Scaling Up BioNLP: Application of a Text Annotation Architecture to Noun Compound Bracketing

Preslav Nakov, Ariel Schwartz, Brian Wolf
Computer Science Division
University of California, Berkeley
Berkeley, CA 94720
{nakov,sariel}@cs.berkeley.edu

Marti Hearst
SIMS
University of California, Berkeley
Berkeley, CA 94720
hearst@sims.berkeley.edu

Abstract

We describe the use of the Layered Query Language and architecture to acquire statistics for natural language processing applications. We illustrate system's use on the problem of noun compound bracketing using MEDLINE.

1 Introduction

As our natural language processing (NLP) algorithms become ever more successful, we need methods for conveniently re-using their results, both for additional processing, and for end applications such as text mining and information retrieval. We have developed a query language called the *Layered Query Language* (LQL) and a system architecture that supports queries over layers of annotation on natural language text. The model allows for both hierarchical and overlapping layers and for querying at multiple levels of description. The implementation is built on top of a standard RDBMS, and, by using carefully constructed indexes, can execute complex queries efficiently.

We illustrate the use of LQL by applying it to an important language analysis problem for bioscience text: noun compound (NC) bracketing. Consider the phrases *liver cell antibody* and *liver cell line*. Although equivalent at the part of speech (POS) level, they have different syntactic trees. In the former case, an *antibody* targets a *liver cell*, while in the latter we talk about a *cell line* which is derived from the *liver*. The distinction can be represented as a binary tree or, equivalently, as a binary bracketing:

[[liver cell] antibody]	(left bracketing)
[liver [cell line]]	(right bracketing)

There is little prior work on NC bracketing. Best known is that of Lauer (1995) who introduces the probabilistic dependency[1] model for the syntactic disambiguation of NCs and argues against the adjacency[2] model, proposed in (Pustejovsky et al., 1993). Lapata and Keller (2004) propose using the Web as a baseline with an application to six NLP tasks, including the syntactic disambiguation of NCs, and show that variations on bigram counts perform nearly as well as more elaborate methods.

We have extended the Lapata and Keller (2004) work and have developed an algorithm for distinguishing between left and right bracketing by making use of n-gram frequencies, paraphrases and surface features (e.g., dashes and possessive makers) whose counts are derived from Web search engines (Nakov and Hearst, 2005). For example, if the NC is *amino acid sequence*, we query on this term and analyze the results brought back by the search engine. These may include text in which a dash falls between the first two words as in *amino-acid*. If this happens sufficiently frequently, then the NC is likely to have a left bracketing. Similarly, if we find *brain's stem cells*, this suggests a right bracketing for *brain stem cells*, while *brain stem's cells* would favor a left one. A majority vote is used to combine the various features and to determine left versus right bracketing. Currently our approach yields accuracy

[1]The probabilistic dependency model compares $\Pr(w_1|w_3)$ to $\Pr(w_2|w_3)$. Here $\Pr(w_i|w_j)$ means the probability that w_i precedes w_j, for a given w_j.

[2]The adjacency model compares $\Pr(w_1|w_2)$ to $\Pr(w_2|w_3)$.

of 89.34% (Nakov and Hearst, 2005) on Lauer's data set (baseline 66.80%).

However, the use of Web search engines imposes limitations on what kinds of queries we can write, mainly because of the lack of linguistic annotation. For example, if we want to estimate the probability that *health* precedes *care* $\frac{\#(''health\ care'')}{\#(care)}$, we need the frequencies of *"health care"* and *care*, where both words are nouns. The problem is that a query for *care* will return many pages where it is used as a verb, while in *health care* it would nearly always occur as a noun. Even when both *health* and *care* are used as nouns and are adjacent, they may belong to different NPs but sit next to each other only by chance. Furthermore, since search engines ignore punctuation characters, the two nouns may also come from different sentences.

Other Web search engine restrictions prevent querying directly for terms containing hyphens or possessive markers such as *amino-acid sequence* and *protein synthesis' inhibition*. They also disallow querying for a term like *bronchoalveolar lavage (BAL) fluid*, in which the internal parenthesized abbreviation suggests a left bracketing. Finally, search engines cannot support queries that make use of generalized POS information such as

stem cells VERB PREP brain

in which the uppercase patterns stand for any verb and any preposition.

Further, using page hits as a proxy for n-gram frequencies can produce some counter-intuitive results. Consider the bigrams w_1w_4, w_2w_4 and w_3w_4 and a page that contains each bigram exactly once. A search engine will contribute a page count of 1 for w_4 instead of a frequency of 3; thus the page hits for w_4 can be smaller than the page hits for the sum of the individual bigrams. See Keller and Lapata (2003) for more potential problems with page hits.

In the remainder of this paper we describe the use of our Layered Query Language and architecture to acquire statistics for language phenomena that make use of NLP results. These examples include using POS and shallow-parse annotation layers. The system also supports queries on other kinds of annotation layers not shown here, such as gene/protein names, MeSH labels, and abbreviation expansions.

2 LQL and Noun Compound Bracketing

Below we illustrate LQL via some example queries supporting the NC bracketing experiments. A full description of the language syntax and a broader example-based introduction can be found at http://biotext.berkeley.edu/lql/ .

2.1 Selecting a Data Set

Query (1) below creates a set of NCs to work with:

```
FROM
   [layer='shallow_parse' && tag_type='NP'
     ^ [layer='pos' && tag_type="noun"]
       [layer='pos' && tag_type="noun"]
       [layer='pos' && tag_type="noun"] $
   ] AS compound
SELECT compound.content
```

This LQL query looks for a noun phrase (NP) from the shallow-parse layer, containing exactly three nouns from the POS layer. Each layer in an LQL query is enclosed in brackets and is optionally followed by a binding statement. Layers have names, e.g., `pos` and `shallow_parse`, which determine a possible set of types such as `NP` for `shallow_parse` and `NN` for `pos`. There are macros for groups of tags, such as `noun`, which refers to all parts of speech which are nouns in the Penn Treebank. Single quotes are used for exact matches, and double quotes for case insensitive matches and macros.

Enclosure of one layer within another indicates that the outer spans the inner. By default, layers' contents are adjacent; the keywords `ALLOW GAPS` change this default. We use the UNIX delimiters ˆ and $ to constrain the results so that no other words can preceed or follow the three nouns within the NP. Finally, the `select` statement specifies that the contents of the *compound* are to be returned.

Query (1) is case sensitive. While this is useful in the biomedical domain, where the capitalization can distinguish between different genes or proteins, to get frequency counts we want normalization. Query (2) below converts the results to lowercase by using the corresponding SQL function:

```
SELECT LOWER(compound.content) lc, COUNT(*) AS freq
FROM
  BEGIN_LQL
     FROM
        [layer='shallow_parse' && tag_type='NP'
          ^ [layer='pos' && tag_type="noun"]
            [layer='pos' && tag_type="noun"]
            [layer='pos' && tag_type="noun"] $
        ] AS compound
     SELECT compound.content
  END_LQL
GROUP BY lc
ORDER BY freq DESC
```

It also encloses the LQL inside an SQL block and then uses SQL aggregation functions to produce a sorted list (in descending order) of the NCs and their corresponding frequencies. This query will extract 3-noun NCs, provided that the POS tagging and the shallow-parsing annotations are correct. However, because it requires that the NP consists of exactly 3 nouns (without preceding adjectives, determiners etc.), it will omit some 3-noun NCs that are preceded by modifiers and determiners (such as *the*, *a*, *this*).

Query (3) below asserts that the nouns should occupy the three last positions in the NP and disallows other nouns within the same NP, but allows other POS. Note the tradeoff between query complexity and amount of control afforded by the language.

```
SELECT LOWER(compound.content) lc,
       COUNT(*) AS freq
FROM
  BEGIN_LQL
    FROM
      [layer='shallow_parse' && tag_type='NP'
    ^ ( { ALLOW GAPS }
        ![layer='pos' && tag_type="noun"]
         ( [layer='pos' && tag_type="noun"] AS n1
           [layer='pos' && tag_type="noun"] AS n2
           [layer='pos' && tag_type="noun"] AS n3 ) $
      ) $
      ]
    SELECT n1.content, n2.content, n3.content
  END_LQL
GROUP BY lc
ORDER BY freq DESC
```

The new query contains an assertion that a layer does not exist, via the negation operator ("!"). We need the outer brackets to indicate that gaps (intermediate words) are allowed between any layers, thus disallowing the non-existing layer anywhere before the sequence of three nouns (not just when it immediately precedes the first noun). One more pair of brackets counteracts ALLOW GAPS and keeps the three internal nouns adjacent to one another.

2.2 Extracting Features

2.2.1 n-grams

Query (4) below obtains frequencies for bigrams such as *immunodeficiency virus*. It allows for inflections of the second word, respecting the sentence boundaries. It also requires both words to be nouns and to be inside the same NP, thus filtering any false bigrams in which the nouns are adjacent by chance. Note that we did not need to select both words, just one of them, as we are only interested in the total count. Also note the double quotes around the words, which ensure a case insensitive matching.

```
SELECT COUNT(*) AS freq FROM
  BEGIN_LQL
    FROM
      [layer='shallow_parse' && tag_type='NP'
        [layer='pos' && tag_type="noun"
           && content="immunodeficiency"] AS word1
        [layer='pos' && tag_type="noun"
           && (content="virus" || content="viruses")] ]
    SELECT word1.content
  END_LQL
```

Unigram counts are calculated in a similar way.

2.2.2 Paraphrases

Paraphrases are an important feature type for determining NC bracketing. For example, an author describing the concept of *human immunodeficiency virus* may choose to write it in a more expanded manner, such as *immunodeficiency virus in humans*. If this **prepositional paraphrase** occurs often, it suggests that the full NC has a right bracketing, since *immunodeficiency* and *virus* are kept together in the expanded version.

There are NCs whose meaning cannot be expressed with a paraphrase at all, or at least not with a prepositional one (Warren, 1978). In the latter case, one could try a **copula paraphrase** like *immunodeficiency virus that/which is human* (right bracketing), or a **verb paraphrase** such as *virus causing human immunodeficiency* (left) or *immunodeficiency virus found in humans* (right).

Instead of trying to manually decide the correct paraphrases, we can issue queries on the appropriate paraphrase patterns and find out how often each occurs in the corpus. We then add up the number of hits predicting a left versus a right bracketing and compare the counts to make a decision.

Consider the first right-predicting prepositional paraphrase above. It starts with the second word, followed by the third word, followed by a preposition, then by an optional article, and finally by the first word. In Query (5) below, we require the words to have the correct POS and we allow inflections for every word except *immunodeficiency* because it modifies *virus*. We allow any preposition (tag_type='IN') and we count the examples for each one. A question mark is used to express the optionality of the layer corresponding to the determiner.[3] Verbal and copula paraphrases are handled similarly.

[3]This could also be expressed by writing two separate LQL queries: one with and one without the optional layer, which are then connect by the SQL `UNION` operator.

```
SELECT LOWER(prep.content) lp, COUNT(*) AS freq FROM
  BEGIN_LQL
    FROM
      [layer='sentence'
        [layer='pos' && tag_type="noun" &&
          content = "immunodeficiency"]
        [layer='pos' && tag_type="noun" &&
          content IN ("virus","viruses")]
        [layer='pos' && tag_type='IN'] AS prep
       ?[layer='pos' && tag_type='DT' &&
          content IN ("the","a","an")]
        [layer='pos' && tag_type="noun" &&
          content IN ("human", "humans")] ]
    SELECT prep.content
  END_LQL
GROUP BY lp ORDER BY freq DESC
```

3 Evaluation

We experimented on a collection of 1.4 million MEDLINE abstracts, including 10 million sentences (320 million annotations). We translated Query (2) into SQL using our automatic translator, obtaining 418,678 NCs. We manually investigated the most frequent ones, removing those with errors in tokenization (e.g., *transplan* or *tation*), and in POS or shallow-parsing (e.g., *situ hybridization*, without *in*). Two annotators judged the remaining examples as *left*, *right* or *both* (agreement: 88%, kappa .606). We retained the top 232 examples that were unambiguously annotated as either left (193) or right (39), yielding a baseline of 83.19%. We then collected n-gram counts and paraphrases using Queries (4) and (5), plugging the word forms in automatically (using the UMLS *Specialist*[4] lexicon to produce variants). The results are shown in Table 1. In addition to probabilities (Pr), we also tried counts (#) and χ^2, but the prepositional paraphrases were much more accurate: 93.30% (with 83.62% coverage). By combining paraphrases with the χ^2 models in a majority vote, and by assigning the undecided cases to right-bracketing, we achieved 92.24% accuracy.

4 Discussion

Use of LQL is not limited to NC bracketing and can be used for other tasks such as automatic paraphrase acquisition. For example, the most frequent verbal paraphrases for the NC *bone marrow cells* are *cells derived from bone marrow* (22 instances) and *cells isolated from bone marrow* (14 instances). The most frequent prepositional rephrases are *cells in bone marrow* (456 instances) and *cells from bone marrow* (108 instances). All express the correct meaning using different paraphrases.

[4]http://www.nlm.nih.gov/pubs/factsheets/umlslex.html

Model	$\surd$	$\times$	$\emptyset$	**P**(%)	**C**(%)
# adjacency	196	36	0	84.48	100.00
Pr adjacency	173	59	0	74.57	100.00
χ^2 adjacency	200	32	0	86.21	100.00
# dependency	195	37	0	84.05	100.00
Pr dependency	193	39	0	83.19	100.00
χ^2 dependency	196	36	0	84.48	100.00
PrepPar	181	13	38	93.30	83.62
PP+χ^2adj+χ^2dep	207	13	12	94.09	94.83
PP+χ^2adj+χ^2dep$\rightarrow$right	214	18	0	**92.24**	100.00
Baseline (choose left)	193	39	0	83.19	100.00

Table 1: **Bracketing results.** Shown are the numbers for correct ($\surd$), incorrect ($\times$), and no prediction ($\emptyset$), followed by precision (P, calculated over $\surd$ and $\times$ only) and coverage (C, % examples with prediction). "$\rightarrow$ right" means assigning right in case of $\emptyset$, and "+" means a majority vote combination.

Other linguistic annotations may be potentially useful: we can construct a set of "hard" examples for NC bracketing by writing an LQL query that looks for 3-noun NCs $w_1w_2w_3$ such that both w_1w_2 and w_2w_3 are MeSH terms, e.g., *stem cell line*.

Acknowledgements We thank Janice Hamer for helping us with the annotations and examples. This research was supported by NSF DBI-0317510, and a gift from Genentech.

References

Frank Keller and Mirella Lapata. 2003. Using the Web to obtain frequencies for unseen bigrams. *Computational Linguistics*, 29:459–484.

Mirella Lapata and Frank Keller. 2004. The Web as a baseline: Evaluating the performance of unsupervised Web-based models for a range of NLP tasks. In *Proceedings of HTL-NAACL*, pages 121–128, Boston.

Mark Lauer. 1995. *Designing Statistical Language Learners: Experiments on Noun Compounds*. Ph.D. thesis, Department of Computing Macquarie University NSW 2109 Australia.

Preslav Nakov and Marti Hearst. 2005. Search engine linguistics beyond the n-gram: Application to noun compound bracketing. In *Proceedings of the 9th coNLL.*

James Pustejovsky, Peter Anick, and Sabine Bergler. 1993. Lexical semantic techniques for corpus analysis. *Computational Linguistics*, 19(2):331–358.

Beatrice Warren. 1978. Semantic patterns of noun-noun compounds. In *Gothenburg Studies in English 41, Goteburg, Acta Universtatis Gothoburgensis*.