Using Corpus-Based Techniques to Customize a Lexicon for a New Domain

G. Jan Wilms
Department of Mathematics and Computer Science
Union University
2447 HWY 45 Bypass Box 1857
Jackson, TN 38305-3691
wilms@cs.msstate.edu

Abstract

A hybrid system is proposed that combines traditional knowledge-based approaches with stochastic, corpus based techniques to improve the portability and robustness of natural language parsing systems. Techniques to automatically customize the lexicon to a new domain are investigated by applying the *PUNDIT* parser to the *Merck Veterinary Manual*.

The shortcomings of **traditional knowledge-based approaches** to natural language processing (lack of portability and robustness) quickly become apparent when a system is applied to a new domain. The alternative method of **stochastic and corpus-based analysis**, which is coming back in vogue after disappointing result with machine translation in the late fifties, suffers from sparse training data and is an inadequate model for recognizing long distance dependencies. A more promising strategy is to build a **hybrid system**, which augments a traditional wide-coverage parser with statistical methods to facilitate scaling up and porting to a new sublanguage.

The vehicle for this research is a **wide-coverage parsing system** developed by Unisys, named ***PUNDIT*** (Prolog UNDerstanding of Integrated Text). It is a logic grammar implementation of Sager's *string grammar*. Prolog's top-down backtracking mechanism uses a set of context free BNF rules that have been augmented with restrictions and a regularization rule. The latter converts the surface parse tree into a canonical form called *ISR (Intermediate Syntactic Representation*); it regularizes the relationships between predicates and arguments, and hides some of the details unnecessary for semantic analysis. *PUNDIT* was explicitly designed to separate domain-independent from domain-specific knowledge. The domain-independent module consists of some 65,000 lines of Prolog code that took several person-years to develop. It contains the procedural components and a large set of general lexical entries and grammar rules. This engine has been applied to several different domains (*Voyager, ATIS, MUC*, and currently a directory assistance application with voice input), each time requiring extensive effort to provide specialized grammar rules, lexicon additions, and domain specific semantic rules.

As far as **syntactic parsing** goes, *PUNDIT* goes through the following steps: a lexical lookup phase, which is done *prior to* (not in tandem with) the syntactic parsing. Next the *restriction grammar* constructs a surface tree and the *ISR*, which in turn is input to a filter of semantic co-occurrence constraints. Phrases are scrutinized for allowable and negative co-occurrence patterns, which is instrumental in limiting the number of allowable parses.

The *PUNDIT* core is being applied to a **new domain**, the ***Merck Veterinary Manual***. This on-line reference manual is the testbed for a large ongoing project at Mississippi State University, which attempts to automate the extraction of the expert knowledge contained in the text. Besides the immediate goal of providing parsed input to the *knowledge analyzer* component, the main research issue is to investigate what type of information is required to move to a new domain, and how this port can be automated as much as possible.

PUNDIT allows both interactive and **batch processing**, and can look at input sentences in isolation or as part of a larger context to which it applies discourse analysis. For our research batch mode is used, although we are currently focussing on syntactic analysis solely. A UNIX shell script has been written to convert the raw text from the *Merck* manual to the batch format specified by the *PUNDIT* designers, with restrictions and protocols deriving both from *PUNDIT* design decisions and the Prolog language (e.g. initial upper case letters must be converted because they indicate variables). The conversion is complicated somewhat by the fact that the on-line text of the *Merck* manual doesn't have any line breaks and is interspersed with typesetting codes. Several other tools have been written for gathering statistics and performing regression analysis.

Corpus-based techniques can contribute to robustness and portability by augmenting the **three knowledge sources** that *PUNDIT* employs for syntactic parsing: lexicon, grammar, and selectional patterns. In the last few years it has become clear that many domain sublanguages have grammars of their own. Although stochastic methods that have attempted to *induce* a grammar from scratch have met with only limited success, they may prove useful in **augmenting** an existing grammar. Also, *PUNDIT* itself contains some hooks that allow the gathering of statistical information on the frequency of application of *semantic* rules, which may be helpful in selecting the most likely parse first. Finally, stochastic methods have their use not only in facilitating the porting process to a different domain by adding missing elements, but they can also enhance performance by **pruning** the lexicon/grammar of rules and word senses/parts-of-speech that don't occur in the corpus. Such a minimal, customized grammar/lexicon helps decrease the search space and eliminate many spurious parses.

There are several possible **sources** for corpus-based techniques; as *on-line dictionaries* become increasingly available, they are looked upon as a convenient repository of all kinds of knowledge, some readily accessible, some requiring challenging and sophisticated analysis (e.g. common sense knowledge). Dictionaries seem an obvious candidate for lexical acquisition, but *too much* information is almost as bad as too little of it, especially if the multiple options aren't properly discriminated. Another approach is to use the *text source as a resource*; several researchers have reported very high accuracy (>95%) with probabilistic taggers, which tag the words with part-of-speech labels. They are based on *hidden Markow models*, which are built from a small subset of hand-tagged text. Other on-line texts, like the *Brown Corpus* and *LOB Corpus* may serve as evidence (outside the domain at hand) for co-occurrence analysis. Finally, several organizations are making *treebanks* available, files of (manually) *parsed* text, with a structure as theory-neutral as possible (usually in the form of bracketing). Having grammatical information available, like clause and phrase boundaries, prepositional phrase attachment and conjunction disambiguation, makes it

possible to gather statistics on long-distance dependencies.

At this point we are concentrating on investigating corpus-based techniques to augment the **lexicon**. Lexical entries contain morphological and part-of-speech information, and limited syntactic features, like complement types of verbs. Lexical semantics are assigned later, during semantic analysis. *PUNDIT* adopts the same lexical subclasses as the *String Grammar*. A sample entry for *head* follows:

```
:(head,root:head,[v:[12],tv:[12,plural],12:[objlist:[nstgo,
pn:[pval:[toward,to]],npn:[pval:[toward,to]]]]]).
:(headed,root:head,[tv:[12,past],ven:[14],14:[12,pobjlist
:[nullobj,p:[pval:[toward,to]],pn:[pval:[toward,to]]]]]).
:(heading,root:head,[ving:[12]]).
:(heads,root:head,[tv:[12,singular]]).
:(heading,root:heading,[n:[singular]]).
:(headings,root:heading,[n:[plural]]).
```

There are two concerns with respect to the lexicon in porting *PUNDIT* to a new domain: **adding new entries** for unknown words (or, more subtly, add new senses/parts-of-speech/features) and **removing unnecessary lemmas** to speed up the parsing. As it turns out, since the lexical lookup occurs *prior* to parsing, the size of the lexicon has minimal impact on the parsing speed, but reducing the set of senses and features *will* reduce the search space and cause less backtracking. *PUNDIT* has a user-friendly front-end for interactively updating the lexicon, but it only recognizes the major features, and, though probably more accurate, is less desirable than automated routines. Currently, when operating in batch mode, *PUNDIT* will attempt to *guess* the part of speech of a word, but since this is done prior to syntactic analysis, it must be done purely on the basis of morphology. Whereas this simple heuristic does an adequate job of recognizing adverbs, gerunds, and past participles, it defaults to *noun* for all other cases! Whereas this doesn't prevent *PUNDIT* from recognizing sentences with new adjectives (it parses the clause as a noun-noun modifier), it causes the parser to completely fail for new verbs (and in the few cases where it *does* find a parse, the result is predictably completely wrong). It is also worth pointing out that if a novel word occurs multiple time in a sentence, *PUNDIT* will blindly guess again (and come up with the same answer).

Three strategies have been explored to automate the addition of new lemmas: using a **machine-readable dictionary** (like the *Funk and Wagnall's*). While this is certainly more accurate than the suffix heuristic, it is undesirable because in trying to be exhaustive the dictionary offers too many variants without discriminating amongst them (even if the dictionary did rank senses and parts-of-speech, this would vary depending on the application domain). The dictionary can also be used to find the root of words, as look-up is more accurate than rule-based reduction. A second approach is to use a **stochastic tagger** (like the one developed at *Mississippi State University*) to gather information on the actual part-of-speech of words in the *Merck* chapter (a minor problem here is that there isn't a one-to-one mapping between the tagger's and *PUNDIT*'s tag set). This leads to a major improvement in number of sentenses that can be parsed (irrespective of whether the resulting parse tree is correct), mostly because of the misclassified verbs mentioned above. In some instances, though, sentences that were previously recognized no longer are accepted; this can be attributed partially to the fact that the tagger occasionally does make mistakes (e.g. *industry* as *adj.*), and because of *leaks* in the grammar that surface now the new word is correctly classified (i.e. the parse *PUNDIT* had originally come up with was based on an incorrect guess, and most likely a *false positive*). Some of the errors of the probabilistic tagger can be caught by checking

its assignment against the set of parts-of-speech for that word in the dictionary, but there are many counter-examples. Finally, instead of using the probabilistic tagger as a pre-processor to add new entries to the lexicon from which the lexical component must choose, another possibility is to limit the search space by **rewriting the lexical component** to simply go with the assignment made by the tagger for that word in that particular sentence.

Currently we are looking into using corpus-based techniques to go beyond part-of-speech assignment, and attempt to **discover some of the lexical features**, as for example *countable/mass* nouns. *PUNDIT* again uses a heuristic based on morphology to classify nouns as countable (all singular words with an *-ss* suffix and plural nouns ending in *-es*), but this is very questionable. A better technique may be to look for co-occurrence clues like numerals (*three books/*informations*), indefinite articles (*a book/*information*) and certain indefinite quantifiers (*many books - much information*). For long(er)-distance dependencies, it may be necessary to look at a bracketed treebank. It remains to be seen how many such syntactic features can be automatically discovered to facilitate porting to a new application.

Ball Catherine, Dowding John, Lang FranÇois, Weir Carl. (1988) *PUNDIT user's guide*, Technical Report, Unisys Corporation.

Boggess L, Agarwal R, Davis R (1991) Disambiguation of Prepositional Phrases in Automatically Labelled Technical Text. *Proceedings of AAAI-91 National Conference on AI*, Menlo Park, CA, pp 155-159.

Church K (1988) A Stochastic Parts Program and Noun Phrase Parser for Unrestricted Text. *Proceedings of the 2nd ACL Conference on Applied NLP*, pp 136-143.

Grishman R, Kittredge R (eds.) (1986) *Analyzing Language in Restricted Domains: Sublanguage Description Processing*. Lawrence Erlbaum Associates Publishers, Hillsdale NJ.

Kittredge R, Lehrberger J (eds.) (1982) *Sublanguage: Studies of Language in Restricted Semantic Domains*. de Gruyter, NY.

Sager Naomi (ed). (1981) *Natural Language Information Processing*. Addison-Wesley Publishing.