iSimp: A Sentence Simplification System for Biomedical Text

Yifan Peng[†*], Catalina O. Tudor[†‡*], Manabu Torii[†‡], Cathy H. Wu[†‡], K. Vijay-Shanker[†]
[†]*Computer & Information Sciences*
[‡]*Center for Bioinformatics & Computational Biology*
University of Delaware, Newark, DE
Email: {ypeng,tudor,torii,wuc,vijay}@cis.udel.edu

***Abstract*—Text mining applications using natural language processing are often confronted with long and complicated sentences. This is observed particularly in the abstracts of scientific articles where authors summarize, in few sentences, the various facts described throughout the manuscript. Being rich in novel and important information, the abstract has been the primary target of biomedical text mining applications. In this work, we aim to simplify complex sentences in abstracts of biomedical text so that they can be readily processed by text mining applications. We focus on syntactic constructs that are frequently encountered in the biomedical literature, such as coordinations, relative clauses, and appositions, with emphasis on their boundary detection. Our approach yielded good detection performance (average F-measure between 86.5% and 92.7%), and aided in improving biomedical text mining applications, RLIMS-P and $Rank_{\text{Pref}}$.**

***Keywords*-sentence simplification, information extraction, natural language processing, text mining**

I. Introduction

With the rapid growth in the number of scientific publications, it has become increasingly difficult for scientists to manually find information published in the literature. Therefore, text mining and information extraction systems have been developed to automatically extract information of particular interest to scientists.

Many of these tools detect the information in text if it fits some common patterns reliably. For example, if the task is to detect phosphorylation information (e.g., kinase/substrate), we might look for sentences that are written in the form of "X phosphorylates Y", as shown in Ex.1.

> It was suggested that **Yak1** phosphorylates **Crf1** to promote its nuclear entry. (Ex.1)

This sentence mentions the phosphorylation in a format that is easy to process by text mining systems. However, we also see sentences containing complex grammatical structures. In Ex.2, the kinase/substrate pairs are [Raf-2, MEK1], [MEK1, ERK1], and [MEK1, ERK2].

> Active **Raf-2** phosphorylates and activates **MEK1**, which phosphorylates and activates the MAP kinases signal regulated kinases, **ERK1** and **ERK2**. (Ex.2)

Humans can easily grasp the phosphorylation information in this sentence, skipping over grammatical complexities and focusing on the phosphorylation information alone. But an automated text mining system may not identify [MEK1, ERK2] as a kinase/substrate pair without complex rules and patterns. However, designing rules and patterns for all possible syntactic variations is next to impossible, because sentence constructions and writing styles vary considerably from one author to another, and one publication to another.

This paper proposes an alternative approach to detect and extract information from complex sentences. Instead of matching all possible variations in text, we propose to simplify complex sentences first, and then attempt to match ordinary patterns in the simplified sentences. The hypothesis is that sentence simplification can alleviate the problems of text mining tools when dealing with complexities [1,2,3]. For example, after identifying all constructs in Ex.2 (see Figure 1), simplified sentences Ex.3a–Ex.3c can be generated. It is now possible to extract the kinase/substrate pairs from the simplified sentences, by using the simple pattern "X phosphorylates Y".

- Active **Raf-1** phosphorylates **MEK1**. (Ex.3a)
- **MEK1** phosphorylates **ERK1**. (Ex.3b)
- **MEK1** phosphorylates **ERK2**. (Ex.3c)

We developed iSimp, a sentence simplifier, which aims to reduce the syntactic complexity of a sentence. It detects various constructs of a sentence, and then transforms them into a format that is easily accessible to text mining tools.

There are three main contributions in this work: (1) the development of iSimp, which uses an alternative method to detect various simplification structures in text in linear time; (2) the evaluation on different types of simplifications, which not only shows the promising results of sentence simplification, but also does so for the first time in terms of precision and recall in this domain; and (3) the evaluation on the impact of the simplifier for the performance of information extraction applications.

The rest of the manuscript is organized as follows. We first present related work. In Section III, we enumerate the types of syntactic constructs and the challenges encountered in their detection. Then, we describe iSimp in Section IV, followed by an evaluation of the sentence simplification in Section V and its impact on information extraction tools in Section VI. We conclude in the last section.

[*] Y. Peng and C. O. Tudor contributed equally to this work.

978-1-4673-2560-8/12/$31.00 ©2012 IEEE

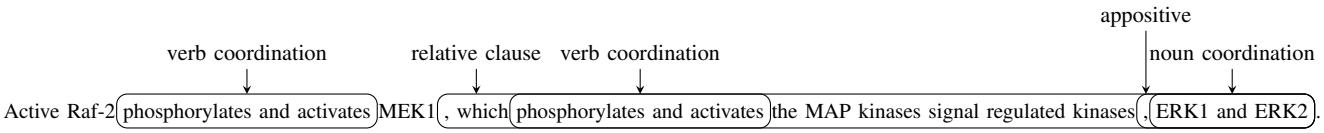

Figure 1. A sample sentence with simplification structures marked.

II. Related work

Automatic simplification of sentences was first introduced by Chandrasekar et al. [1,4] to improve the performance of parsing tools. Later, a broader range of simplification approaches was proposed to help people with aphasia [5,6], increase the readability of literature [7], or improve the performance of natural language processing (NLP) applications in various domains [8]. In particular, we note [2,7,9] are in biomedical domain.

Prior sentence simplification systems relied on syntactic parsing, where the simplifications were extracted directly from the parsing tree [2,4,9]. However, full parsing is slow and time-consuming on long sentences, being typically of $O(n^3)$. Furthermore, they are known to be prone to errors, especially, on long and complex sentences (which are typically the ones that require simplification), and they are less robust due to overly explicit analysis.

Previous efforts of building simplifiers using shallow parsing [3,4] have not performed adequately on sentences involving nested constructs. The key contributions of iSimp, which uses shallow parsing, are in the development of sophisticated rules to detect simplification constructs, including when they are nested. iSimp is more comprehensive in the detection of various types of simplifications, and can be used in a broader range of biomedical text processing applications. Due to the shallow parsing, iSimp is efficient and accurate to process large amount of text quickly.

III. Syntactic Simplification and Challenges

In contrast to full parsing, where detection of simplification constructs is part of the task, shallow parsing alone is not sufficient to detect all forms of simplifications. In this section, we define major syntactic constructs for simplification, and discuss the challenges that are frequently encountered during simplification. In the examples given throughout this manuscript, the syntactic structures are enclosed in square brackets and words that *trigger* the potential presence of simplification are emphasized in bold.

A. Simplification constructs

1) Coordinations: Coordinations are complex syntactic structures that link together two or more conjuncts of syntactically equal status [10]. These conjuncts are linked by coordinating conjunctions (e.g., "and", "or", and "but"). For example, the following sentence contains a coordination:

> An integral membrane protein can be found in [*multivesicular bodies/lysosomes* ***and*** *secretory granules*]. (Ex.4)

2) Relative clauses: Relative clauses are clauses that modify noun phrases[1]. Our method detects two categories of relative clauses. Full relative clauses are always introduced by relative pronouns, such as "which", "who", and "that" (see Ex.5). Reduced relative clauses start with a gerund/past participle and have no overt subject (see Ex.6).

> This gene is composed of multiple exons, [***which*** *span at least five cosmids*]. (Ex.5)

> A total of 11 additional families [***carrying*** *this mutation*] were identified. (Ex.6)

3) Appositions: Appositions are constructs of two noun phrases next to each other, typically separated by comma and referring to the same entity. One of the noun phrases (appositive) normally renames or describes the other.

> The Batten disease gene **,** [*CLN3*], maps to chromosome 16p12.1. (Ex.7)

4) Others: Subordinate clauses, introductory phrases, and parenthesised elements are also important types of simplification considered by iSimp. Their detection, however, is somewhat trivial, so we will not include them in the evaluation.

B. Challenges

1) Detection of a construct: Detecting reduced relative clauses and appositions is particularly challenging, as these may not be clearly marked by trigger words. Although reduced relative clauses start with gerund or past participles (Ex.8b), their presence does not always indicate the beginning of a construct (Ex.8a).

- the rate **limiting** enzyme in cholesterol synthesis (Ex.8a)
- the enzyme [***limiting*** *the endogenous pathway of cholesterol synthesis*] (Ex.8b)

Similarly, commas between two noun phrases are not enough to guarantee the presence of an apposition:

- eIF2alpha dephosphorylation, GADD34 and CreP, ... (Ex.9a)
- Two markers, [*D16S3070 and D16S3275*], ... (Ex.9b)

2) Left boundary detection: Detecting the left boundary of a coordination is particularly challenging, as it is not always clear how many conjuncts are involved. Commas are not always helpful in identifying the exact location of the left boundary. Sometimes, a coordination can be part of an apposition (Ex.9) or preceded by an introductory phrase, both of which are marked by commas.

[1]These noun phrases are called *referring name phrases* and are underlined in our examples.

Even for two conjuncts, there may be multiple candidates to consider. In addition to part-of-speech information, similarity of conjuncts can help in some cases. Consider the next example, for instance. Here, we might incorrectly mark "the IGF-I promoter" and "an ApaI polymorphism" as the two conjuncts.

> Glucose-stimulated insulin secretion uses hyperglycemic clamps in carriers of [*a CA repeat in the IGF-I promoter* ***and*** *an ApaI polymorphism in the IGF-II gene*]. (Ex.10)

3) Right boundary detection: Detecting the right boundary of simplification constructs is also challenging. Even for a simple coordination, such as "B and T cells", we need to recognize that "cells" refers to both "B" and "T", and thus mark the coordination as "[***B and T***] cells".

For nested construct, we need to determine which construct should be identified first. The following sentence shows three levels of relative clause nesting.

> We identified a chromosome translocation [*associated with ADPKD* [*that disrupts PBP* [*encoding a 14 kb transcript in the PKD1 candidate region*]]]. (Ex.11)

IV. Methodology

The system architecture contains three stages: preprocessing the original sentence, detecting simplification constructs, and generating simplified sentences. This section will describe in detail each of the three stages.

A. POS Tagging and Chunking

Given a raw sentence, we first apply a part-of-speech tagger to determine the corresponding linguistic category of each word in the sentence. These categories can be nouns, verbs, prepositions, etc. We trained and tested a tagger using maximum entropy model [11] on the GENIA corpus [12]. The training set contained 18,409 sentences, and the test set contained 2,045 sentences, on which the average F-measure was 0.98.

Next, we apply a shallow parser to group together words sharing related parts of speech. Our chunker was also developed, trained, and tested using maximum entropy on the GENIA corpus. To simplify the task, three types of chunks were investigated: noun phrases (NP), verb groups (VG), and prepositional phrases (PP). All other words not placed in one of these three types were marked with other (O). The F-measure obtained was 0.99.

B. Detection of Simplifications

For each type of simplification (see Section III), call it C, we construct a finite state machine, M_C. Three outcomes are possible for each machine: 1) *success*, in which the machine reaches a final state indicating the simplification structure is detected; 2) *failure*, in which the machine reaches a final state denoting the structure requirements are not met; and 3) *pending*, in which the machine suspects that another structure is nested inside and should be detected first.

Algorithm 1 Algorithm for simplification detection

```
for each words w of a sentence do
  if w is a trigger word of construct C then
    start and run an instance of machine M_C, say m_C
    if m_C is pending then
      push m_C into pending list
    else
      push m_C into complete list
    end if
  end if
end for
if pending list is not empty then
  solve nested cases using easy-case-first strategy
end if
```

The detection algorithm is provided as pseudocode in Algorithm 1. Generally, the input sentence is scanned from left to right. Whenever a trigger word is encountered marking the presence of one of the simplification types, we call the corresponding finite state machine. If this does not finish in a success or failure state, then we put it in the "pending" list, as this indicates potential nested constructs. When two or more constructs are nested, we apply the easy-case-first strategy. This means that constructs in the "pending" list are revisited after all the other constructs are attempted.

We now describe how the simplifier detects each of the simplification types.

1) Coordination: The trigger word for a coordination is a coordinating conjunction (e.g., "and", "or", etc.). Sometimes correlative conjunctions are used together with coordinating conjunctions (e.g., "both ... and", "between ... and", "either ... or", "neither ... nor", etc.). These are helpful to mark the beginning of a coordination construct.

By definition, all conjuncts in a coordination must be subject to the same syntactic category. Thus, we check if chunk sequences before and after the conjunction contain similar types of words. Our approach detects eight types of coordination: noun, noun phrase, verb, verb group, preposition and prepositional phrase, adjective, and adverb. We look at the similarity of the chunks proposed for the coordination (see Tables I and II) and consider those that are most similar for further processing. Our approach uses the head words (e.g., main words) of phrases for the comparison. We assume the head word of a noun phrase is the last noun or pronoun, the head of a verb group is the first verb, and the head of a prepositional phrase is the first preposition. We say two head words have the same type if they both follow the similarity rules listed in Tables I and II.

2) Relative clauses: Relative pronouns mark the beginning of a wh/that relative clause. Gerund/past participles mark the beginning of a reduced relative clause. Figure 2 shows the finite state machine that can detect a wh/that relative clause. For simplicity, we ignore the fail state. Failure can occur at any state if there is no corresponding edge for an input.

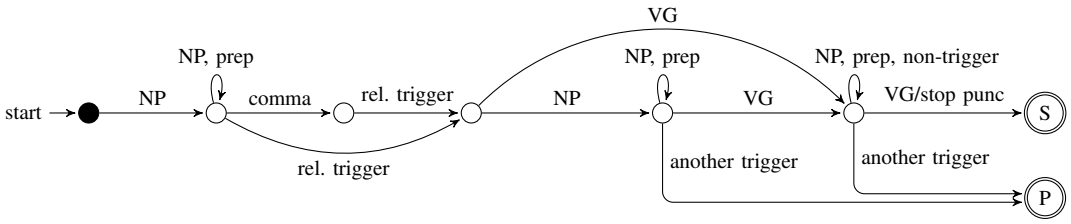

Figure 2. Finite state machine of detecting full relative clauses.

Table I
CRITERIA FOR NOUN PHRASE SIMILARITY

1. same word	6. uppercase words
2. numbers	7. containing hyphen/dash/slash
3. Greek alpha-beta	8. parenthesis elements
4. numbers followed by letters	9. common prefix
5. letters followed by numbers	10. common suffix

Table II
CRITERIA FOR VERB GROUP SIMILARITY

- They have same tense (e.g., present, past, future)
- They are of same grammatical number (singular or plural)
- They have same part-of-speech
- They are in same semantic group etc. (determined using DISCO [13])

3) Appositions: An appositive generally occurs between commas. However, not all structures that fall within commas are appositives. Therefore, one of the criteria in Table III must be satisfied. Whenever we encounter a noun phrase followed by a comma, we look to see if there is a noun phrase after the comma beginning with a determiner (e.g., "D16S3275, a microsatellite marker") or a number (e.g., "two markers, D16S3070 and D16S3275"), and so on. We further check if any or both of the noun phrases are outside coordination boundaries.

C. Generation of Simplified Sentences

After detecting the simplification constructs and their referring noun phrases, the simplifier generates separate sentences for each. For a coordination, the original sentence can be split into multiple ones, each containing one conjunct. For instance, Ex.4 can be simplified as:

- An integral membrane protein can be found in *multivesicular bodies/lysosomes*. (Ex.12a)
- An integral membrane protein can be found in *secretory granules*. (Ex.12b)

We note here that splitting noun phrase conjunctions plays an important role in information extraction tasks. Although the simplified sentences may not necessarily look simpler, information extraction would be more likely to be applicable to them.

A sentence containing a relative clause can be simplified into two sentences: one that skips over the relative clause and the other that combines the referring noun phrase with the relative clause. For instance, Ex.6 can be simplified as:

Table III
CRITERIA FOR APPOSITION DETECTION

One of two noun phrases begins with a number, a determiner (e.g., "a", "an", "the"), or words "other" or "another"

- If one noun phrase contains a number or word "both", the other one must contain a noun phrase coordination with the same number of conjuncts
- Both noun phrases can not be part of a noun or noun phrase coordination
- The first noun phrase can not be part of a introductory phrase

Table IV
PUBMED CORPUS FOR SENTENCE SIMPLIFICATION DETECTION

Types	**# of instances**	**% of sentences**	**% of abstracts**
Coordination	526	53%	98%
Relative clause	244	24%	85%
Apposition	43	4%	31%

- A total of 11 additional families were identified. (Ex.13a)
- 11 additional families *are carrying this mutation*. (Ex.13b)

Appositions can be simplified in the same manner as relative clauses.

V. EVALUATION ON SENTENCE SIMPLIFICATION

We evaluated our simplifier for detecting coordinations, relative clauses, and appositions using a manually annotated corpus. Such evaluation is non-trivial and has no precedence, as previous works focused only on the impact of sentence simplification.

A. Experimental design

Our simplifier was constructed based on a development data set consisting of Medline abstracts concerning proteins and genes. For evaluation, we randomly selected 100 Medline abstracts (a total of 954 sentences), having the words "protein" and "gene" in the title (see Table IV). We asked five judges to mark the six constructs. For annotation consistency, each sentence was annotated by two judges independently of each other. Whenever there were conflicts (57 sentences in total), a third opinion was sought.

Here we report two sets of results: (1) results on the assignment of the simplification type to a construct; and (2) results on the detection of the construct boundaries.

Table V
RESULTS FOR SIMPLIFICATION DETECTION

Types	Assignment			Assign.+boundary		
	P	R	F	P	R	F
Coordination	100.0	87.9	91.7	76.8	85.5	80.9
Relative clause	100.0	93.0	96.4	88.5	91.3	89.8
Apposition	93.8	83.3	88.2	93.8	83.3	88.2
Average	97.9	88.1	92.7	86.3	86.7	86.5

B. Results

For the first evaluation concerning types of simplification disregarding the boundary detection, we report an average precision of 97.9%, recall of 88.1%, and F-measure of 92.7%. Note that the distributions of the individual constructs are not the same (Table V). For the second evaluation, we counted as true positive all the instances in which both the simplification type and the boundaries were correctly identified. We report an average precision of 86.3%, recall of 86.7%, and F-measure of 86.3%.

C. Discussion

The majority of false negatives in the first evaluation were in the case of coordinations. In our approach, we do not rely only on lexical and syntactic information. We also look at the similarity of the conjuncts to be considered in a coordination (see Tables I and II). Although including the similarity feature helps identify the proper coordinations more often than it hurts, some errors are inevitable. Cases like "we measured [*m(b), and food intake*]..." are missed by our simplifier. In the future, rather than not attempting any coordination detection when there is no similarity, we may default to selecting the closest two candidates.

Most of the remaining false negatives were attributed to reduced relative clauses. Missed cases were due to the over-reliance on the part-of-speech tagger. Since one of the trigger words for a reduced relative clause is a past participle verb, we ignored the cases in which the verb was erroneously tagged as a past tense verb. To address this issue, we plan to use contextual clues besides part-of-speech.

In the second evaluation, the detection of the type, as well as the detection of the left and right boundaries, were considered (see second part of Table V). While there is hardly any drop in recall, we note that this time the issue is with false positives relevant to the boundary detection.

In the case of coordinations, half of false positives were attributed to erroneously attaching a left noun to the first conjunct or a right noun to the last conjunct rather than to the entire coordination.

> We further investigated the occurrence and frequency of $_{\text{incorrect}}[$ gene $_{\text{correct}}[$ *amplification and over-expression*] affecting RHBDD2 in 131 breast samples. (Ex.14)

The right boundary of a coordination was incorrectly identified in a few cases also due to the issue of prepositional phrase attachment. It is often difficult to identify if a prepositional phrase is attached to the last conjunct or to the entire coordination.

> The aim of this study was to investigate a possible association between [*haemochromatosis (HFE) gene mutations and the prevalence*]$_{\text{incorrect}}$ of Parkinson's disease]$_{\text{correct}}$. (Ex.15)

In the case of relative clause detections, a majority of false positives were due to nested constructs, involving a coordination, relative clause, or appositive.

> The results [*presented here*]$_{\text{correct}}$, and those of previous studies]$_{\text{incorrect}}$, suggest that ... (Ex.16)

VI. EVALUATION ON APPLICATIONS WITH SIMPLIFICATION

To demonstrate the utility of the simplifier, we further evaluated the impact of simplification on information extraction tools.

A. Evaluation on RLIMS-P

RLIMS-P is an information extraction system for protein phosphorylation information ([14]). It uses shallow parsing, semantic type assignment, and other techniques to process input text, and then applies hand-coded syntactic patterns to extract protein kinase, substrate, and phosphorylation site where an event trigger word is detected, e.g., "CaMKII$_{\text{kinase}}$ phosphorylates$_{\text{trigger}}$ serine 10$_{\text{site}}$ of p27$_{\text{substrate}}$". With sentence simplification, the number of extraction patterns could be greatly reduced, while maintaining or even improving the system performance. Fewer extraction patterns also means an advantage in terms of system maintainability, runtime, scalability, as well as portability to extraction of other posttranslational modification types.

To evaluate the utility of sentence simplification for RLIMS-P, we retrieved 1,000 Medline abstracts containing trigger words (phosphorylat + e, es, ted, and ion). 2,010 sentences were identified to contain both a trigger word(s) and a protein mention(s), based on the outputs of the RLIMS-P pre-processing modules. In these sentences, 2,824 pairs of trigger-protein were detected. Of them, 1,768 were identified as trigger-argument pairs by high-precision patterns. We took these to be valid pairs after a quick manual review. After applying sentence simplification, we extracted 343 additional pairs, which were manually inspected and confirmed as valid pairs. Overall, simplification allowed us to correctly extract phosphorylation information from nearly 20% more sentences using RLIMS-P rules and patterns.

B. Evaluation on $Rank_{Pref}$ for Sentence Selection

The second tool, $Rank_{Pref}$, deals with the ranking and selection of sentences containing a given gene name, a given relevant term, and the description of a relationship between the two [15]. This tool has been built to accompany the results of eGIFT [16], which automatically extracts relevant information for a gene from the biomedical literature. For example, consider the following sentence: "Bmp-2$_{\text{gene}}$

is known to promote osteoblastic cell differentiation and osteogenesis$_{\text{relevant_term}}$". The relationship between the two terms can be determined from the verb "to promote".

In [15], the authors also report results where a simplifier was used. The results are slightly different than the ones reported here, because their simplifier version uses other POS tagger and chunker, so the patterns are matched differently.

One important feature, used in the learning process for selecting and ranking sentences, marks the presence or absence of a syntactic relationship between the gene and the relevant term. This relationship is determined based on pattern matching applied at the syntax level of the sentence. The idea here is that matching of the patterns can be improved if sentence simplification is used.

Two evaluations were conducted. The first evaluation reports on the performance of the learned model to pick one sentence from a pair of sentences. When sentence simplification was applied, we observed an increase of 6% in accuracy from 74% to 80%, which represents a relative increase of 7.4% in the tool's performance. The second evaluation reports on the performance of the learned model to choose one sentence from a group of sentences. We observed an increase of 7% in accuracy from 67% to 74%, which represents a relative increase of 10.4% in the tool's performance, when the sentence simplifier was incorporated.

VII. Conclusions

We have developed iSimp, which detects various syntactic constructs of a complex sentence, and generates simplified sentences that can be readily processed by text mining applications. iSimp compares favorably with other simplifiers reported in the biomedical domain. In particular, iSimp handles nested simplifications, uses shallow parsing instead of full parsing, and evaluates well on the three types of syntactic constructs used in our approach: coordinations (76%), relative clauses (86%), and appositions (97%). The thorough evaluation and careful analysis also pointed to areas for future improvement.

We further demonstrated that iSimp can improve the results of information extraction tools: a *relative* improvement of nearly 20% in recall for phosphorylation information extraction and that of over 10% in accuracy for ranking sentences pertaining to biological events. We plan to evaluate our system for other text mining applications in the biomedical domain in the future.

Acknowledgment

This work has been supported in part by grants from the National Science Foundation (grant number 1062520) and the National Institutes of Health (grant number 1G08LM010720).

References

[1] R. Chandrasekar, C. Doran, and B. Srinivas, "Motivations and methods for text simplification." in *Procs. of COLING*, 1996, pp. 1041–1044.

[2] S. Jonnalagadda and G. Gonzalez, "Biosimplify: an open source sentence simplification engine to improve recall in automatic biomedical information extraction," in *Procs. of AMIA*, 2010, pp. 13–17.

[3] A. Siddharthan, "Syntactic simplification and text cohesion," Ph.D. dissertation, University of Cambridge, 2003.

[4] R. Chandrasekar and B. Srinivas, "Automatic induction of rules for text simplification," *Knowledge-Based Systems*, vol. 10, pp. 183–190, 1997.

[5] S. Devlin and J. Tait, "The use of a psycholinguistic database in the simplification of text for aphasic readers," *Linguistic Databases*, pp. 161–173, 1998.

[6] J. Carroll, G. Minnen, Y. Canning, S. Devlin, and J. Tait, "Practical simplification of English newspaper text to assist aphasic readers," in *Procs. of AAAI*, 1998, pp. 7–10.

[7] E. Ong, J. Damay, G. Lojico, K. Lu, and D. Tarantan, "Simplifying text in medical literature," *Journal of Research in Science, Computing and Engineering*, vol. 4, no. 1, pp. 37–47, 2007.

[8] D. Vickrey and D. Koller, "Sentence simplification for semantic role labeling," in *Procs. of ACL*, 2008, pp. 344–352.

[9] M. Miwa, R. Saetre, Y. Miyao, and J. Tsujii, "Entity-focused sentence simplification for relation extraction," in *Procs. of COLING*, 2010, pp. 788–796.

[10] R. D. Huddleston and G. K. Pullum, *The Cambridge Grammar of the English Language*. Cambridge University Press, 2002.

[11] A. Berger, S. D. Pietra, and V. D. Pietra, "A maximum entropy approach to natural language processing," *Computational Linguistics*, vol. 22, no. 1, pp. 39–71, 1996.

[12] Y. Tateisi, A. Yakushiji, T. Ohta, and J. Tsujii, "Syntax annotation for the genia corpus," in *Procs. of the IJCNLP, Companion volume*, 2005, pp. 222–227.

[13] P. Kolb, "Experiments on the difference between semantic similarity and relatedness," in *Procs. of NODALIDA*, vol. 4, 2009, pp. 81–88.

[14] Z.-Z. Hu, M. Narayanaswamy, K. E. Ravikumar, K. Vijay-Shanker, and C. H. Wu, "Literature mining and database annotation of protein phosphorylation using a rule-based system," *Bioinformatics*, vol. 21, no. 11, pp. 2759–2765, 2005.

[15] C. O. Tudor and K. Vijay-Shanker, "Rank$_{\text{pref}}$: Ranking sentences describing relation between biomedical entities with an application," in *Procs. of BioNLP in conjunction with NAACL-HLT*, 2012.

[16] C. O. Tudor, C. J. Schmidt, and K. Vijay-Shanker, "eGIFT: Mining gene information from the literature," *BMC Bioinformatics*, vol. 11, p. 418, 2010.