slot fills and to replace generic references such as "maker of parallel processing subsystems" with an actual company name.

Figure 3 shows another concept definition. This one applies to the second sentence in Figure 1, "He succeeds Jack Harper ..." This extracts a *Person_In* from the subject and a *Person_Out* from the direct object. Later processing will replace "he" with the actual name and merge this output with the case frame from the definition in Figure 2.

As the preceding examples illustrate, a concept definition applies constraints to syntactic constituents of an instance (e.g. to the subject, verb, direct object, or

[2]The BADGER and CRYSTAL software were provided by the Natural Language Processing Laboratory, University of Massachusetts Computer Science Department, Amherst, Massachusetts.

```
Concept type: Succession Event
Constraints:
   SUBJ::
      Classes include:   <Person>
      Extract:           Person_In
   VERB::
      Root:              SUCCEED
      Mode:              active
   OBJ::
      Classes include:   <Person>
      Extract:           Person_Out
```

Figure 3: A concept definition that applies to "He succeeds Jack Harper, a company founder ..."

prepositional phrases). CRYSTAL does not depend on a particular syntactic analysis and uses whatever syntactic labels are found in the training instances. The only requirement is that the instances are presented as a flat list of syntactic constituents with no embedded constituents.

The terms constraint is an unordered list of words that must be included in the syntactic constituent. The classes constraint is an unordered list of semantic classes that must be present, either directly or through an IS-A relationship. Lexical and semantic constraints may also make a distinction between head terms or classes and modifier terms or classes. Terms found as the last term of the phrase, or just before punctuation, before a preposition, or before an adverb are considered to be head terms. All others are considered modifiers. The root constraint is used if the sentence analyzer provides morphological analysis (i.e. verb roots). The mode constraint is used if the sentence analyzer labels phrases as affirmative/negative or as active/passive.

```
Constraints on syntactic constituents:
    Terms
    Head terms
    Modifier terms
    Classes
    Head classes
    Modifier classes
    Root
    Preposition
    Mode (affirmative/negative, active/passive)
```

Figure 4: Constraints in a concept definition

3 The CRYSTAL Algorithm

CRYSTAL is a supervised learning algorithm, and as such needs training instances that have been annotated by a human expert. CRYSTAL is given a training set of texts in which every instance of the concept being learned (e.g. management succession event) has been explicitly marked in the text. Any phrases not marked as positive instances of the target concept are considered to be negative instances.

```
The CRYSTAL Algorithm:

Rules = NULL
Derive an initial definition from each positive instance
Do for each initial definition D not covered by Rules
   Loop:
      D' = the most similar initial definition to D
      If D' = NULL, exit loop
      U = the unification of D and D'
      Test U on the training set
      If the error rate of U > error tolerance
         Exit loop
      Set D = U
   Add D to the Rules
Return the Rules
```

The goal of CRYSTAL is to find a set of concept definitions that are generalized enough to have good coverage on previously unseen texts, yet constrained tightly enough to operate reliably. CRYSTAL's approach is to begin with highly specific concept definitions and gradually relax the constraints. Each proposed generalization is tested for extraction errors on the training set, which has been hand-tagged with the desired phrases to be extracted. Generalization continues until further relaxation would lead to a definition that exceeds a user-specified error tolerance.

CRYSTAL begins by selecting a positive instance of the target concept as a seed. CRYSTAL then takes the most specific concept definition that covers this instance and generalizes it. The most general definition within error tolerance is added to the rule base and another seed is selected from positive instances not yet covered by the rule base. This is repeated until all positive instances have been covered or have been selected as seed. This machine learning methodology is called a *covering algorithm* [Michalski 1983] [Clark and Niblett 1989].

An efficient search control for generalizing concept definitions is vital for CRYSTAL because of the expressive representation of its rules. CRYSTAL is able to learn rules that retain or drop any combination of term constraints or semantic constraints on any syntactic constituent of an instance. A typical initial definition has dozens of constraints on the terms and semantic classes found in an instance, resulting in an extremely large space of possible generalizations.

Previous systems [Riloff 1993] [Kim and Moldovan 1992] [Huffman 1996] that learn text analysis rules avoid this problem by restricting the rule representation or by restricting representation of stored training instances [Krupka 1995]. In each of these systems, the text analysis rules (or stored instances) require an anchor word, typically the verb, but allow no other term constraints. Sentence elements that contain information to be extracted have semantic class constraints, but other sentence elements are ignored. This restricted representation makes the search space more manageable, but limits the rules

that can be learned.

CRYSTAL can handle an expressive rule representation because of its efficient search control. Generalization of a concept definition is guided by finding the most similar initial concept definition. CRYSTAL creates a proposed generalization by dropping constraints that are not shared by this most similar definition. This is equivalent to relaxing constraints just enough to cover the most similar positive instance, since each initial concept definition corresponds to a positive training instance.

This strategy has several beneficial results. Features that are merely accidental properties of a particular instance are dropped quickly. Features that are retained are those shared with a similar positive instance, which tend to include essential characteristics of the target concept. The intractable problem of finding an optimal generalization is thus reduced to the simpler problem of finding a similar initial concept definition.

The similarity metric used by CRYSTAL counts the number of relaxations that would be needed to unify the current concept definition with an initial definition. Dropping one word from a term constraint counts as a single relaxation, as does dropping a constraint on the verb root or the mode (affirmative/negative, active/passive). Moving up one level in the semantic hierarchy counts as one relaxation. Entirely dropping a class constraint is equivalent to the number of relaxations needed to reach the root class.

4 Empirical Results

CRYSTAL has been applied successfully to several domains. We will present results from the Management Succession domain and from a "Hospital Discharge" domain. In this second domain, the relevant information is references to symptoms and diagnoses in patients' hospital records. These are further broken down to distinguish present symptoms from absent symptoms and to distinguish confirmed diagnoses from ruled out diagnoses.

Performance is measured in terms of *recall* and *precision*. Recall is the percentage of positive instances of the target concept that were correctly identified[3]. Precision is the percentage of extractions made that were correct[4].

Recall and precision are more useful metrics than *accuracy* when there is an extremely unbalanced distribution of positive and negative instances. Suppose there are 10,000 instances of which 100 are positive and the remaining 99% are negative. A system that identifies 60 out of the 100 positive instances has recall of 60%. If the system reports an additional 20 negative instances as positive, the precision is 75% (60 right out of 80 extractions). The accuracy of this system is 99.4%, since it correctly identifies 9,880 negative instances as well as 60 true positives. Even a totally useless system that extracts nothing has accuracy of 99%.

As a point of comparison, the best system performance for participants in the ARPA-sponsored Message Understanding Conferences has been recall and precision between 50% and 60% [MUC-4 1992] [MUC-5 1993] [MUC-6 1995]. CRYSTAL, which is a component of a full information extraction system, can expect somewhat higher performance than a full system.

Figure 5 shows CRYSTAL'S performance for Management Succession as the amount of training increases[5]. A corpus of 599 annotated texts with 16,325 instances was randomly partitioned with 20%, with 40%, and with 80% of the texts as training and the remainder as a blind test set. These results are the averages of ten partitions at each training level. The number of positive training instances is shown beneath each set of recall and precision.

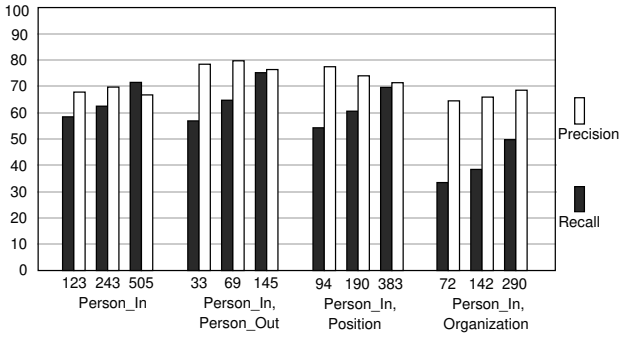

Figure 5: Learning curves for the Management Succession domain

Recall increases with each doubling of the training size with no significant difference in precision[6]. Precision remains fairly level as an artifact of the error tolerance parameter. The error tolerance was kept at 0.20, which would result in precision of about 80 if error rates on the test set exactly mirrored error rates on the training.

The error tolerance parameter can be used to manipulate a trade-off between recall and precision. Raising the error tolerance causes CRYSTAL to accept high coverage concept definitions, even if they make extraction errors. This increases recall at the expense of precision. The concept *Person_In* shown in Figure 5 trained on 40% of the corpus at error tolerance 0.20 has recall 62.6 and precision 69.8. At error tolerance 0.0, recall is 49.1 and precision 81.9. When the error tolerance is raised to 0.40,recall is 71.3 and precision 53.4.

5 Handling Extremely Noisy Input Data

Perfect recall and precision is beyond the reach of CRYSTAL, or any system that takes its input from the results

[3]Recall = TP / (TP + FN), where TP is the number of true positives, and FN is the number of false negatives (actually positive, but missed by the system).

[4]Precision = TP / (TP + FP), where TP is the number of true positives and FP is the number of false positives.

[5]CRYSTAL learned concept definitions for each combination of the four slots for succesion event case frames.

[6]Two-tailed paired t-test with $p < 0.05$

of previous automatic text processing. Noisy input is inevitable. Syntactic analysis may be inadequate or contain errors. Mistaking a verb for a noun, for example, will lead to bad clausal bracketing. Semantic tagging may fail to support distinctions needed for the domain. The wrong word sense may be assigned to a word, or none at all if it is not covered by the semantic lexicon. Human annotators also make errors in marking the training texts and produce inconsistent training data.

Even if the input contains no errors, a single instance may not contain sufficient context to distinguish the target concept. There is no way to be certain that the sentence "He succeeds Mr. Adams" is a corporate management succession. It may refer to a political appointment, which is considered irrelevant to the Management Succession domain.

Another limitation comes from the variability of free text. A finite set of training texts will never contain all possible ways to refer to the target concept. Low-frequency terms or combinations of terms that occur only in a blind test set will not be covered by the rules.

6 A Comparison with Hand-coded Rules

Experiments were conducted to see how close CRYSTAL comes to the performance of hand-coded rules. Table 1 compares CRYSTAL to hand-coded rules in the Management Succession domain. Each used the same input, a single partition with 359 texts as training and the remaining 240 texts as a blind test set. CRYSTAL achieves over 90% of the performance of hand-coded rules, with performance equal to hand-coded for one of the concepts.

Concept	CRYSTAL			Hand-coded			Ratio of Avg. R,P
	R	P	Avg	R	P	Avg	
Person_In	67.2	66.9	67.0	70.6	75.0	72.8	92.0
Person_In,Person_Out	77.6	75.4	76.5	79.1	80.3	79.7	96.0
Person_In,Position	60.0	71.5	65.8	61.7	86.0	71.8	91.6
Person_In,Organization	52.1	69.5	60.8	47.9	72.0	60.0	101.3

Table 1: Management Succession: a comparison with hand-coded rules

CRYSTAL comes close to the performance of hand-coded rules in the Hospital Discharge domain, as well. Table 2 compares the performance of CRYSTAL with hand-coded rules for this domain. 251 Hospital Discharge texts were used as training with another 251 kept as a blind test set. CRYSTAL achieves 93% the performance of hand-coded rules for three of the concepts and 88% for a fourth.

CRYSTAL can achieve nearly the performance of hand-crafted rules when the human is given exactly the same training input as CRYSTAL. A human expert brings to bear knowledge beyond that contained in the training data, but manual engineering also faces a ceiling on performance given imperfect input and limited training examples.

Concept	CRYSTAL			Hand-coded			Ratio of Avg. R,P
	R	P	Avg	R	P	Avg	
Symptom,Present	61.9	65.6	63.8	64.9	79.3	72.1	88.4
Symptom,Absent	80.0	78.9	79.4	79.6	91.9	85.8	92.6
Diagnosis,Confirmed	74.8	69.1	72.0	76.8	77.5	77.2	93.3
Diagnosis,Ruled_Out	73.5	83.1	78.3	81.2	87.2	84.2	93.0

Table 2: Hospital Discharge: a comparison with hand-coded rules

7 A Nearly Optimal Search Strategy

Another way to evaluate how close CRYSTAL comes to optimal rules is to increase the amount of search expended in generalizing concept definitions. CRYSTAL is able to navigate efficiently through a large space of possible concept definitions because of the "greedy" nature of the algorithm. At every step in generalizing a concept definition, CRYSTAL is faced with several choices of constraints to relax. CRYSTAL makes the choice that seem to be the best at the time, even though a different choice may turn out later to have been better.

We tried an alternate approach that is more computationally expensive, but has a greater chance of making optimal choices. A *beam search* tries several paths in a search space in parallel. The amount of search effort is controlled by two parameters, the beam width w and branching factor b. When CRYSTAL generalizes from a seed instance, a *beam set* of size w is maintained. These are the best w generalized concept definitions found so far.

For each definition in the beam set, CRYSTAL finds b distinct relaxations by unifying with the most similar initial definition, the next most similar, and so forth. This produces a list of wb generalized definitions, which is sorted to keep the best w distinct definitions. The metric used to choose the best definitions is to count the number of positive training instances covered. If two definitions cover the same number of positive instances, the definition that covers fewer negative is considered better.

The CRYSTAL algorithm is equivalent to a beam search with $w = 1$. We ran experiments for the Management Succession domain and the Hospital Discharge domain at a range of beam sizes. Beam width was set to 1, 2, 5, and 10, with branching factor equal to the beam width.

Figure 6 shows results at beam width 1, 2, 5, and 10 for four representative Management Succession concepts. The shaded dot indicates the average of recall and precision.

Increasing the beam width results in a gain in recall that is almost exactly offset by a drop in precision. The greatest change in recall and precision comes in moving from beam width 1 to beam width 2. There is little effect from moving from beam width 5 to 10. This holds generally across concepts in both domains[7].

[7]Most of the changes in recall and precision are statistically significant, but none of the changes in average recall and precision.

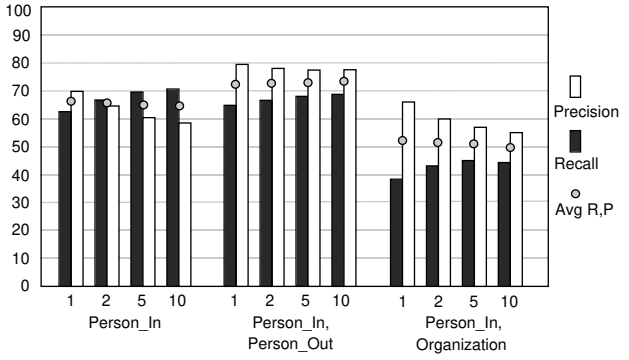

Figure 6: Management Succession results at beam width 1, 2, 5, and 10

Why should precision go down as recall goes up? This is a case of a well known phenomenon in machine learning called *overfitting* [Quinlan and Cameron-Jones 1995]. A machine learning algorithm may create a concept description that fits accidental characteristics of the training. Increased search for an optimal generalization turns out to increase the likelihood of finding rules that overfit the training data. This appears to increase accuracy when measured on the training data, but will actually reduce accuracy on the test set.

8 Conclusions

Automated text understanding requires an enormous amount of knowledge, even when the problem is narrowly focused on a limited domain. Rules must be created for each domain that identify concepts of interest based on domain-specific linguistic context. CRYSTAL helps overcome a knowledge engineering bottleneck by deriving these rules automatically from training examples.

CRYSTAL derives rules with performance nearly equal to that of hand-coded rules. Its learning algorithm navigates an extremely large space of possible rules efficiently. There is no benefit on the average from increasing the search effort to find optimal rules.

The range of information extraction tasks to which CRYSTAL can be applied is limited only be the ability of a user to identify target concepts that are grounded clearly in the text. A domain expert must mark each reference to the target concept in a set of representative texts. Developing a set of rules by hand also requires a set of annotated examples to guide development for all but the simplest of information extraction tasks. This means that CRYSTAL's training corpus is not an additional expense over a manual engineering approach.

References

[Clark and Niblett 1989] Clark, P. and Niblett, T. The CN2 Induction Algorithm. *Machine Learning*, 3, 261-283, 1989.

[Fisher *et al.* 1995] Fisher, D., Soderland, S., McCarthy, J., Feng, F., Lehnert, W. Description of the UMass System as Used for MUC-6. In *Proceedings of the Sixth Message Understanding Conference*, Morgan Kaufmann Publishers, 221-236, 1995.

[Huffman 1996] Huffman, S. Learning Information Extraction Patterns from Examples. *Connectionist, Statistical, and Symbolic approaches to Learning for Natural Language Processing.* Springer, 246-260, 1996.

[Kim and Moldovan 1992] Kim, J. and Moldovan, D. PALKA: A System for Linguistic Knowledge Acquisition. Technical Report PKPL 92-8, USC Department of Electrical Engineering Systems, 1992.

[Krupka 1995] Krupka, G. Description of the SRA System as Used for MUC-6. In *Proceedings of the Sixth Message Understanding Conference*, Morgan Kaufmann Publishers, 221-236, 1995.

[Michalski 1983] Michalski, R. S. A Theory and Methodology of Inductive Learning. *Artificial Intelligence*, 20, 111-161, 1983.

[MUC-4 1992] *Proceedings of the Fourth Message Understanding Conference*, Morgan Kaufmann Publishers, 1992.

[MUC-5 1993] *Proceedings of the Fifth Message Understanding Conference*, Morgan Kaufmann Publishers, 1993.

[MUC-6 1995] *Proceedings of the Sixth Message Understanding Conference*, Morgan Kaufmann Publishers, 1995.

[Quinlan and Cameron-Jones 1995] Quinlan, J.R. and Cameron-Jones, R.M. Oversearching and Layered Search in Emperical Learning. In *Proceedings of the Fourteenth International Joint Conference on Artificial Intelligence*, 1019-1024, 1995.

[Riloff 1993] Riloff, E. Automatically Constructing a Dictionary for Information Extraction Tasks. In *Proceedings of the Eleventh National Conference on Artificial Intelligence*, 811-816, 1993.

[Soderland *et al.* 1995] Soderland, S., Fisher, D., Aseltine, J., Lehnert, W. CRYSTAL: Inducing a Conceptual Dictionary. In *Proceedings of the Fourteenth International Joint Conference on Artificial Intelligence*, 1314-1321, 1995.

[Soderland 1996] Soderland, S. Learning Text Analysis Rules for Domain-specific Natural Language Processing. Ph.D. thesis, technical report UM-CS-1996-087 University of Massachusetts, Amherst, 1996.