FINDING APPROXIMATE LANGUAGE PATTERNS

Samuel W. K. Chan
Dept. of Decision Sciences, The Chinese University of Hong Kong, Hong Kong

Keywords: Bio-inspired language parsing, semantic labelling.

Abstract: A two-phase annotation method for semantic labeling based on the edit distance is proposed. This dynamic programming approach stresses on a non-exact string matching technique that takes full advantage of the underlying grammatical structure of 65,000 parse trees in a Treebank. It is based on the assumption that human language understanding is relevant to concrete past language experiences rather than any abstract linguistic rules. This shallow technique is inspired by the research in the area of bio-molecular sequences analysis which advocates high sequence similarity usually implies significant function or structural similarity. Experimental results for recognizing various labels in 10,000 sentences are used to justify its significances.

1 INTRODUCTION

Automatic information extraction has received a great deal of attention in the latest development of information retrieval. While a plethora of issues relating to questions of accuracy and efficiency have been thoroughly discussed, the problem of extracting meaning from natural language has scarcely been addressed. When the size and quantity of documents available on the Internet are considered, the demand for a highly efficient system that identifies the semantic meaning is clear. Case frame is one of the most important structures that are used to represent the meaning of sentences (Fillmore, 1968). One could consider a case frame to be a special, or distinguishing, form of knowledge structure about sentences. Although several criteria for recognizing case frames in sentences have been considered in the past, none of the criteria serves as a completely adequate decision procedure. Most of the studies in natural language processing (NLP) do not provide any hints on how to map input sentences into case frames automatically. As a result, both the efficiency and robustness of the techniques used in information extraction is highly in doubt when they are applied to real world applications.

The objective of this research is twofold. First, a shallow but effective sentence chunking process is developed. The process is to extract all the phrases from the input sentences, without being bogged down into deep semantic parsing and understanding. Second, a novel semantic labeling technique that is based on the syntactic and semantic tags of the latest Treebank is being constructed (CKIP, 2004). One of our primary goals in this research is to design a shallow but robust mechanism which can annotate sentences using a set of semantic labels. While the classical syntactic and semantic analysis is extremely difficult, if not impossible, to systematize the current research in NLP, our approach does not require any deep linguistic analysis to be formalized. The annotation will provide piecemeal the underlying semantic labels of the sentence. The organization of the paper is as follows. The related work in semantic labeling and sentence chunking are first described in Section 2. In this research, each word in sentences has two attributes, i.e. part-of-speech (POS) and semantic classes (SC). Any input sentence is first transformed into a feature-enhanced string. A two-phase feature-enhanced string matching algorithm which is based on the edit distance is devised. Section 3 shows how the algorithm can be applied in the semantic labeling using 65,000 parse trees in a Treebank. The system has already been implemented using Java language. In order to demonstrate the capability of our system, an experiment with 10,000 sentences is conducted. A detailed evaluation is explained in Section 4 followed by a conclusion.

W. K. Chan S. (2008).
FINDING APPROXIMATE LANGUAGE PATTERNS.
In *Proceedings of the First International Conference on Bio-inspired Systems and Signal Processing*, pages 295-301
DOI: 10.5220/0001063802950301

2 RELATED WORK

Following the framework of case grammar which is originally proposed by Fillmore in 1968, it has been accepted that every nominal constituent in every language bears a single syntactic–semantic case relation (Jackendoff, 1983; Dowty, 1991). In earlier systems, Somers describes a prototype computer program that attempts to map surface strings of English onto a formalism representing one level of a deep structure (Somers, 1982). Weischedel *et al*. (1993) predict the intended interpretation of an utterance when more than one interpretation satisfies all known syntactic and semantic constraints, and ascertain its semantic labels. It is on the basis that semantic features inherent in the main verb of a sentence can be used to infer the potential semantic labels of the sentence. Utsuro *et al*. (1993) describe a method for acquiring surface semantic labels of Japanese verbs from bilingual corpora. They make use of translation examples in two distinct languages that have quite different syntactic structures and word meanings. Similarly, Kurohashi and Nagao (1994) have developed a powerful parser for Japanese sentences based on the case frames encoded in a verb dictionary. The dictionary contains some typical example sentences for each case frame. The dictionary then tags the proper case frame for an input sentence based on the sentence similarities.

Any high level language understanding process, such as semantic labeling, must involve chunking sentences into segments. Motivated by the psycholinguistic evidence which demonstrates that intonation changes or pauses would affect the language understanding processes in humans (Gee & Grosjean, 1983), Abney (1991) proposes the concept of text chunking as a first step in the full parsing. A typical chunk of a text is defined as consisting of a single content word surrounded by a constellation of function words, matching a fixed template. Church also uses a simple model for finding base non-recursive NPs in sequence of POS tags (Church, 1988). Turning sentence chunking into a bracketing problem, Church calculates the probability of inserting both the open and close brackets between POS tags. Each chunking alternative is ranked and the best alternative is selected. Using transformation-based learning with rule-template referring to neighboring words, POS tags and chunk tags, Ramshaw & Marcus (1995) identify essentially the initial portions of non-recursive noun phrases up to the head, including determiners. These chunks are extracted from the Treebank parses, by selecting NPs that contain no nested NPs. While the above approaches have been proposed to recognize common subsequences and to produce some forms of chunked representation of an input sentence, the recognized structures do not include any recursively embedded NPs. As the result, the resultant fragments bear little resemblance to the kind of phrase structures that normally appear in our languages.

While it may be too computationally demanding to have a full syntactic and semantic analysis of every sentence in every text, Sima'an (2000) presents a Tree-gram model which integrates bilexical dependencies, and conditions its substitutions based on the structural relations of the trees that are involved. The Tree-gram model is a typical example of data-oriented parsing (DOP) advocated by Bod et al. (2003). The basic ideas of the Tree-gram model are to (i) take a corpus of utterances annotated with labeled trees; (ii) decompose every corpus tree into the bag of all its subtrees; (iii) treat the union of all these subtree bags as a stochastic tree substitution grammar, where the substitution probability of each subtree is estimated as the relative frequency of this subtree among the subtrees with the same root label. Inspired by the Tree-gram model, in this research, we propose a mechanism in shallow semantic labeling as well as sentence chunking by matching any input sentence with the trees in a Treebank through a two-phase feature-enhanced string matching. Different from the stochastic tree substitution grammar proposed in the Tree-gram model, our approach, characterized by an optimization technique, looks for a transformation with a minimum cost, or called edit distance. While the concept of edit distance is commonly found in the conventional pattern matching techniques (Gusfield, 1997; Tsay & Tsai, 1989), we take a step further in applying the technique in shallow semantic labeling. The detailed discussion of the algorithm is shown as follows.

3 TWO-PHASE FEATURE-ENHANCED STRING MATCHING ALGORITHM

Our labeling is defined as a two-phase feature-enhanced string matching using the edit operations. For every input sentence, a coarse-grained syntactic matching is conducted in our first phase of matching. The matching relies on a set of coarse-grained but global part-of-speech (POS) tags. The major objective of this phase is to shortlist all the potential trees among 65,000 parse trees in the CKIP Treebank, which are relevant to the input sentence,

without getting bogged down into computational complexity with other linguistic details. The second phase of the matching is followed to compute the dissimilarity measure between the input sentence and every short-listed candidate that is identified in the first phase. Detailed POS and semantic class (SC) tags will be employed. As a result, a candidate tree which has the minimum dissimilarity with the input sentence will be identified. The underlying semantic labels and phrases of the candidate tree are used to determine the shallow language patterns of the input sentence. The details of the two-phase matching are explained in the following.

3.1 Coarse-Grained Syntactic Matching

In the first phase of matching, each word is represented by its corresponding POS. Let S be an input sentence and the T be a tree in a Treebank, s_i and t_j be two tokens in S and T with attribute POS_i and POS_j respectively. We define the cost function for the *change* operation in the traditional edit operations (Wagner & Fischer, 1974) $s_i \rightarrow t_j$ to be

$$R(s_i \rightarrow t_j) = u(POS_i, POS_j) \quad (1)$$

where $u(POS_i, POS_j)$ defines the cost due to the difference between the POS of the two tokens. The POS tags from the Chinese Knowledge Information Processing Group (CKIP) of Academia Sinica are employed (Chen *et al*., 1996). The tags are subdivided into 46 major POS classes which are further refined into more than 150 subtypes. However, in this coarse-grained matching, only the major POS classes will be considered. To figure out the cost function $u(\cdot,\cdot)$ in the coarse-grained matching, all the major POS tags are organized into a hierarchical structure with an associated hard-coded cost function. Figure 1 shows the structure of notional words and describes the relative distances between the adjectives (`A`), verbs (`V`), status-verbs (`VH`), measure-words (`Nf`), nouns (`N`), position-words (`Ng`), time-words (`Nd`) and place-words (`Nc`). All notional words have definite meanings in the language. The cost function is based on their interchangeability, the degree of flexibility in placement in the syntax, and the similarity of their acceptable modifiers. For example, Chinese verbs and adjectives share a lot of common features syntactically, i.e. both can be predicates or modified by adverbs and the word, *not*. All these features fail to appear in nouns. The abbreviations in bracket indicate the original POS tags marked by the CKIP. The corresponding tree structure of the XML is shown in Figure 2.

```
<Head toll="5">
  <NodeB toll="2">
    <NodeC toll="2">
      <Adjective toll="5"/>
      <Verb toll="5"/>
    </NodeC>
    <Status-Verb toll="7"/>
  </NodeB>
  <NodeD toll="2">
    <Measure-Word toll="7"/>
    <NodeE toll="2">
      <Noun toll="5"/>
      <NodeF toll="2">
        <Position-word toll="3"/>
        <NodeG toll="1">
          <Time-word toll="2"/>
          <Place-word toll="2"/>
        ...
</Head>
```

Figure 1: XML illustrating the relative distances between 8 different types of POS.

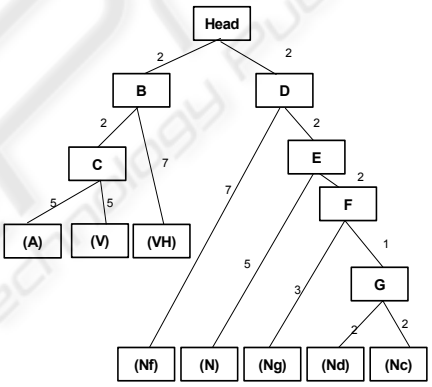

Figure 2: Corresponding tree structure of the XML shown in Figure1.

The cost function $u(\cdot,\cdot)$ reflects the difference based on the tag `toll` encoded in the XML as shown in Figure 1. The function also indicates the degree of alignment between the syntactic structure of the input sentence and the trees in the Treebank. Although two feature-enhanced strings with the same POS sequence do not imply they will share the same syntactic structure, this coarse-grained syntactic matching shortlists the potential trees by imposing a necessary, even not sufficient, constraint on its syntactic structure and limits the potential search space in the subsequent stage of semantic matching.

3.2 Computation of Semantic Dissimilarity

What this second phase matching basically does is to make a detailed comparison between the input sentence and the short-listed trees in its earlier stage. In this phase, each Chinese token has two attributes, i.e. a detailed part-of-speech (*POS*) and semantic class (*SC*). Similar to the approach in Section 3.1, we define the cost function for the *change* operation $s_i \rightarrow t_j$ to be

$$R(s_i \rightarrow t_j) = f(u(POS_i, POS_j), v(SC_i, SC_j)) \qquad (2)$$

where the function f is the dissimilarity function relied on two major components. The first component $u(POS_i, POS_j)$ defines the partial cost due to the difference between the detailed POS of the words. The detailed POS tags are organized in XML format, similar to the approach demonstrated in Figure 1. Figure 3 shows the further breakdown of the nouns (`Na`) which is divided into in-collective (`Nae`) and collective (`Na1`) nouns. The collective nouns are then subdivided into in-collective concrete uncountable nouns (`Naa`), in-collective concrete countable nouns (`Nab`), in-collective abstract countable nouns (`Nac`), in-collective abstract uncountable nouns (`Nad`). The figure associated with the arcs in the Figure 3 illustrates the cost function.

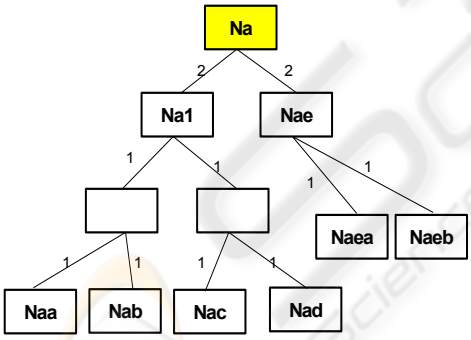

Figure 3: Tree structure of Nouns (`Na`) based on the CKIP Academia Sinica.

The second term in Eqn. (2) defines another partial cost due to the semantic differences. In our approach, the words in the input sentences and the trees are identified using a bilingual thesaurus similar to the Roget's Thesaurus. The *is-a* hierarchy in the bilingual thesaurus, shown the underlying ontology, can be viewed as a directed acyclic graph with a single root. Figure 4 shows one of the *is-a* hierarchies in the thesaurus using our Tree Editor. While the upward links correspond to generalization, the specialization is represented in the downward links. The hierarchies demonstrated in the thesaurus are based on the idea that linguists classify lexical items in terms of similarities and differences. They are used to structure or rank lexical items from more general to the more special.

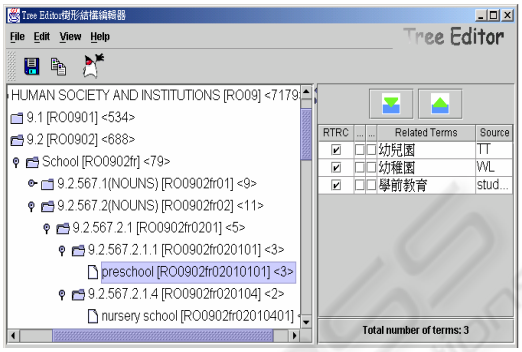

Figure 4: *is-a* hierarchy in the bilingual thesaurus.

Based on the *is-a* hierarchy in the thesaurus, we define the conceptual distance d between two notional words by their shortest path lengths. Given two words t_1 and t_2 in an *is-a* hierarchy of the thesaurus, the semantic distance d between the tokens is defined as follows:

$$d(t_1, t_2) = \text{minimal number of } \textit{is-a} \text{ relationships in the shortest path between } t_1 \text{ and } t_2 \qquad (3)$$

The shortest path lengths in *is-a* hierarchies are calculated. Initially, a search fans out through the *is-a* relationships from the original two nodes to all nodes pointed to by the originals, until a point of intersection is found. The paths from the original two nodes are concatenated to form a continuous path, which must be a shortest path between the originals. The number of links in the shortest path is counted. Since $d(t_1, t_2)$ is positive and symmetric, $d(t_1, t_2)$ is a metric which means (i) $d(t_1, t_1) = 0$; (ii) $d(t_1, t_2) = d(t_2, t_1)$; (iii) $d(t_1, t_2) + d(t_2, t_3) \geq d(t_1, t_3)$. At the same time, the semantic similarity measure between the items is defined by:

$$v(t_i, t_j) := \begin{cases} d(t_i, t_j) & \text{if } d(t_i, t_j) \leq d_{max} \\ MaxInt & \text{otherwise} \end{cases} \qquad (4)$$

where d_{max} is proportional to the number of lexical items in the system and *MaxInt* is a maximum integer of the system. This semantic similarity measure defines the degree of relatedness between the words. Obviously, strong degree of relatedness exists between the lexical tokens under the same nodes. On the other hand, for the cost of the *insert* and *delete* operations, we make use the concept of *collocation* that measures how likely two words are to co-occur in a window of text. To better

distinguish statistics based ratios, work in this area is often presented in terms of the mutual information (MI), which is defined as

$$MI(t_{j-1}, t_j) = \log_2 \frac{P(t_{j-1}, t_j)}{P(t_{j-1}) \times P(t_j)} \quad (5)$$

where t_{j-1} and t_j are two adjacent words. While $P(x, y)$ is the probability of observing x and y together, $P(x)$ and $P(y)$ are the probabilities of observing x and y anywhere in the text, whether individually or in conjunction. Note that tokens that have no association with each other and co-occur together according to chance will have a MI value close to zero. This leads to the cost function for insertion and deletion shown in Eqns. (6) and (7) respectively.

$$R(\lambda \to t_j) = \begin{cases} k \times e^{-z} & \text{if } z > \varepsilon > 0 \\ MaxInt & \text{otherwise} \end{cases} \quad (6)$$

where $z = \min \{MI(t_{j-1}, t_j), MI(t_j, t_{j+1})\}$

$$R(t_j \to \lambda) = \begin{cases} l \times e^{-MI(t_{j-1}, t_{j+1})} & \text{if } MI(t_{j-1}, t_{j+1}) > \varepsilon > 0 \\ MaxInt & \text{otherwise} \end{cases} \quad (7)$$

where k, l, ε are three constants relied on the size of the active corpus.

Obviously, the insertion operation will be penalized if the co-occurrence between the newly inserted word and its neighbors is low. Similarly, the deletion operation is most likely to happen if there is a high co-occurrence between the adjacent pairs after the deletion. Using the above cost functions for the three types of edit operations, the tree in the Treebank with minimum cost is being chosen to be the best approximation of the input sentence and its associated semantic labels will be adopted. Shallow language patterns are then extracted based on the recursive structures and semantic labels appeared in the Treebank. The experimental results of the semantic labeling are shown in the section below.

4 EXPERIMENTAL RESULTS

As mentioned in Eqn. (2), several approaches have been used to define the dissimilarity function f by combining the semantic differences and the detailed POS tags in our second phase feature-enhanced string matching. In our evaluations, five different types of dissimilarity function f are applied. They are

(i) $f_1(u, v) = u(POS_i, POS_j)$
(ii) $f_2(u, v) = v(SC_i, SC_j)$
(iii) $f_3(u, v) = u(POS_i, POS_j) + v(SC_i, SC_j)$
(iv) $f_4(u, v) = \min (u(POS_i, POS_j), v(SC_i, SC_j))$
(v) $f_5(u, v) = \max (u(POS_i, POS_j), v(SC_i, SC_j))$

Dissimilarity function $f_1(u, v)$ provides a detailed version of our coarse-grained syntactic matching. Detailed POS tags are used as the dissimilarity measure in the labeling. Similarly, $f_2(u, v)$ considers only the semantic class of the words. The other three combine both syntactic and semantic features in defining the dissimilarity measures. We have tested our shallow semantic labeling with 10,000 sentences with the Treebank. Since this research is concerning with shallow semantic labeling, we have no incentive to match the trees/subtrees in the Treebank with very complicated structures. The average sentence length is around 13.7 characters per sentence. Table 1 summarizes the results of our system evaluation. The third and fourth columns in the table are number of sentences in each range of edit distance and their average edit distances. The edit distance is defined as a minimum cost in transforming the input sentence with the closest sentence pattern in the Treebank. In other words, the smaller the distance, the higher similarity they have.

Table 1: Sentence analysis in the experiment. Edit distance is defined as a minimum cost in transforming the input sentence with the closest sentence pattern in the Treebank. The smaller the distance, the higher similarity they have.

Dissimilarity function f	Range of Edit distance	% of sentences	Average edit distance
$f_1(u, v)$	0-25	13.9	19.2
	26-50	16.3	40.5
	51-75	19.7	63.6
	76-100	27.9	89.6
	101-150	22.2	124.9
$f_2(u, v)$	0-25	11.3	19.3
	26-50	15.6	41.4
	51-75	17.7	65.2
	76-100	29.3	91.8
	101-150	26.1	125.7
$f_3(u, v)$	0-25	24.1	17.9
	26-50	31.6	38.2
	51-75	22.7	62.3
	76-100	12.8	85.5
	101-150	8.8	121.4
$f_4(u, v)$	0-25	18.6	19.1
	26-50	19.1	41.3
	51-75	30.2	64.7
	76-100	14.7	88.6
	101-150	17.4	124.2
$f_5(u, v)$	0-25	20.5	19.6
	26-50	22.4	40.9
	51-75	26.9	58.2
	76-100	15.3	87.9
	101-150	14.9	128.4

If it is considered as a good match where the edit distances are equal to or less than 50, then it can be observed, in Table 1, that the dissimilarity functions f_3, f_5 and f_4 all produce higher percentage of sentences with lower edit distance. This reflects both the information from syntactic tags and semantic classes provide useful clues in our shallow semantic labeling. Our experiments are not conducted with perfect information. It is worthwhile to mention that more than 530 sentences have incomplete information which mainly comes from proper nouns, or out-of-vocabulary (OOV) words. Both of them have neither defined POS nor semantic class. All these information will be annotated with a default value which will certainly induce errors in our labeling. While it is inevitable to have OOV words in any real corpus, the performance, due to the coverage of POS and semantic classes, does not deteriorate much in our system. The labeling is still feasible over the sentences with OOV words. This tolerance ability provides the graceful degradation in our shallow semantic labeling. While other systems are brittle and working only in all-or-none basis, the robustness of our system is guaranteed. At the same time, while real text tends to have grammatical mistakes and error-prone, these problems can be tackled with an acceptable tolerance in our system.

In our second evaluation, we have tested our algorithm in recognizing several major semantic labels that appear in our sentences. The semantic labels include `theme`, `goal`, `property`, `range`, `agent`, `predication`, `location`, `time`. As with other text analysis, the effectiveness of the system appears to be dictated by recall and precision parameters where recall (R) is a percentage of how many correct labels can be identified while precision (P) is the percentage of labels, tackled by our system, which are actually correct. In addition, a common parameter F is used as a single-figure measure of performance which combines recall (R) and precision (P) as in follows,

$$F = \frac{(\beta^2 + 1) \times P \times R}{\beta^2 \times P + R} \quad (8)$$

We set $\beta = 1$ to give no special preference to either recall or precision. The recall, precision and F-score for the semantic labels in dissimilarity function f_3 are shown in Table 2.

As shown in the last row in Table 2, the precision and recall of all semantic labels are calculated by considering all the semantic labels that appear in the sentences, rather than by averaging the measures for individual semantic labels. It is worth noting that the greatest differences in performance are the recall while the precision remains relatively steady in most semantic labels.

Table 2: Evaluation of different semantic labels in the dissimilarity function f_3. Brackets show the results obtained in the derivation subtrees.

	Elementary Subtree (Derivation		
Semantic Label	Recall	Precision	F-score
`theme`	0.79 (0.88)	0.82 (0.85)	0.805
`goal`	0.80 (0.78)	0.79 (0.76)	0.795
`property`	0.89 (0.78)	0.91 (0.83)	0.900
`range`	0.94 (0.93)	0.92 (0.91)	0.930
`agent`	0.92 (0.92)	0.87 (0.85)	0.894
`predication`	0.76 (0.80)	0.81 (0.78)	0.784
`location`	0.92 (0.92)	0.91 (0.89)	0.915
`Time`	0.93 (0.93)	0.95 (0.94)	0.940
`experiencer`	0.87 (0.89)	0.86 (0.88)	0.865
`manner`	0.79 (0.85)	0.84 (0.83)	0.814
`possessor`	0.91 (0.93)	0.88 (0.89)	0.895
`condition`	0.80 (0.84)	0.82 (0.81)	0.810
`all labels`	0.88 (0.84)	0.89 (0.88)	0.885

One possible explanation is that the low recall rates in some labels are due to less complete coverage of linguistic phenomena. In addition, we define an elementary subtree that spans only on a sequence of words, as well as a derivation subtree that contains at least one branch of elementary subtree. It may be expected the F-score of the derivation subtrees will be much worse than its counterpart, however, Table 2 shows surprisingly the differences in the overall accuracy in two main types of subtrees are not significant. An explanation is that we have approached chunking as well as assigning the most salient semantic label to the chunks based on the POS and semantic tags. Even though there may be some misclassification in the terminal nodes, this will not hinder the system to tag the semantic labels in the longer chunks. In other words, the longer chunks are less error prone in our semantic labeling. This shallow semantic labeling technique produces an output that abstract away the details but retains the core semantic structure of the actual sentence. Pure linguistic theories may well have solutions to the semantic labeling that tends to be highly theory-specific with less emphasis on real text. Our shallow approach does not focus on how well it explains various structural and interpretive phenomena in linguistics perceptive, but on how well it predicts the semantic label of sentences. It aims at theory-neutral annotation and derives linguistically-plausible semantic labels or short phrase structures using a Treebank. We have suggested that semantic labels can be detected by grouping sequences of words that occur together more often with high mutual information. While the approach has been

implemented successfully in the Chinese language as illustrated in our evaluation, the idea delineated certainly does not limit or tailor-made for any particular language. Only a minor modification is needed to apply the technique to other languages.

5 CONCLUSIONS

In this paper, we have illustrated a shallow technique in which semantic labels are extracted in forms of chunks of phrases or words using a two-phase feature-enhanced string matching algorithm. While the first phase is to shortlist the potential trees in the Treebank, chunks are further tagged with semantic labels in the second phase. Based on the linguist's conception of phrase structure, our approach does not require a full syntactic parse to pursue semantic analysis and the recursively embedded phrases can also be identified without pain. This shallow technique is inspired by the research in the area of bio-molecular sequences analysis which advocates *high sequence similarity usually implies significant function or structural similarity*. It is characteristic of biological systems that objects have a certain form that has arisen by evolution from related objects of similar but not identical form. This *sequence-to-structure* mapping is a tractable, though partly heuristic, way to search for functional or structural universality in biological systems. With the support from the results as shown, we conjecture this sequence-to-structure phenomenon appears in our sentences. The sentence sequence encodes and reflects the more complex linguistic structures and mechanisms described by linguists. While our system does not claim to deal with all aspects of language, we suggest an alternate, but plausible, way to handle the real corpus.

ACKNOWLEDGEMENTS

The work described in this paper was partially supported by the grants from the Research Grants Council of the Hong Kong Special Administrative Region, China (Project Nos. CUHK4438/04H and CUHK4706/05H).

REFERENCES

Abney, S. (1991). Parsing by chunks. In Berwick, R., Abney, S. & Tenny, C. (Eds.), *Principle-Based Parsing*. Kluwer Academic.

Bod, R., Scha, R., & Sima'an, K. (2003). *Data-Oriented Parsing*. Stanford: California, CSLI.

Chen, K.-J., Huang, C.-R., Chang, L.-P., & Hsu. H.-L. (1996). Sinica Corpus: Design Methodology for Balanced Corpora. *Proceedings of the 11th Pacific Asia Conference on Language, Information, and Computation (PACLIC II)*, Seoul Korea, 167-176.

Church, K. (1988). A stochastic parts program and noun phrase parser for unrestricted text. *Proceedings of Second Conference on Applied Natural Language Processing*, Austin, Texas.

CKIP (2004). *Sinica Chinese Treebank: An Introduction of Design Methodology*. Academic Sinica.

Dowty, D. (1991). Thematic proto-roles and argument selection. *Language*, 67, 547-619.

Fillmore, C.J. (1968). The case for case. In E. Bach & R.T. Harms (Eds.), *Universals in Linguistic Theory*, 1-90. Holt, Rinehart & Winston.

Gee, J., & Grosjean, F. (1983). Performance structures: A psycholinguistic and linguistic appraisal. *Cognitive Psychology*, 15, 4, 411-458.

Gusfield, D. (1997). *Algorithms on Strings, Trees, and Sequences: Computer Science and Computational Biology*. Cambridge University Press.

Jackendoff, R. (1983). *Semantics and Cognition*. MIT Press.

Kurohashi, S., and Nagao, M. (1994). A method of case structure analysis for Japanese sentences based on examples in case frame dictionary. *IEICE Transactions on Information and Systems*, vol. E77-D, no. 2, 227-239.

Ramshaw, L. A., & Marcus, M.P. (1995). Text chunking using transformation-based learning. *Proceedings of the Third Workshop on Very Large Corpora*, 82-94.

Sima'an, K. (2000). Tree-gram parsing: lexical dependencies and structural relations. *Proceedings of the 38th Annual Meeting of the Association for Computational Linguistics*, 53-60, Hong Kong.

Somers, H.L. (1982). The use of verb features in arriving at a 'meaning representation'. *Linguistics*, 20, 237-265.

Tsay, Y.T., & Tsai, W.H. (1989). Model-guided attributed string matching by split-and-merge for shape recognition. *International Journal of Pattern Recognition and Artificial Intelligence*, 3, 2, 159-179.

Utsuro, T., Matsumoto, Y., and Nagao, M. (1993). Verbal case frame acquisition from bilingual corpora. *Proceedings of the Thirteenth International Joint Conference on Artificial Intelligence*, vol. 2, 1150-1156.

Wagner, R.A., & Fischer, M.J. (1974). The string-to-string correction problem. *Journal of the Association for Computing Machinery*, 21, 1, 168-173.

Weischedel, R., Meteer, M., Schwartz, R., Ramshaw, L., Palmucci, J. (1993). Coping with ambiguity and unknown words through probabilistic models. *Computational Linguistics*, 19, 2, 359-382.