Mixed-Initiative Development of Language Processing Systems

David Day, John Aberdeen, Lynette Hirschman,
Robyn Kozierok, Patricia Robinson and Marc Vilain

Advanced Information Systems Center
The MITRE Corporation
202 Burlington Road
Bedford, Massachusetts 01730 U.S.A.
{day,aberdeen,lynette}@mitre.org
{robyn,parann,mbv}@mitre.org

Abstract

Historically, tailoring language processing systems to specific domains and languages for which they were not originally built has required a great deal of effort. Recent advances in corpus-based manual and automatic training methods have shown promise in reducing the time and cost of this porting process. These developments have focused even greater attention on the bottleneck of acquiring reliable, manually tagged training data. This paper describes a new set of integrated tools, collectively called the Alembic Workbench, that uses a *mixed-initiative* approach to "bootstrapping" the manual tagging process, with the goal of reducing the overhead associated with corpus development. Initial empirical studies using the Alembic Workbench to annotate "named entities" demonstrates that this approach can approximately double the production rate. As an added benefit, the combined efforts of machine and user produce domain-specific annotation rules that can be used to annotate similar texts automatically through the Alembic NLP system. The ultimate goal of this project is to enable end users to generate a practical domain-specific information extraction system within a single session.

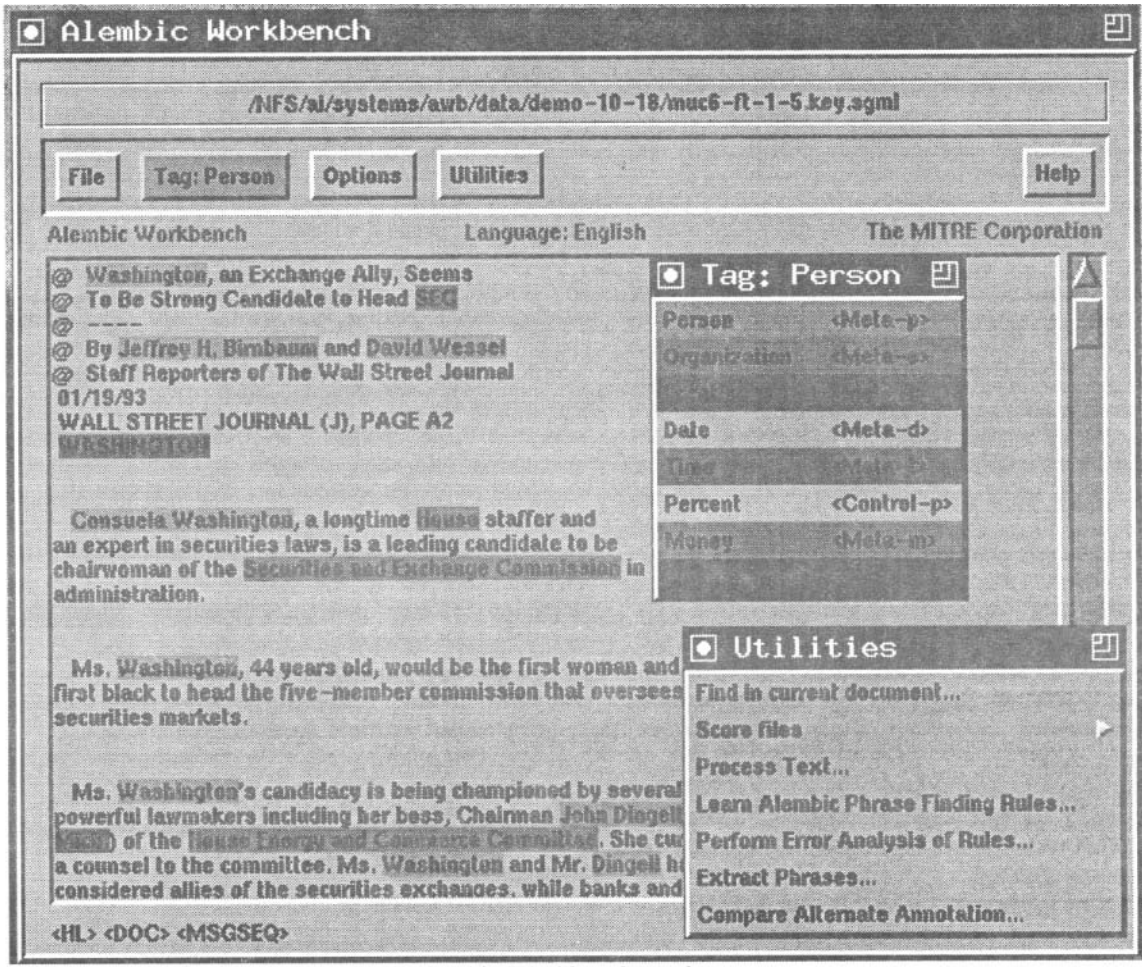
1. Introduction

In the absence of complete and deep text understanding, implementing information extraction systems remains a delicate balance between general theories of language processing and domain-specific heuristics. Recent developments in the area of corpus-based language processing systems indicate that the successful application of any system to a new task depends to a very large extent on the careful and frequent evaluation of the evolving system against training and test corpora. This has focused increased attention on the importance of obtaining reliable training corpora. Unfortunately, acquiring such data has usually been a labor-intensive and time-consuming exercise.

The goal of the Alembic Workbench is to dramatically accelerate the process by which language processing systems are tailored to perform new tasks. The philosophy motivating our work is to maximally reuse and re-apply every kernel of knowledge available at each step of the tailoring process. In particular, our approach applies a bootstrapping procedure to the development of the training corpus itself. By re-investing the knowledge available in the earliest training data to pre-tag subsequent un-tagged data, the Alembic Workbench can transform the process of manual tagging to one dominated by manual *review*. In the limit, if the pre-tagging process performs well enough, it becomes the domain-specific automatic tagging procedure itself, and can be applied to those new documents from which information is to be extracted.

As we and others in the information extraction arena have noticed, the quality of text processing heuristics is influenced critically not only by the power of one's linguistic theory, but also by the ability to evaluate those theories quickly and reliably. Therefore, building new information extraction systems requires an integrated environment that supports: (1) the development of a domain-specific annotated corpus; (2) the multi-faceted analysis of that corpus; (3) the ability to quickly generate hypotheses as to how to extract or tag information in that corpus; and (4) the ability to quickly evaluate and analyze the performance of those hypotheses. The Alembic Workbench is our attempt to build such an environment.

As the Message Understanding Conferences move into their tenth year, we have seen a growing recognition of the value of balanced evaluations against a common test corpus. What is unique in our approach is to integrate system development with the corpus annotation process itself. The early indications are that at the very least this integration can significantly increase the productivity of the corpus annotator. We believe that the benefits will flow in the other direction as well, and that a concomitant increase in system performance will follow as one applies the same mixed-initiative development environment to the problem of domain-specific tailoring of the language processing system.



@ Washington, an Exchange Ally, Seems
@ To Be Strong Candidate to Head SEC
@ ----
@ By Jeffrey H. Birnbaum and David Wessel
@ Staff Reporters of The Wall Street Journal
01/19/93
WALL STREET JOURNAL (J), PAGE A2
WASHINGTON

Consuela Washington, a longtime House staffer and an expert in securities laws, is a leading candidate to be chairwoman of the Securities and Exchange Commission in administration.

Ms. Washington, 44 years old, would be the first woman and first black to head the five-member commission that oversees securities markets.

Ms. Washington's candidacy is being championed by several powerful lawmakers including her boss, Chairman John Dingell Mich.) of the House Energy and Commerce Committee. She cu a counsel to the committee. Ms. Washington and Mr. Dingell h considered allies of the securities exchanges, while banks and

Figure 1. Screen dump of a typical Alembic Workbench session.

2. Alembic Workbench: A brief description

The Alembic Workbench provides a graphical user interface by which texts can be annotated using the mouse and user-defined key bindings. The Workbench mouse interface is engineered specifically to minimize hand motion. This allows text markup to proceed very quickly. Once a text has been marked up, the user's annotations are highlighted in colors specified by the user. A "mouse line" at the bottom of the text window provides further visual feedback indicating all of the annotations associated with the location under the mouse cursor, including document structure markup, if available. An example screen image from a typical session with the Workbench is shown above.

Our focus in building the Alembic Workbench is to provide a natural but powerful environment for annotating texts in the service of developing natural language processing systems. To this end we have incorporated a growing number of analysis and reporting features. The current set of utilities includes:

- A string-matching mechanism that can automatically replicate new markup to identical instances elsewhere in the document.
- A rule language for constructing task-specific phrase tagging and/or pre-tagging rule sets.
- A tool that generates phrase-based KWIC ("key-word in context") reports to help the user identify common patterns in the markup.
- A procedure that generates word lists based on their frequency. This tool also measures the degree to which a word occurs in different markup contexts.
- A visualization component for viewing inter-annotator (or key/answer) agreement.
- A scorer that allows arbitrary SGML markup to be selected for scoring.
- A full-featured interface to the multi-stage architecture of the Alembic text processing system.
- An interface to Alembic's phrase-rule learning system for generating new application-specific rule sets.
- The Alembic Workbench also provides specialized interfaces for supporting more complex, linked markup such as that needed for coreference. Another

interface is geared towards capturing arbitrary n-ary relations between tagged elements in a text (these have been called "Scenario Templates" in MUC).

More details about the implementation of the Workbench are provided in Section 7.

The development of the Alembic Workbench environment came about as a result of MITRE's efforts at refining and modifying our natural language processing system, Alembic [1,7], to new tasks: the Message Understanding Conferences (MUC5 and MUC6), and the TIPSTER Multi-lingual Entity Task (MET1). (See [6] for an overview and history of MUC6 and the "Named Entity Task".) The Alembic text processing system applies Eric Brill's notion of *rule sequences* [2,3] at almost every one of its processing stages, from part-of-speech tagging to phrase tagging, and even to some portions of semantic interpretation and inference.

While its name indicates its lineage, we do not view the Alembic Workbench as wedded to the Alembic text processing system alone. We intend to provide a well-documented API in the near future for external utilities to be incorporated smoothly into the corpus/system development environment. We envision two classes of external utilities: *tagging* utilities and *analysis* utilities. By integrating other *tagging* modules (including complete NLP systems), we hope those systems can be more efficiently customized when the cycle of *analysis, hypothesis generation* and *testing* is tightened into a well-integrated loop. The current version of the tool supports viewing, annotating and analyzing documents in 7-bit, 8-bit and 2-byte character sets. Current support includes the Latin-1 languages, Japanese (JIS), Chinese (GB1232), Russian, Greek and Thai.

3. Increasing manual annotation productivity through pre-tagging

A motivating idea in the design of the Alembic Workbench is to apply any available information as early and as often as possible to reduce the burden of manual tagging. In addition to careful interface design and support for user-customization, a core mechanism for enhancing this process is through *pre-tagging*.

The generation of reliably tagged text corpora requires that a human annotator read and certify all of the annotations applied to a document. This is especially true if the annotations are to be used for subsequent manual or automatic training procedures. However, much of the drudgery of this process can be removed if the most obvious and/or oft-repeated expressions can be tagged prior to the annotator's efforts. One way of doing this is to apply tags to any and all strings in a document that match a given string. This is the nature of the "auto-tagging" facility built-in to the Workbench interface. For example, in annotating journalistic document collections with "Named Entity" tags, one might want to simply pre-tag every occurrence of "President Clinton" with *Person*..[1] Of course, these actions should be taken with some care, since mis-tagging entities throughout a document might actually lead to an *increase* in effort required to accurately fix or remove tags in the document.

A more powerful approach is to allow *patterns*, or rules, to form the basis for this pre-tagging. The Alembic phrase-rule interpreter provides the basis for developing rule-based pre-tagging heuristics in the Workbench. In the current version of the Workbench, the user is free to compose these "phraser" rules and group them into specialized rule sets. Figure 2 shows an example sequence of rules that could be composed for pre-tagging a corpus with *Person* tags. The Brill control regime interprets these rules strictly sequentially: rule n is applied wherever in the text it can be; it is then discarded and rule n+1 is consulted. There is no unconstrained forward chaining using a "soup" of rules as in a standard production (or rule-based) system. The Alembic "phraser" rule interpreter has been applied to tagging named entities, sentence chunks, simple entity relations ("template element" in the parlance of MUC6), and other varieties of phrases.

```
(def-phraser-rule
   :anchor       :lexeme
   :conditions   (:left-1 :lex ("Mr." "Ms." "Dr." ...))
   :actions      (:create-phrase :person))

(def-phraser-rule
   :conditions   (:phrase :phrase-label :person)
                 (:right-1 :pos :NNP)
   :actions      (:expand :right-1))
```

Figure 2. An example Alembic rule sequence that (1) produces Person phrases around any word immediately to the right of a title and/or honorific, and then (2) grows the extent of the phrase to the right one lexeme, if that word is a proper noun.

4. Mixed-initiative text annotation

In addition to allowing users to define pre-tagging rules, we have developed a learning procedure that can be used to induce these rules from small training corpora. Operationally, an annotator starts by generating a small initial corpus and then invokes the learner to derive a set of pre-tagging rules. These rules can then be applied to new, unseen texts to pre-tag them. Figure 3 illustrates this bootstrapping cycle.

[1] The Named Entity task from MUC6 consists of adding tags to indicate expressions of type *Person, Location, Organization, Date, Time and Money*, see [6].

The earlier we can extract heuristic rules on the basis of manually tagged data, the earlier the user can be relieved from some portion of the chore of physically marking up the text—the user will need to edit and/or add only a fraction of the total phrases in a given document. In our experience of applying the Alembic phrase rule learner to named-entity and similar problems, our error-reduction learning method requires only modest amounts of training data. (We present performance details in Section 6.)

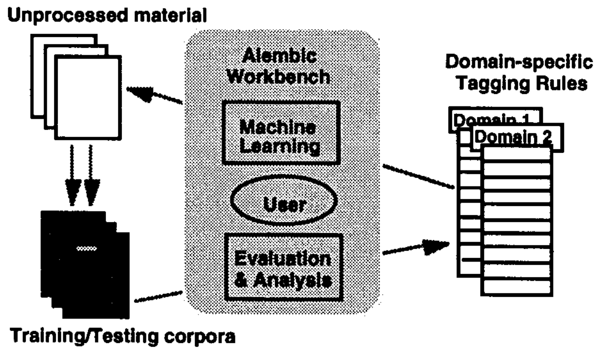


Figure 3. The Alembic Workbench seeks to involve the user in a corpus development cycle, making use of pre-tagging facilities, analysis facilities, and the automatic generation of pre-tagging rule sets through machine learning.

As the human annotator continues generating reliable training data, she may, at convenient intervals, re-invoke the learning process. As the amount of training data increases, the performance of the learned rules tends to increase, and so the amount of labor saved in pre-tagging subsequent training data is further increased. The bootstrapping effect tends to increase over time. For the "named entity" task in MUC6 approximately 25,000 words were provided as annotated training data by the conference organizers ("formal training" and "dryrun" data sets). Prior to developing the Alembic Workbench, we were able to use this amount of data in Alembic to generate a system performing at 85.2 P&R on unseen test data.[2] Based on the tagging rates we have measured thus far using the Workbench, it would take somewhere between 1.5 to 2.5 hours to tag these 25,000 words of data.

There is a limit on how much one can reduce the time-requirements for generating reliable training data—this is the rate required by a human domain expert to carefully read and edit a perfectly pre-annotated training corpus. Training data cannot be generated without this human investment.[3] Indeed, in situations where the quality of the data is particularly important (as it is in, say a multi-system evaluation such as MUC), it is typical that multiple reviews of the same corpus is performed by various annotators, especially given the known ambiguity of any annotation task definition.

5. Manual refinement of automatically derived pre-tagging heuristics

In the previous section we presented our approach to mixed-initiative corpus development and tagging heuristics without assuming any sophistication on the part of the human user beyond a clear understanding of the information extraction task being addressed. Usually, however, even a lay end-user is likely to have a number of intuitions about how the un-annotated data could be pre-tagged to reduce the burden of manual tagging. Hand-coded rules can be applied in concert with the machine-derived rules mentioned earlier. One way this can be done is by invoking the rule learning subsequent to the application of the hand-coded pre-tagging rules. On the other hand, if the user notices a consistent mistake being made by the machine-learned rules early in the bootstrapping process, the user can augment the machine-derived rule sequence with manually composed rules. In fact, every rule composed by the learning procedure is completely inspectable by the user, and so some users may want to modify individual machine-derived rules, perhaps to expand their generality beyond the particular data available in the emerging corpus.

This is another way, then, that the Alembic Workbench environment enables and encourages the mixed, or cooperative, application of human and machine skills to the combined task of developing a domain-specific corpus and set of extraction heuristics.

Of course, composing rules is somewhat akin to programming, and not all users will be inclined, or well-equipped, to become involved in this process. One impediment to end-users composing their own rules is the particular syntax of Alembic's phraser rules, so we anticipate exploring other, simpler rule languages that will encourage end-user participation. Another approach that we are interested in exploring involves supporting more indirect feedback or directives from the user that are rooted more closely to examples in the data.

[2] P&R (or F-measure) is a weighted combination of recall and precision.

[3] This is not to say that high-quality machine-tagged data cannot be generated faster than this, and that these data may indeed be helpful in the learning procedure of some other systems. But all such data will remain suspect as far as being considered part of an annotated training corpus until inspected by a human, given the vagaries of genre and style that can easily foil the most sophisticated systems.

Similarities and differences between manual and automatic rule formation

The automatic rule-learning procedure uses a generate-and-test approach to learn a *sequence* of rules. A set of rule schemata, defining a set of possible rule instances determines the rule space that the learning procedure explores. The learner uses indexing based on the actual data present in the corpus to help it explore the rule space efficiently. The learning process is initiated by deriving and applying an initial labeling function based on the differences between an un-annotated version and a correctly annotated version of the corpus. Then, during each learning cycle, the learner tries out applicable rule instances and selects the rule that most improves the score when applied to the corpus. The score is determined by evaluating the corpus as currently annotated against the correctly annotated version, using some evaluation function (generally precision, recall or F-measure). The corpus annotation is updated by applying the chosen rule, and the learning cycle repeats. This cycle is continued until a stopping criterion is reached, which is usually defined as the point where performance improvement falls below a threshold, or ceases. Other alternatives include setting a strict limit on the number of rules, and testing the performance improvement of a rule on a corpus distinct from the training set.

Of course, there are two important advantages that a human expert might have over the machine algorithm: linguistic intuition and world knowledge. Rules that include references to a single lexeme can be expanded to more general applicability by the human expert who is able to predict alternatives that lie outside the current corpus available to the machine. By supporting multiple ways in which rules can be hypothesized, refined and tested, the strengths of both sources of knowledge can be brought to bear.

6. Experimental Results

We are still in the early stages of evaluating the performance of the Alembic Workbench along a number of different dimensions. However, the results from early experiments are encouraging. Figure 4 compares the productivity rates using different corpus development utilities. These are indicated by the four categories on the X-axis: (1) using SGML-mode in emacs (by an expert user); (2) using the Workbench interface and "auto-tag" string-matching utility only; (3) using the Workbench following the application of learned tagging rules derived from 5 short documents—approximately 1,500 words, and (4) using the Workbench following the application of learned tagging rules again, but this time with the learned rules having trained on 100 documents (approximately 48,000 words), instead of only five documents.

As can be seen in these experiments, there is a clear increase in the productivity as a function of both the user interface (second column) and the application of pre-tagging rules (third and fourth columns). The large step in performance between columns three and four indicate that repeated invocation of the learning process during the intermediate stages of the corpus development cycle will likely result in acceleration of the annotation rate. (As it happens, these results are probably underestimating the pre-tagging productivity. The reason for this is that the version of the Workbench used was not yet able to incorporate date and time annotations generated by a separate pre-processing step; this date and time tagger performs at an extremely high level of precision for this genre—in the high nineties P&R.) These initial experiments involved a single expert annotator on a single tagging task (MUC6 named entity). The annotator was very familiar with the tagging task.

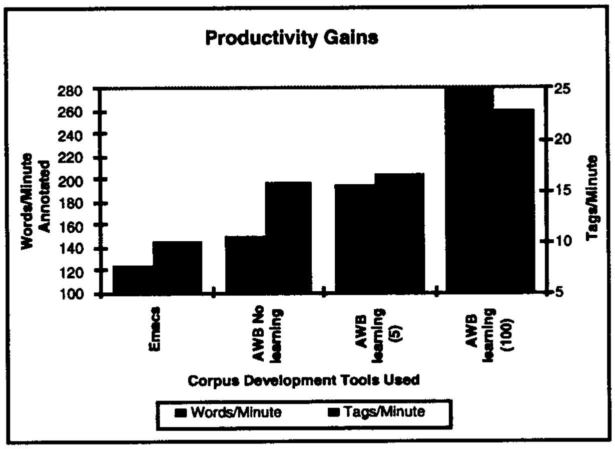


Figure 4. Two measures of corpus annotation productivity using the Alembic Workbench. The X-axis indicates what kind of corpus-development utilities were used: (1) SGML-mode of emacs text editor; (2) Workbench (AWB) manual interface only, (3) AWB rule-learning bootstrap method with 5-document training set; (4) AWB rule-learning bootstrap method with 100-document training set. See discussion in text.

To place this in the perspective of the human annotator, after only about 15 minutes of named entity tagging, having annotated some 1,500 words of text with approximately 150 phrases, the phrase rule learner can derive heuristic rules that produce a pre-tagging performance rate (P&R) of between 50 and 60 percent. Of course, this performance is far short of what is needed for a practical extraction system, but it already constitutes a major source for labor savings, since 50 to 60 percent of the annotations that need to be moused (or clicked) in are already there. Since the precision at this early stage is only around 60 percent, there will be extra phrases that need (1) to be removed, (2) their assigned category changed (from, say, *organization* to *person*), or (3) their boundaries adjusted. It turns out that for the first two of these kinds of precision errors, the manual corrections are extremely quick to perform. (Boundaries are not really difficult to modify, but the time required is approximately the same as inserting a tag from scratch.) In addition, making these corrections removes both a precision and a recall error at the same time. Therefore, it turns out that even at this very early stage, the modest pre-tagging performance gained from applying the learning procedure provides measurable performance improvement.

In order to obtain more detailed results on the effect of pre-tagging corpora, we conducted another experiment in which we made direct use of the iterative automatic generation of rules from a growing manually-tagged corpus. Using the same skilled annotator, we introduced a completely new corpus for which named-entity tagging happened to be needed within our company. We randomly divided approximately 50 documents of varying sizes into five groups. The word counts for these five groups were: Groupl: 19,300; Group2: 13,800; Group3: 6,300; Group4: 15,800; Group5: 8,000; for a total of 63,000 words. After manually tagging the first group, we invoked the rule learning procedure. Applying the learning procedure on each training set required two to three hours of elapsed time on a Sun Sparc Ultra. The new tagging rules were then applied to the next ten documents prior to being manually tagged/edited. This enlarged corpus was then used to derive a new rule set to be applied to the next group of documents, and so on. A summarization of the results are presented in Figure 5.

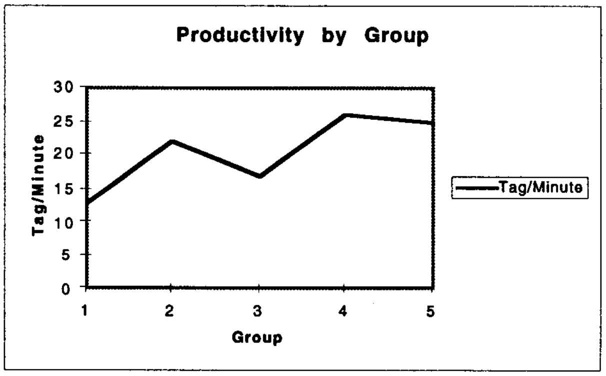


Figure 5. Tagging productivity gains with the incremental application of automatically acquired rule sets.

The first observation we make is that there is a clear and obvious direction of improvement—by the time 30 documents have been tagged, the annotation rate on Group 4 has increased considerably. It is important to note, however, that there is still noise in the curve. In addition, the granularity is perhaps still too coarse to measure the incremental influences of pre-tagging rules.

Clearly, more experiments are called for—we plan to conduct these across different annotators, task types, and languages, to better evaluate productivity, quality and other aspects of the annotation process.

It is extremely difficult to control many of the features that influence the annotation process, such as the intrinsic complexity of the topic in a particular document, the variation in tag-density (tags per word) that may occur, the user's own training effect as the structure and content of documents become more familiar, office distractions, etc. In order to gain a better understanding of the underlying tagging performance of the rule learner, and so separate out some of these human factors issues, we ran an automated experiment in which different random subsets of sentences were used to train rule sets, which were then evaluated on a static test corpus. The results shown in Figure 6 give some indication of the ability of the rule-sequence learning procedure to glean useful generalizations from meager amounts of training data.

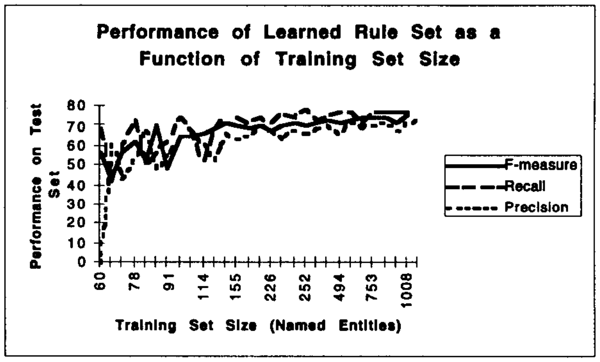


Figure 6. Performance of learned rules on independent test set of 662 sentences.

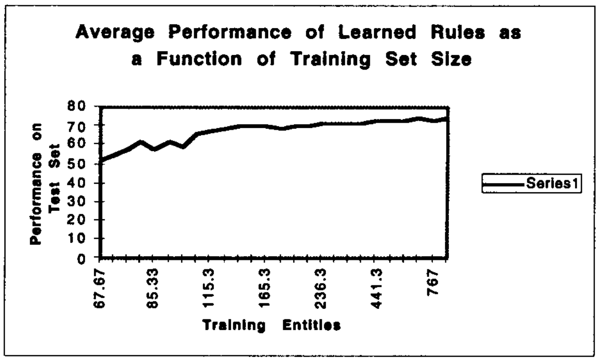


Figure 7. Averaged F-measure performance figures.

One clear effect of increasing training set size is a reduction in the sensitivity of the learning procedure to particular training sets. We hypothesize that this effect is partly indicative of the generalization behavior on which the learning procedure is based, which amplifies the effects of choosing more or less representative training sentences by chance. Since the learning process is not merely memorizing phrases, but generating contextual rules to try to predict phrase types and extents, the rules are very sensitive to extremely small selections of training sentences. Figure 7 shows the F-measure performance smoothed by averaging neighboring data points, to get a clearer picture of the general tendency.

We should note that the Alembic Workbench, having been developed only recently in our laboratory, was not available to us in the course of our effort to apply the Alembic system to the MUC6 and MET tasks. Therefore we have not been able to measure its influence in preparing for a particular new text processing task. We intend to use the system to prepare for future evaluations (including MUC7 and MET2) and to carefully evaluate the Alembic Workbench as an environment for the mixed-initiative development of information extraction systems in multiple languages.

7. Implementation

The Alembic Workbench interface has been written in Tcl/Tk. Some of the analysis and reporting utilities (available from within the interface as well as Unix command-line utilities) are written in Perl, C or Lisp. The separate Alembic NLP system consists of C preprocessing taggers (for dates, word and sentence tokenization and part-of-speech assignments) and a Lisp image that incorporates the rest of Alembic: the phrase-rule interpreter, the phrase rule learner, and a number of discourse-level inference mechanisms described in [8]. This code currently runs on Sun workstations running Sun OS 4.1.3 and Solaris 2.4 (Sun OS 5.4) and greater; we have begun porting the system to Windows NT/Windows 95. We anticipate providing an API for integrating other NLP systems in the near future.

The Workbench reads and saves its work in the form of SGML-encoded files, though the original document need not contain any SGML mark-up at all. These files are parsed with the help of an SGML normalizer.[4] During the course of the annotation process the Workbench uses a "Parallel Tag File" (PTF) format, which separates out the embedded annotations from the source text, and organizes user-defined sets of annotations within distinct "tag files." While these files are generally hidden from the user, they provide a basis for the combination and separation of document annotations ("tagsets") without needing to modify or otherwise disturb the base document. This allows the user to view only Named Entity tags, or only tokenization tags, or any desired subset of tagsets. Thus, the Workbench is written to be TIPSTER-compliant, though it is not itself a document manager as envisioned by that architecture (see [5]). We anticipate integrating the Workbench with other TIPSTER compliant modules and document managers via the exchange of SGML-formatted documents. The Parallel Tag File (PTF) format used by the Workbench provides another means by which a translator could be written.

8. Future Work

Broadly defined, there are two distinct types of users who we imagine will find the Workbench useful: NLP researchers and information extraction system end-users. While our dominant focus so far has been on supporting the language research community, it is important to remember that new domains for language processing generally, and information extraction in particular, will have their own domain experts, and we want the text annotation aspects of the tool to be quite usable by a wide population. In this vein we would like to enable virtually any user to be able to compose new patterns (rules) for performing pre-tagging on the data. While the current rule language has a simple syntax, as well as an extremely simple control regimen, we do not imagine all users will want to engage directly in an exploration for pre-tagging rules. A goal for our future research is to explore new methods for incorporating end-user feedback to the learning procedure. This feedback might include modifying a very simplified form of a single rule for greater generality by integrating thesauri to construct word-list suggestions. We also would like to give users immediate feedback as to how a single rule applies (correctly and incorrectly) to many different phrases in the corpus.

In this paper we have concentrated on the named entity task as a generic case of corpus annotation. Of course, there are many different ways in which corpora are being annotated for many different tasks. Some of the specific extensions to the user interface that we have already begun building include part-of-speech tagging (and "dense" markup more generally), and full parse syntactic tagging (where we believe reliable training data can be obtained much more quickly than heretofore). In these and other instances the tagging process can be accelerated by applying partial knowledge early on, transforming the task once again into that of editing and correcting. Most of these tagging tasks would be improved by making use of methods that preferentially select ambiguous data for manual annotation—for example, as described in [4].

There are a number of psychological and human factors issues that arise when one considers how the pre-annotated data in a mixed-initiative system may affect

[4] In cases where documents use some of the more complex aspects of SGML, the user supplies a Document Type Description (DTD) file for use in normalization. For simple SGML documents, or documents with no original SGML markup at all, no DTD needs to be specified.

the human editing or post-processing. If the pre-tagging process has a relatively high recall, then we hypothesize that the human will tend increasingly to trust the pre-annotations, and thereby forget to read the texts carefully to discover any phrases that escaped being annotated. A similar effect seems possible for relatively high precision systems, though proper interface design (to highlight the type assigned to a particular phrase) should be able to mitigate these tendencies. A more subtle interaction is "theory creep," where the heuristics induced by the machine learning component begin to be adopted by the human annotator, due, in many cases, to the intrinsic ambiguity of defining annotation tasks in the first place. In all of these cases the most reliable method for detecting these human/machine interactions is probably to use some representative sub-population of the corpus documents to measure and analyze the inter-annotator agreement between human annotators who have and who have not been exposed to the machine derived heuristics for assigning annotations.

9. Conclusions

On the basis of observing our own and others' experiences in building and porting natural language systems for new domains, we have come to appreciate the pivotal role played in continuous evaluation throughout the system development cycle. But evaluation rests on an oracle, and for text processing, that oracle is the training and test corpora for a particular task. This has led us to develop a tailoring environment which focuses all of the available knowledge on accelerating the corpus development process. The very same learning procedure that is used to bootstrap the manual tagging process leads eventually to the derivation of tagging heuristics that can be applied in the operational setting to unseen documents. Rules derived manually, automatically, and through a combination of efforts have been applied successfully in a variety of languages, including English, Spanish, Portuguese, Japanese and Chinese.

The tailoring environment, known as the Alembic Workbench, has been built and used within our organization, and we are making it available to other organizations involved in the development of language processing systems and/or annotated corpora. Initial experiments indicate an significant improvement in the rate at which annotated corpora can be generated using the Alembic Workbench methodology. Earlier work has shown that with the training data obtained in the course of only a couple of hours of text annotation, an information extraction system can be induced purely automatically that achieves a very competitive level of performance.

References

[1] John Aberdeen, John Burger, David Day, Lynette Hirschman, David Palmer, Patricia Robinson, and Marc Vilain. 1996. The Alembic system as used in MET. In *Proceedings of the TIPSTER 24 Month Workshop*, May.

[2] Eric Brill. 1992. A simple rule-based part of speech tagger. In *Proceedings of the Third Conference on Applied Natural Language Processing*, Trento.

[3] Eric Brill. 1993. *A Corpus-Based Approach to Language Learning*. Ph.D. thesis, University of Pennsylvania, Philadelphia, Penn.

[4] Sean P. Engelson and Ido Dagan. 1996. Minimizing manual annotation cost in supervised training from corpora. *Computation and Linguistic E-Print Service (cmp-lg/9606030)*, June.

[5] Ralph Grishman. 1995. TIPSTER phase II architecture design. *World Wide Web document.* URL=http://cs.nyu.edu/cs/faculty/grishman/tipster.html

[6] Ralph Grishman and Beth Sundheim. 1996. Message Understanding Conference—6: A Brief History. In *International Conference on Computational Linguistics*, Copenhagen, Denmark, August. The International Committee on Computational Linguistics.

[7] Marc Vilain and David Day. 1996. Finite-state parsing by rule sequences. In *International Conference on Computational Linguistics*, Copenhagen, Denmark, August. The International Committee on Computational Linguistics.

[8] Marc Vilain. 1993. Validation of terminological inference in an information extraction task. In *Proceedings of the ARPA Workshop on Human Language Technology*, Plainsboro, New Jersey.