Fast Statistical Parsing of Noun Phrases for Document Indexing

Chengxiang Zhai
Laboratory for Computational Linguistics
Carnegie Mellon University
Pittsburgh, PA 15213
{cz25@andrew.cmu.edu}

Abstract

Information Retrieval (IR) is an important application area of Natural Language Processing (NLP) where one encounters the genuine challenge of processing large quantities of unrestricted natural language text. While much effort has been made to apply NLP techniques to IR, very few NLP techniques have been evaluated on a document collection larger than several megabytes. Many NLP techniques are simply not efficient enough, and not robust enough, to handle a large amount of text. This paper proposes a new probabilistic model for noun phrase parsing, and reports on the application of such a parsing technique to enhance document indexing. The effectiveness of using syntactic phrases provided by the parser to supplement single words for indexing is evaluated with a 250 megabytes document collection. The experiment's results show that supplementing single words with syntactic phrases for indexing consistently and significantly improves retrieval performance.

1 Introduction

Information Retrieval (IR) is an increasingly important application area of Natural Language Processing (NLP). An IR task can be described as to find, from a given document collection, a subset of documents whose content is relevant to the information need of a user as expressed by a query. As the documents and query are often natural language texts, an IR task can usually be regarded as a special NLP task, where the document text and the query text need to be processed in order to judge the relevancy. A general strategy followed by most IR systems is to transform documents and the query into certain level of representation. A query representation can then be compared with a document representation to decide if the document is relevant to the query. In practice, the level of representation in an IR system is quite "shallow" — often merely a set of word-like strings, or indexing terms. The process to extract indexing terms from each document in the collection is called indexing. A query is often subject to similar processing, and the relevancy is judged based on the matching of query terms and document terms. In most systems, weights are assigned to terms to indicate how well they can be used to discriminate relevant documents from irrelevant ones.

The challenge in applying NLP to IR is to deal with a large amount of unrestricted natural language text. The NLP techniques used must be very efficient and robust, since the amount of text in the databases accessed is typically measured in gigabytes. In the past, NLP techniques of different levels, including morphological, syntactic/semantic, and discourse processing, were exploited to enhance retrieval (Smeaton 92; Lewis and Sparck Jones 96), but were rarely evaluated using collections of documents larger than several megabytes. Many NLP techniques are simply not efficient enough or are too labor-intensive to successfully handle a large size document set. However, there are some exceptions. Evans et al. used selective NLP techniques, that are especially robust and efficient, for indexing (Evans et al. 91). Strzalkowski reported a fast and robust parser called TTP in (Strzalkowski 92; Strzalkowski and Vauthey 92). These NLP techniques have been successfully used to process quite large collections, as shown in a series of TREC conference reports by the CLARIT^{TM}[1] system group and the New York University (later GE/NYU) group (cf., for example, (Evans and Lefferts 95; Evans et al. 96), and (Strzalkowski 95; Strzalkowski et al. 96)) These research efforts demonstrated the feasibility of using selective NLP to handle large collections. A special NLP track emphasizing the evaluation of NLP techniques for IR is currently held in the context of TREC (Harman 96).

In this paper, a fast probabilistic noun phrase parser is described. The parser can be exploited to

[1] CLARIT is a registered trademark of CLARITECH Corporation.

automatically extract syntactic phrases from a large amount of documents for indexing. A 250-megabyte document set[2] is used to evaluate the effectiveness of indexing using the phrases extracted by the parser. The experiment's results show that using syntactic phrases to supplement single words for indexing improves the retrieval performance significantly. This is quite encouraging compared to earlier experiments on phrase indexing. The noun phrase parser provides the possibility of combining different kinds of phrases with single words.

The rest of the paper is organized as follows. Section 2 discusses document indexing, and argues for the rationality of using syntactic phrases for indexing; Section 3 describes the fast noun phrase parser that we use to extract candidate phrases; Section 4 describes how we use a commercial IR system to perform the desired experiments; Section 5 reports and discusses the experiment results; Section 6 summarizes the conclusions.

2 Phrases for Document Indexing

In most current IR systems, documents are primarily indexed by single words, sometimes supplemented by phrases obtained with statistical approaches, such as frequency counting of adjacent word pairs. However, single words are often ambiguous and not specific enough for accurate discrimination of documents. For example, only using the word "*bank*" and "*terminology*" for indexing is not enough to distinguish "*bank terminology*" from "*terminology bank*". More specific indexing units are needed. Syntactic phrases (i.e., phrases with certain syntactic relations) are almost always more specific than single words and thus are intuitively attractive for indexing. For example, if "*bank terminology*" occurs in the document, then, we can use the phrase "*bank terminology*" as an additional unit to supplement the single words "*bank*" and "*terminology*" for indexing. In this way, a query with "*terminology bank*" will match better with the document than one with "*bank terminology*", since the indexing phrase "*bank terminology*" provides extra discrimination.

Despite the intuitive rationality of using phrases for indexing, syntactic phrases have been reported to show no significant improvement of retrieval performance (Lewis 91; Belkin and Croft 87; Fagan 87). Moreover Fagan (Fagan 87) found that syntactic phrases are not superior to simple statistical phrases. Lewis discussed why the syntactic phrase indexing has not worked and concluded that the problems with syntactic phrases are for the most part statistical (Lewis 91). Indeed, many (perhaps most) syntactic phrases have very low frequency and tend to be over-weighted by the normal weighting method. However, the size of the collection used in these early experiments is relatively small. We want to see if a much larger size of collection will make a difference. It is possible that a larger document collection might increase the frequency of most phrases, and thus alleviate the problem of low frequency.

We only consider noun phrases and the subphrases derived from them. Specifically, we want to obtain the full modification structure of each noun phrase in the documents and query. From the viewpoint of NLP, the task is noun phrase parsing (i.e., the analysis of noun phrase structure). When the phrases are used only to supplement, not replace, the single words for indexing, some parsing errors may be tolerable. This means that the penalty for a parsing error may not be significant. The challenge, however, is to be able to parse gigabytes of text in practically feasible time and as accurately as possible. The previous work taking on this challenge includes (Evans et al. 91; Evans et al. 96; Evans and Zhai 96; Strzalkowski and Carballo 94; Strzalkowski et al. 95) among others. Evans et al. exploited the "attestedness" of subphrases to partially reveal the structure of long noun phrases (Evans et al. 91; Evans et al. 96). Strzalkowski et al. adopted a fast Tagged Text Parser (TTP) to extract head modifier pairs including those in a noun phrase (Strzalkowski 92; Strzalkowski and Vauthey 92; Strzalkowski and Carballo 94; Strzalkowski et al. 95). In (Strzalkowski et al. 95), the structure of a noun phrase is disambiguated based on certain statistical heuristics, but there seems to be no effort to assign a full structure to every noun phrase. Furthermore, manual effort is needed in constructing grammar rules. Thus, the approach in (Strzalkowski et al. 95) does not address the special need of scalability and robustness along with speed. Evans and Zhai explored a hybrid noun phrase analysis method and used a quite rich set of phrases for document indexing (Evans and Zhai 96). The indexing method was evaluated using the Associated Press newswire 89 (AP89) database in Tipster Disk1, and a general improvement of retrieval performance over the indexing with single words and full noun phrases was reported. However, the phrase extraction system as reported in (Evans and Zhai 96) is still not fast enough to deal with document collections measured by gigabytes.[3]

We propose here a probabilistic model of noun phrase parsing. A fast statistical noun phrase parser has been developed based on the probabilistic model. The parser works fast and can be scaled up to parse gigabytes text within acceptable time.[4] Our goal is to generate different kinds of candidate syntactic

[2]the Wall Street Journal database in Tipster Disk2 (Harman 96)

[3]It was reported to take about 3.5 hours to process 20 MB documents

[4]With a 133MH DEC alpha workstation, it is estimated to parse at a speed of 4 hours/gigabyte-text or 8 hours/gigabyte-nps, after 20 hours of training with 1 gigabyte text

phrases from the structure of a noun phrase so that the effectiveness of different combinations of phrases and single words can be tested.

3 Fast Noun Phrase Parsing

A fast and robust noun phrase parser is a key to the exploration of syntactic phrase indexing. Noun phrase parsing, or noun phrase structure analysis (also known as compound noun analysis[5]), is itself an important research issue in computational linguistics and natural language processing. Long noun phrases, especially long compound nouns such as "*information retrieval technique*", generally have ambiguous structures. For instance, "*information retrieval technique*" has two possible structures: "*[[information retrieval] technique]*" and "*[information [retrieval technique]]*". A principal difficulty in noun phrase structure analysis is to resolve such structural ambiguity. When a large corpus is available, which is true for an IR task, statistical preference of word combination or word modification can be a good clue for such disambiguation. As summarized in (Lauer 95), there are two different models for corpus-based parsing of noun phrases: the adjacency model and the dependency model. The difference between the two models can be illustrated by the example compound noun "*information retrieval technique*". In the adjacency model, the structure would be decided by looking at the adjacency association of "*information retrieval*" and "*retrieval technique*". "*information retrieval*" will be grouped first, if "*information retrieval*" has a stronger association than "*retrieval technique*", otherwise, "*retrieval technique*" will be grouped first. In the dependency model, however, the structure would be decided by looking at the dependency between "*information*" and "*retrieval*" (i.e., the tendency for "*information*" to modify "*retrieval*") and the dependency between "*information*" and "*technique*". If "*information*" has a stronger dependency association with "*retrieval*" than with "*technique*", "*information retrieval*" will be grouped first, otherwise, "*retrieval technique*" will be grouped first. The adjacency model dates at least from (Marcus 80) and has been explored recently in (Liberman and Sproat 92; Pustejovsky et al. 93; Resnik and Hearst 93; Lauer 95; Strzalkowski et al. 95; Evans and Zhai 96); The dependency model has mainly been studied in (Lauer 94). Evans and Zhai (Evans and Zhai 96) use primarily the adjacency model, but the association score also takes into account some degree of dependency. Lauer (Lauer 95) compared the adjacency model and the dependency model for compound noun disambiguation, and concluded that the dependency model provides a substantial advantage over the adjacency model.

We now propose a probabilistic model in which the dependency structure, or the modification structure, of a noun phrase is treated as "hidden", similar to the tree structure in the probabilistic context-free grammar (Jelinek et al. 90). The basic idea is as follows.

A noun phrase can be assumed to be generated from a word modification structure (i.e., a dependency structure). Since noun phrases with more than two words are structurally ambiguous, if we only observe the noun phrase, then the actual structure that generates the noun phrase is "hidden". We treat the noun phrases with their possible structures as the complete data and the noun phrases occurring in the corpus (without the structures) as the observed incomplete data. In the training phase, an Expectation Maximization (EM) algorithm (Dempster et al. 77) can be used to estimate the parameters of word modification probabilities by iteratively maximizing the conditional expectation of the likelihood of the complete data given the observed incomplete data and a previous estimate of the parameters. In the parsing phase, a noun phrase is assigned the structure that has the maximum conditional probability given the noun phrase.

Formally, assume that each noun phrase is generated using a word modification structure. For example, "*information retrieval technique*" may be generated using either the structure "$[X_1[X_2X_3]]$" or the structure "$[[X_1X_2]X_3]$". The log likelihood of generating a noun phrase, given the set of noun phrases observed in a corpus $NP = \{np_i\}$ can be written as:

$$L(\phi) = \sum_{np_i \in NP} c(np_i) log \sum_{s_j \in S} P_\phi(np_i, s_j)$$

where, S is the set of all the possible modification structures; $c(np_i)$ is the count of the noun phrase np_i in the corpus; and $P_\phi(np_i, s_j)$ gives the probability of deriving the noun phrase np_i using the modification structure s_j.

With the simplification that generating a noun phrase from a modification structure is the same as generating all the corresponding word modification pairs in the noun phrase and with the assumption that each word modification pair in the noun phrase is generated independently, $P_\phi(np_i, s_j)$ can further be written as

$$P_\phi(np_i, s_j) = P_\phi(s_j) \prod_{(u,v) \in M(np_i, s_j)} P_\phi(u, v)^{c(u,v;np_i,s_j)}$$

where, $M(np_i, s_j)$ is the set of all word pairs (u, v) in np_i such that u modifies (i.e., depends on) v according to s_j.[6] $c(u, v; np_i, s_j)$ is the count of the

[5] Strictly speaking, however, compound noun analysis is a special case of noun phrase analysis, but the same technique can often be used for both.

[6] For example, if np_i is "*information retrieval technique*", and s_j is "$[[X_1X_2]X_3]$", then, $M(np_i, s_j) = \{(information, retrieval), (retrieval, technique)\}$.

modification pairs (u,v) being generated when np_i is derived from s_j. $P_\phi(s_j)$ is the probability of structure s_j; while $P_\phi(u,v)$ is the probability of generating the word pair (u,v) given any word modification relation. $P_\phi(s_j)$ and $P_\phi(u,v)$ are subject to the constraint of summing up to 1 over all modification structures and over all possible word combinations respectively.[7]

The model is clearly a special case of the class of the algebraic language models, in which the probabilities are expressed as polynomials in the parameters(Lafferty 95). For such models, the M-step in the EM algorithm can be carried out exactly, and the parameter update formulas are:

$$\begin{aligned} & P_{n+1}(u,v) \\ & \quad = \lambda_1^{-1} \sum_{np_i \in NP} c(np_i) \sum_{s_j \in S} P_n(s_j|np_i) c(u,v;np_i,s_j) \end{aligned}$$

$$\begin{aligned} & P_{n+1}(s_k) \\ & \quad = \lambda_2^{-1} \sum_{np_i \in NP} c(np_i) P_n(s_k|np_i) \end{aligned}$$

where, λ_1 and λ_2 are the Lagrange multipliers corresponding to the two constraints mentioned above, and are given by the following formulas:

$$\lambda_1 = \sum_{(u,v)\in WP} \sum_{np_i \in NP} c(np_i) \sum_{s_j \in S} P_n(s_j|np_i) c(u,v;np_i,s_j)$$

$$\lambda_2 = \sum_{s_k \in S} \sum_{np_i \in NP} c(np_i) P_n(s_k|np_i)$$

where, WP is the set of all possible word pairs.

$P_n(s_j|np_i)$ can be computed as:

$$\begin{aligned} & P_n(s_j|np_i) \\ & \quad = \frac{P_n(np_i,s_j)}{P_n(np_i)} \\ & \quad = \frac{P_n(np_i,s_j)}{\sum_{s_k \in S} P_n(np_i,s_k)} \\ & \quad = \frac{P_n(s_j) \prod_{u,v \in M(np_i,s_j)} P_n(u,v)^{c(u,v;np_i,s_j)}}{\sum_{s_k \in S}(P_n(s_k) \prod_{u,v \in M(np_i,s_k)} P_n(u,v)^{c(u,v;np_i,s_k)})} \end{aligned}$$

[7]One problem with such simplification is that the model may generate a set of word modification pairs that do not form a noun phrase, although such "illegal noun phrases" are never observed. A better model would be to write the probability of each word modification pair as the conditional probability of the modifier (i.e., the modifying word) given the head (i.e., the word being modified). That is,
$P_\phi(np_i,s_j) =$

$$P_\phi(s_j) P_\phi(h(np_i)|s_j) \prod_{(u,v)\in M(np_i,s_j)} P_\phi(u|v)^{c(u,v;np_i,s_j)}$$

where $h(np_i)$ is the head (i.e., the last word) of the noun phrase np_i(Lafferty 96).

The EM algorithm ensures that $L(n+1)$ is greater than $L(n)$. In other words, every step of parameter update increases the likelihood. Thus, at the time of training, the parser can first randomly initialize the parameters, and then, iteratively update the parameters according to the update formulas until the increase of the likelihood is smaller than some pre-set threshold.[8] In the implementation described here, the maximum length of any noun phrase is limited to six. In practice, this is not a very tight limit, since simple noun phrases with more than six words are quite rare. Summing over all the possible structures for any noun phrase is computed by enumerating all the possible structures with an equal length as the noun phrase. For example, in the case of a three-word noun phrase, only two structures need to be enumerated.

At the time of parsing noun phrases, the structure of any noun phrase np ($S(np)$) is determined by

$$\begin{aligned} S(np) &= argmax_s P(s|np) \\ &= argmax_s P(np|s)P(s) \\ &= argmax_s \prod_{(u,v)\in M(np,s)} P(u,v)P(s) \end{aligned}$$

We found that the parameters may easily be biased owing to data sparseness. For example, the modification structure parameters naturally prefer left association to right association in the case of three-word noun phrases, when the data is sparse. Such bias in the parameters of the modification structure probability will be propagated to the word modification parameters when the parameters are iteratively updated using EM algorithm. In the experiments reported in this paper, an over-simplified solution is adopted. We simply fixed the modification structure parameter and assumed every dependency structure is equally likely.

Fast training is achieved by reading all the noun phrase instances into memory.[9] This forces us to split the whole noun phrase corpus into small chunks for training. In the experiments reported in this paper, we split the corpus into chunks of a size of around 4 megabytes. Each chunk has about 170,000 (or about 100,000 unique) raw multiple word noun phrases. The parameters estimated on each sub-corpus are then merged (averaged). We do not know how much the merging of parameters affects the parameter estimation, but it seems that a majority of phrases are correctly parsed with the merged parameter estimation, based on a rough check of the parsing results. With this approach, it takes a 133-MHz DEC Alpha workstation about 5 hours to train the parser over the noun phrases from a 250-megabyte

[8]For the experiments reported in this paper, the threshold is 2.

[9]An alternative way would be to keep the corpus in the disk. In this way, it is not necessary to split the corpus, unless it is extremely large.

text corpus. Parsing is much faster, taking less than 1 hour to parse all noun phrases in the corpus of a 250-megabyte text. The parsing speed can be scaled up to gigabytes of text, even when the parser needs to be re-trained over the noun phrases in the whole corpus. However, the speed has not taken into account the time required for extracting the noun phrases for training. In the experiments described in the following section, the CLARIT noun phrase extractor is used to extract all the noun phrases from the 250-megabyte text corpus.

After the training on each chunk, the estimation of the parameter of word modifications is smoothed to account for the unseen word modification pairs. Smoothing is made by "dropping" a certain number of parameters that have the least probabilities, taking out the probabilities of the dropped parameters, and evenly distributing these probabilities among all the unseen word pairs as well as those pairs of the dropped parameters. It is unnecessary to keep the dropped parameters after smoothing, thus this method of smoothing helps reduce the memory overload when merging parameters. In the experiments reported in the paper, nearly half of the total number of word pairs seen in the training chunk were dropped. Since, word pairs with the least probabilities generally occur quite rarely in the corpus and usually represent semantically illegal word combinations, dropping such word pairs does not affect the parsing output so significantly as it seems. In fact, it may not affect the parsing decisions for the majority of noun phrases in the corpus at all.

The potential parameter space for the probabilistic model can be extremely large, when the size of the training corpus is getting larger. One solution to this problem is to use a class-based model similar to the one proposed in (Brown et al. 92) or use parameters of conceptual association rather than word association, as discussed in (Lauer 94)(Lauer 95).

4 Experiment Design

We used the CLARIT commercial retrieval system as a retrieval engine to test the effectiveness of different indexing sets. The CLARIT system uses the vector space retrieval model(Salton and McGill 83), in which documents and the query are all represented by a vector of weighted terms (either single words or phrases), and the relevancy judgment is based on the similarity (measured by the cosine measure) between the query vector and any document vector(Evans et al. 93; Evans and Lefferts 95; Evans et al. 96). The experiment procedure is described by Figure 1.

First, the original database is parsed to form different sets of indexing terms (say, using different combination of phrases). Then, each indexing set is passed to the CLARIT retrieval engine as a source document set. The CLARIT system is configured to accept the indexing set we passed as is to ensure that

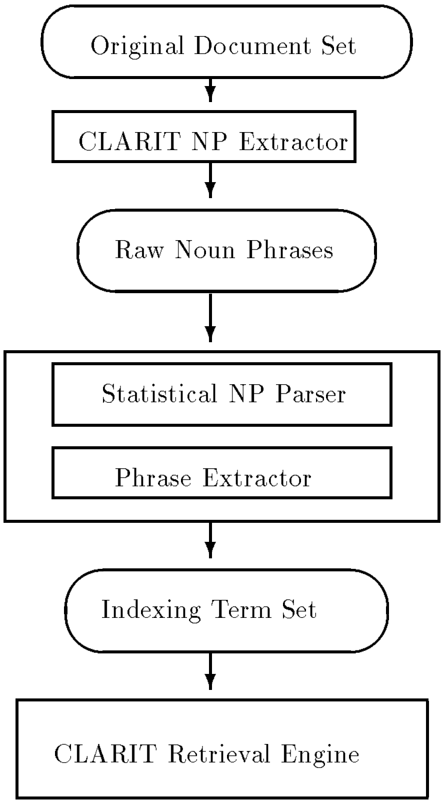

Figure 1: Phrase indexing experiment procedure

the actual indexing terms used inside the CLARIT system are exactly those generated.

It is possible to generate three different kinds/levels of indexing units from a noun phrase: (1) single words; (2) head modifier pairs (i.e., any word pair in the noun phrase that has a linguistic modification relation); and (3) the full noun phrase. For example, from the phrase structure "[[[heavy=construction]=industry]]=group]" (a real example from WSJ90), it is possible to generate the following candidate terms:

```
SINGLE WORDs:
  heavy, construction, industry, group
HEAD MODIFIERS:
  construction industry, industry group,
  heavy construction
FULL NP:
  heavy construction industry group
```

Different combinations of the three kinds of terms can be selected for indexing. In particular, the indexing set formed solely of single words is used as a baseline to test the effect of using phrases. In the experiments reported here, we generated four different combinations of phrases:

```
-- WD-SET:
  single word only (no phrases, baseline)
-- WD-HM-SET:
```

```
  single word + head modifier pair
-- WD-NP-SET:
  single word + full NP
-- WD-HM-NP-SET:
  single word + head modifier + full NP
```

The results from these different phrase sets are discussed in the next section.

5 Results analysis

We used, as our document set, the Wall Street Journal database in Tipster Disk2 (Harman 96) the size of which is about 250 megabytes. We performed the experiments by using the TREC-5 ad hoc topics (i.e., TREC topics 251-300). Each run involves an automatic feedback with the top 10 documents returned from the initial retrieval. The CLARIT automatic feedback is performed by adding terms from a query-specific thesaurus extracted from the top N documents returned from the initial retrieval(Evans and Lefferts 95). The results are evaluated using the standard measures of recall and precision. Recall measures how many of the relevant documents have actually been retrieved. Precision measures how many of the retrieved documents are indeed relevant. They are calculated by the following simple formulas:

$$Recall = \frac{number\ of\ relevant\ items\ retrieved}{total\ number\ of\ relevant\ items\ in\ collection}$$

$$Precision = \frac{number\ of\ relevant\ items\ retrieved}{total\ number\ of\ items\ retrieved}$$

We used the standard TREC evaluation package provided by Cornell University and used the judged-relevant documents from the TREC evaluations as the gold standard(Harman 94).

In Table 1, we give a summary of the results and compare the three phrase combination runs with the corresponding baseline run. In the table, "Ret-rel" means "retrieved-relevant" and refers to the total number of relevant documents retrieved. "Init Prec" means "initial precision" and refers to the highest level of precision over all the points of recall. "Avg Prec" means "average precision" and is the average of all the precision values computed after each new relevant document is retrieved.

It is clear that phrases help both recall and precision when supplementing single words, as can be seen from the improvement of all phrase runs (WD-HM-SET, WD-NP-SET, WD-HM-NP-SET) over the single word run WD-SET.

It can also be seen that when only one kind of phrase (either the full NPs or the head modifiers) is used to supplement the single words, each can lead to a great improvement in precision. However, when we combine the two kinds of phrases, the effect is a greater improvement in recall rather than precision. The fact that each kind of phrase can improve precision significantly when used separately shows that

Experiments	Recall (Ret-Rel)	Init Prec	Avg Prec
WD-SET	0.56(597)	0.4546	0.2208
WD-HM-SET	0.60(638)	0.5162	0.2402
inc over WD-SET	7%	14%	9%
WD-NP-SET	0.58(613)	0.5373	0.2564
inc over WD-SET	4%	18%	16%
WD-HM-NP-SET	0.63(666)	0.4747	0.2285
inc over WD-SET	13%	4%	3%
Total relevant documents: 1064			

Table 1: Effects of Phrases with feedback and TREC-5 topics

these phrases are indeed very useful for indexing. The combination of phrases results in only a smaller precision improvement but causes a much greater increase in recall. This may indicate that more experiments are needed to understand how to combine and weight different phrases effectively.

The same parsing method has also been used to generate phrases from the same data for the CLARIT NLP track experiments in TREC-5(Zhai et al. 97), and similar results were obtained, although the WD-NP-SET was not tested. The results in (Zhai et al. 97) are not identical to the results here, because they are based on two separate training processes. It is possible that different training processes may result in slightly different parameter estimations, because the corpus is arbitrarily segmented into chunks of only roughly 4 megabytes for training, and the chunks actually used in different training processes may vary slightly.

6 Conclusions

Information retrieval provides a good way to quantitatively (although indirectly) evaluate various NLP techniques. We explored the application of a fast statistical noun phrase parser to enhance document indexing in information retrieval. We proposed a new probabilistic model for noun phrase parsing and developed a fast noun phrase parser that can handle relatively large amounts of text efficiently. The effectiveness of enhancing document indexing with the syntactic phrases provided by the noun phrase parser was evaluated on the Wall Street Journal database in Tipster Disk2 using 50 TREC-5 ad hoc topics. Experiment results on this 250-megabyte document collection have shown that using different kinds of syntactic phrases provided by the noun phrase parser to supplement single words for indexing can significantly improve the retrieval performance, which is more encouraging than many early experiments on syntactic phrase indexing. Thus, using selective NLP, such as the noun phrase parsing technique we proposed, is not only feasible for use in information retrieval, but also effective in enhancing the retrieval performance.[10]

[10] Whether such syntactic phrases are more effective than simple statistical phrases (e.g., high frequency word

There are two lines of future work:

First, the results from information retrieval experiments often show variances on different kinds of document collections and different sizes of collections. It is thus desirable to test the noun phrase parsing technique in other and larger collections. More experiments and analyses are also needed to better understand how to more effectively combine different phrases with single words. In addition, it is very important to study how such phrase effects interact with other useful IR techniques such as relevancy feedback, query expansion, and term weighting.

Second, it is desirable to study how the parsing quality (e.g., in terms of the ratio of phrases parsed correctly) would affect the retrieval performance. It is very interesting to try the conditional probability model as mentioned in a footnote in section 3 The improvement of the probabilistic model of noun phrase parsing may result in phrases of higher quality than the phrases produced by the current noun phrase parser. Intuitively, the use of higher quality phrases might enhance document indexing more effectively, but this again needs to be tested.

7 Acknowledgments

The author is especially grateful to David A. Evans for his advising and supporting of this work. Thanks are also due to John Lafferty, Nataša Milić-Frayling, Xiang Tong, and two anonymous reviewers for their useful comments. Naturally, the author alone is responsible for all the errors.

References

Belkin, N., and Croft, B. 1987. Retrieval techniques. In: Williams, Martha E.(Ed.), *Annual Review of Information Science Technology*, Vol. 22. Amsterdam, NL: Elsevier Science Publishers. 1987. 110–145.

Brown, P. et al. 1992. Class-based n-gram models of natural language. *Computational Linguistics*, 18(4), December, 1992. 467–479.

Dempster, A. P. et al. 1977. Maximum likelihood from incomplete data via the EM algorithm. *Journal of the Royal Statistical Society*, 39 B, 1977. 1–38.

Evans, D. A., Ginther-Webster, K. , Hart, M., Lefferts, R., Monarch, I., 1991. Automatic indexing using selective NLP and first-order thesauri. In: A. Lichnerowicz (ed.), *Intelligent Text and Image Handling. Proceedings of a Conference, RIAO '91*. Amsterdam, NL: Elsevier. 1991. pp. 624–644.

Evans, D. A., Lefferts, R. G., Grefenstette, G., Handerson, S. H., Hersh, W. R., and Archbold, A. A. 1993. CLARIT TREC design, experiments, and results. In: Donna K. Harman (ed.), *The First Text REtrieval Conference (TREC-1)*. NIST Special Publication 500-207. Washington, DC: U.S. Government Printing Office, 1993. pp. 251–286; 494–501.

bigrams) remains to be tested.

Evans, David A. and Lefferts, Robert G. 1995. CLARIT–TREC experiments, *Information Processing and Management*, Vol. 31, No. 3, 1995. 385–395.

Evans, D., Milić-Frayling, N., and Lefferts, R. 1996. CLARIT TREC-4 Experiments, in Donna K. Harman (Ed.), *The Fourth Text REtrieval Conference (TREC-4)*. NIST Special Publication 500-236. Washington, DC: U.S. Government Printing Office, 1996. pp. 305–321.

Evans, D. and Zhai, C. 1996. Noun-phrase analysis in unrestricted text for information retrieval. *Proceedings of the 34th Annual meeting of Association for Computational Linguistics*, Santa Cruz, University of California, June 24-28, 1996. 17–24.

Fagan, Joel L. 1987. *Experiments in Automatic Phrase Indexing for Document Retrieval: A Comparison of Syntactic and Non-syntactic methods*, PhD thesis, Dept. of Computer Science, Cornell University, Sept. 1987.

Harman, D. 1994. *The Second Text REtrieval Conference (TREC-2)*, NIST Special publication 500-215. National Institute of Standards and Technology, 1994.

Harman, D. 1996. *TREC 5 Conference Notes*, Nov. 20-22, 1996.

Jelinek, F., Lafferty, J.D., and Mercer, R. L. 1990. Basic methods of probabilistic context free grammars. Yorktown Heights,N.Y.: IBM T.J. Watson Research Center, 1990. Research report RC. 16374.

Lafferty, J. 1995. Notes on the EM Algorithm, Information Theory course notes, Carnegie Mellon University.

Lafferty, J. 1996. Personal Communications.

Lauer, Mark. 1994. Conceptual association for compound noun analysis. *Proceedings of the 32nd Annual Meeting of the Association for Computational Linguistics*, Student Session, Las Cruces, NM, 1994. 337–339.

Lauer, Mark. 1995. Corpus statistics meet with the noun compound: Some empirical results. *Proceedings of the 33th Annual Meeting of the Association for Computational Linguistics*, 1995.

Lewis, D. 1991. *Representation and Learning in Information Retrieval.* Ph.D thesis, COINS Technical Report 91-93, Univ. of Massachusetts, 1991.

Lewis, D. and Sparck Jones, K. 1996. Applications of natural language processing in information retrieval. *Communications of ACM*, Vol. 39, No. 1, 1996, 92–101.

Liberman, M. and Sproat, R. 1992. The stress and structure of modified noun phrases in English. In: Sag, I. and Szabolcsi, A. (Eds.), *Lexical Matters*, CSLI Lecture Notes No. 24. University of Chicago Press, 1992. 131–181.

Marcus, Mitchell. 1980. *A Theory of Syntactic Recognition for Natural Language.* MIT Press, Cambridge, MA, 1980.

Pustejovsky, J., Bergler, S., and Anick, P. 1993. Lexical semantic techniques for corpus analysis. In: *Computational Linguistics*, Vol. 19 (2), Special Issue on Using Large Corpora II, 1993. 331–358.

Resnik, P. and Hearst, M. 1993. Structural ambiguity and conceptual relations. In: *Proceedings of the Workshop on Very Large Corpora*: Academic and Industrial Perspectives, June 22, 1993. Ohio State Un iversity. 58-64.

Salton, G. and McGill, M. 1983. *Introduction to Modern Information Retrieval*, New York, NY: McGraw-Hill, 1983.

Smeaton, Alan F. 1992. Progress in application of natural language processing to information retrieval. *The Computer Journal*, Vol. 35, No. 3, 1992. 268–278.

Strzalkowski, T. 1992. TTP: A fast and robust parser for natural language processing. *Proceedings of the 14th International Conference on Computational Linguistics (COLING)*,Nantes, France, July, 1992. 198–204.

Strzalkowski, T. and Vauthey, B. 1992. Information retrieval using robust natural language processing. *Proceedings of the 30th ACL Meeting*, Neward, DE, June-July, 1992. 104–111.

Strzalkowski, T. and Carballo, J. 1994. Recent developments in natural language text retrieval. In: Harman, D. (Ed.), *The Second Text REtrieval Conference (TREC-2)*, NIST Special Publication 500-215. 1994. 123–136.

Strzalkowski, T. 1995. Natural language information retrieval. *Information Processing and Management.* Vol. 31, No. 3, 1995. 397-417.

Strzalkowski, T. et al. 1995. Natural language information retrieval: TREC-3 report. In: Harman, D. (Ed.), *The Third Text REtrieval Conference (TREC-3)*, NIST Special Publication 500-225. 1995. 39–53.

Strzalkowski, T. et al. 1996. Natural language information retrieval: TREC-4 report. In: Harman, D. (Ed.), *The Fourth Text REtrieval Conference (TREC-4)*. NIST Special Publication 500-236. Washington, DC: U.S. Government Printing Office, 1996. pp. 245–258.

Zhai, C., Tong, X., Milić-Frayling, N., and Evans D. 1997. Evaluation of syntactic phrase indexing – CLARIT TREC5 NLP track report. to appear in *The Fifth Text REtrieval Conference (TREC-5)*, NIST special publication, 1997, forthcoming.