# An Artificial Immune System Based Approach for English Grammar Checking

Akshat Kumar and Shivashankar B. Nair

Indian Institute of Technology Guwahati, India
akshat.kumar@gmail.com, sbnair@iitg.ernet.in

**Abstract.** Grammar checking and correction comprise of the primary problems in the area of Natural Language Processing (NLP). Traditional approaches fall into two major categories: Rule based and Corpus based. While the former relies heavily on grammar rules the latter approach is statistical in nature. We provide a novel corpus based approach for grammar checking that uses the principles of an Artificial Immune System (AIS). We treat grammatical error as pathogens (in immunological terms) and build antibody detectors capable of detecting grammatical errors while allowing correct constructs to filter through. Our results show that it is possible to detect a range of grammatical errors. This method can prove extremely useful in applications like Intelligent Tutoring Systems (ITS) and general purpose grammar checkers.

## 1 Introduction

Grammar checking and correction constitute two of the major steps in Natural Language Processing (NLP). Their importance ranges from base applications, like using a word processor, to highly specialized tasks like transforming high level natural language commands into machine understandable forms. There are two well-known approaches for grammar checking namely the Rule-based ones and the pattern-matching-corpus-based approaches. Existing grammar checking systems, such as those described in [10], [1], [6], [4], fall into the former category, addressing the issue with a collection of heuristic rules that approximate a natural language grammar. The other approach is based on the application of corpus linguistics to the task of language processing [11].

While the rule based approach focuses on understanding the grammar of natural language, the corpus based approaches try to statistically analyze the language by taking the advantage of the abundance of available text. Our approach falls in this second category and is unique in the respect that it uses an Artificial Immune System (AIS) based technique. The motivation for our approach comes from the human immune system which is able to distinguish every harmful external entity from the self cells of the human body. We have modeled our grammar checker based on a similar approach so that it is able to identify any entity outside the corpus (regarded as error). The self in our case is the corpus itself.

In section 2 we describe briefly the basics of an Artificial Immune System. Section 3 focusses on how we adapt the AIS for the task of natural language processing. Section 4 describes how we generate antibodies for the grammar checker. Section 5 evaluates our approach on a collection of grammatical errors while Section 6 raises light on the

L.N. de Castro, F.J. Von Zuben, and H. Knidel (Eds.): ICARIS 2007, LNCS 4628, pp. 348–357, 2007.

limitations of our approach. Finally section 7 portrays the conclusions arrived at and future enhancements that could be made.

## 2 Artificial Immune System (AIS)

While the biological immune system generates antibodies to detect and defend the body of the being, its counterpart the artificial immune system (AIS) works on principles and algorithms laid down from theoretical immunology to evolve solutions for a range of problems. A good description of the biological and artificial sides of the immune system can be found in [3] and [9]. We describe the working of an AIS in brief.

Our human immune system is based on a collection of immune cells called *lymphocytes*. These primarily constitute the B-cells and the T-cells. Both these cell types present receptor molecules on their surfaces responsible for recognizing the antigenic patterns displayed by pathogens. The main role of a lymphocyte in an immune system is encoding and storing a point in the solution space or shape space. The match between a receptor and an antigen may not be exact and it takes place with a strength termed as affinity. If this affinity is high the antigen is said to be within the lymphocyte's recognition region.

After the successful recognition of the harmful pathogens an adaptive immune response is invoked. In this response those cells that were capable of identifying the pathogens (non self) proliferate by cloning. They may also undergo controlled mutation (hypermutation)[3] so as to fine tune their receptor molecules resulting in an increase in affinities. A selective mechanism guarantees that those offspring cells (in the clone) that better recognize the antigen and which elicited the response have longer life spans. These cells are called Memory cells. These memory cells are the ones that quickly identify the disease causing organism in future attacks and thus trigger a faster secondary immune response. The whole process of antigen recognition, cell proliferation and differentiation into memory cells is called clonal selection.

## 3 Adapting AIS for Language Processing

The idea of adapting an AIS to the field of NLP comes from the commonly observed fact that a person conversant in a language generally finds it difficult to generate an incorrect sentence in that language. It is argued here that a person fluent in a natural language has already built an immune system to detect and reject incorrect sentences in that language. In short he is immune to an incorrect language generation attack which is why he experiences difficulty in generating examples of incorrect sentences at the same frequency as correct ones. Grammar by itself is never talked off in the initial phases of language learning. A child picks up a language oblivious of grammar. A collection of sentences, that constitute a corpus, is formed initially and subsequent sentence generation largely depends on the combinations of words and grammar contained within it. The corpus thus could be viewed as the collection of correct sentences constituting a collection of the self cells (cases). Detectors could now be generated by constructs that do not exist within this corpus. Any form of grammar not found in the corpus could

be treated as an error and hence could be a candidate to mature into a detector. The underlying assumption that the corpus is complete, however has to hold.

Based on this we propose an architecture for detecting and modifying grammatical errors based on applying the anomaly detection capabilities of an AIS. In our approach we treat the grammatical errors in the sentences as metaphors for the non-self antigens which in turn elicit a response from the immune system viz. the grammar checker described herein. To catch the harmful antigens (grammatical errors) in the system a repertoire of antibodies is to be generated. This generation is based on the method described in [2] with modifications made to suit the language processing scenario. Figure 1 shows how the antibody and the antigen interact in the domain of languages. In the following sections, we describe the definitions of self and non-self followed by the antibody generation technique for grammar checking.

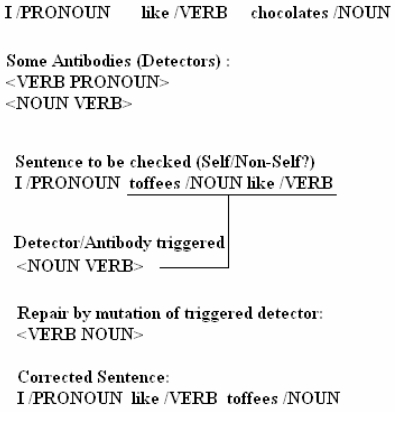



**Fig. 1.** Simple case of formation of antibodies, detection and correction of a sentence

### 3.1 Self

The set of self in our system is an extensively part of speech tagged corpus. Corpus is analogical to the human body and whatever constitutes the corpus is similar to the body cells or self cells. It may be possible that a valid grammatical structure be flagged as an error if it is outside the corpus. Given a sufficiently large corpus our system can perform satisfactorily covering a large number of grammatical constructs.

**The Corpus.** We are using the corpus Reuters-21578, Distribution 1.0 text collection. This corpus is a collection of various news documents appeared on the Reuters newswire in 1987. We have built a parser that can parse this corpus and extract meaningful text from it. The parser filters out documents in cryptic format like share market information and other economics related information to produce this meaningful text.

**Part of Speech Tagging.** The Reuters-21578 corpus is tagged for part of speech by using MontyLingua parser available at MIT [7]. This parser extracts extracts subject/verb/

object tuples, extracts adjectives, noun phrases and verb phrases, and extracts people's names, places, events, dates and times, and also does part of speech tagging from the English sentences within the corpus.

**Self Structure.** The Self is constituted by the bigrams, trigrams and tetragrams from the POS tagged corpus. The following example makes this process clearer.

**Sentence 1.** *Officials/NNS were/VBD not/RB immediately/RB available/JJ for/IN comment/NN ./.*

- *Bigrams: <NNS,VBD>, <VBD,RB>...*
- *Trigrams: <NNS,VBD,RB>, <VBD,RB,RB>...*
- *Tetragram: <NNS,VBD,RB,RB>, <VBD,RB,RB,JJ>...*

The tag-set used has only 36 members and this makes the system very general to properly model the underlying grammar of language. Therefore to capture more information about the actual language we have extended the tag-set by incorporating the use of some of the regular English words as the tags. By studying the existing tag-set, we have noticed that a single tag covers many English words. For example the tag DT covers articles such as *a, an, the, these, those* etc. To make fine distinction in the grammar usage we treat all these words as different tags. A list of words which should be treated differently was made based on the studies carried out on the Penn Treebank tag-set.

**Error Detection.** The basic units of the sentence which are analyzed are the bigram, the trigram and the tetragram of the part of speech sequences and of extended tags sequences in the test sentence. These components were tested against the Antibodies of our AIS. These Antibodies were generated to capture only the non-self (ungrammatical) entities. Once the antibodies recognized an antigen (or grammatical error) the system flags off the corresponding word sequence as one containing an error.

In the following section we describe the antibody generation phase of our AIS.

## 4 Generating Antibodies for Grammar Checking

This section describes the antibody generation phase. We use the Negative Selection algorithm [5] which is based on the principles of self-nonself discrimination in the biological immune system (see figure 2). This negative selection algorithm can be summarized as follows:

- Define *self* as a collection **S** of elements in a feature space **U**, a collection that needs to be monitored. For instance, if **U** corresponds to the space of states of a system represented by a list of features, **S** can represent the subset of states that are considered as normal for the system.
- Generate a set **F** of detectors, each of which fails to match any string in **S**. An approach that mimics the immune system generates random detectors and discards those that match any element in the self set.
- To monitor the new data, continuously check it against the generated set of detectors and if any detector matches regard it as an anomaly or error.

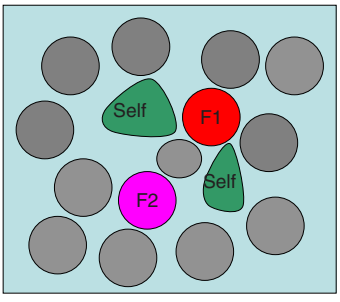



**Fig. 2.** The figure illustrates the concept of self and non-self in a feature space. F1 and F2 etc. indicate different fault condition represented by detectors.

### 4.1 Real Valued Negative Selection Algorithm (RNS)

The RNS detector generation starts with a population of candidate detectors, which are then matured through an iterative process. In particular, the center of each detector is chosen at random and the radius is a variable parameter which determines the size (in the m-dimensional space) of the detector. The basic algorithmic steps of the RNS detector generation algorithm are given in Figure 3.

At each iteration, the radius of each candidate detector is calculated, and the ones that fall inside *self* region are moved (i.e. its centre is successively adjusted by moving it away from training data and existing detectors). The set of non-self detectors are then stored and ranked according to their size (radius). The detectors with larger radii (and smaller overlap with other detectors) are considered as better-fit and selected to go over the next generation. Detectors with very small radii, however, are replaced by the clones of better-fit detectors. The clones of a selected detector are moved at a fixed distance in order to produce new detectors in its close proximity. New areas of the non-self space are explored by introducing some random detectors. The whole detector generation process terminates when a set of mature (minimum overlapping) detectors are evolved which can provide significant coverage of the non-self space.

A **detector** is defined as $d = (c, r_d)$, where $c = (c_1, c_2, .., c_m)$ is an m-dimensional point that corresponds to the center of a hyper-sphere with $r_d$ as its radius. In our system we use numeric references for tags, the self-set for our system consists of 2D (for bigram sequences) points, 3D (for trigram) points and 4D (tetragram) points. The following parameters are used for the detector generation process:

- $r_s$ The threshold value (allowable variation) of a self point; in other words, a point at a distance greater than or equal to $r_s$ from a self sample is considered to be abnormal. In our case since we want a strict checking for errors, this parameter would be zero.
- $\alpha$ The variable parameter to specify the movement of a detector away from a self sample or existing detectors.
- $\gamma$ The maximum allowable overlap among the detectors, which implicates that allowing some overlap among detectors can reduce holes in the non-self coverage.

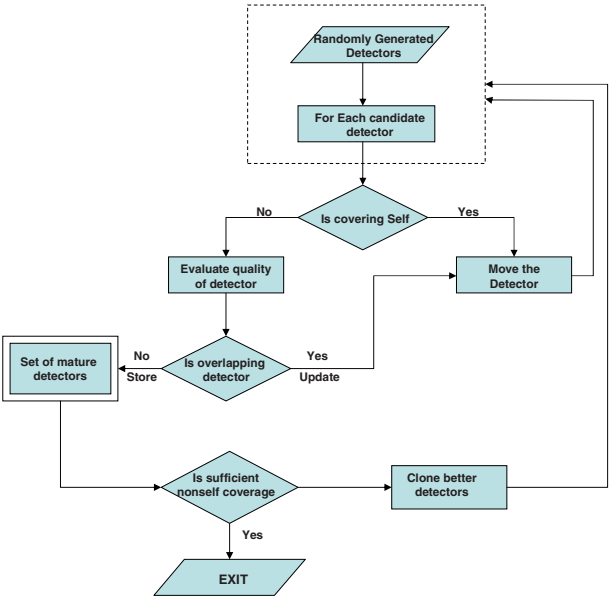



**Fig. 3.** Flow diagram showing the algorithmic steps for real valued negative selection algorithm

**Calculating the Detector Radius.** We have used the Euclidean distance to measure the distance between two points x and y, which is defined as: $D(x, y) = (\Sigma|x_i - y_i|^{\lambda})^{1/\lambda}$

Where $x = x_0, x_1, ..x_n$ and $y = y_0, y_1, ..y_n$ $with$ $\lambda = 2$

This approach allows having variable size detectors to cover the non-self space. As shown in Figure 4(a), if the distance between a candidate detector, $d = (c, r_d)$ and its nearest self point in the training dataset is D, then the detector radius is considered as $r_d = (D - r_s)$.

**Moving the Detector.** Let $d = (c, r_d)$ represents a candidate detector and $d^{nearest} = (c^{nearest}, r_d^{nearest})$ is its nearest detector (or a self point), then the center of d is moved such that

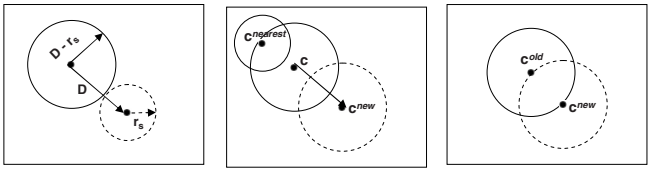
$$c^{new} = c + \alpha * dir/(||dir||) \tag{1}$$

where $dir = c - c^{nearest}$, and $||\ ||$ denotes the norm of an m-dimensional vector. Accordingly, if a detector overlaps significantly with any other existing detectors, then it is also moved away from its nearest neighboring detector.

**Detector Cloning and Random Exploration.** At every generation, a few better- fitted detectors are chosen to be cloned. Specifically, let $d = (c^{old}, r_d^{old})$ be a detector to be cloned and, say $d^{clon} = (c^{clon}, r_d^{clon})$, is a cloned detector whose centre is located at a distance $r_d^{old}$ from d and whose radius is the same as that of the detector, d. Accordingly, the centre of $d^{clon}$ is computed as

$$c^{clon} = c^{old} + r_d^{old} * dir/(||dir||) \tag{2}$$

Where $dir = c^{old} - c^{nearest}$ (where $c^{nearest}$ is the center of d's nearest detector)




(a) Calculation of the detector radius (b) Movement of the candidate detector (c) Cloning of a detector

**Fig. 4.** Steps in detector maturation process

**Evaluation of Non-self Detectors.** Detectors which do not fall in the *self* region are sorted according to their size. A detector with large radius gets selected for the next generation population, if it has small overlap with existing detectors i.e. less than the overlapping threshold. The overlapping measure W of a detector is computed as the sum of its overlap with all other detectors as follows

$$W(d) = \sum_{d \neq d'} w(d, d') \tag{3}$$

where $w(d, d')$ is the measured overlap between two detectors $d = (c, r_d)$ and $d' = (c, r_{d'})$; and is defined by

$$w(d, d') = (exp(\delta) - 1)^m, \text{ m is the dimension of the feature space} \tag{4}$$

$$and\ \delta = (r_d + r_{d'} - D)/2r_d \tag{5}$$

The value of $\delta$ is considered to be bounded between 0 and 1; and D is the distance between two detector centers c and c'. This overlapping measure seems to favor the detectors with bigger radii, i.e. detectors having larger coverage of the non-self space with minimum overlap among them.

## 5 Experimental Results

We have implemented the grammar checker as described in previous sections in JAVA and tested the same using test sentences from a book of grammatical errors [8]. The corpus for our checker is the Reuters-21578, Distribution 1.0 text collection available freely on the web. The list of these error types is provided below with sample sentences denoting errors.

We have found out eight categories of grammatical errors which our grammar checker can detect. However it is to be noted that it may not be able to correct all sentences of these categories as some of the finer distinction of a construct may be absent in the corpus. The incorrect constructs in the sentences are caught by the antibodies in our grammar checker while correct constructs are undetected.

### 5.1 Subject Verb Disagreement

**Sentence 2.** *These lines occurs in Shelleys poem. (DT NNS VBZ IN NN .)*

- ***Error*** *These lines occurs in*
- ***Correct*** *These lines occur in Shelleys poem.*

### 5.2 Choice of Tense

**Sentence 3.** *Those boys fight.(DT NNS VBP .)*

- ***Error*** *those boys fight*
- ***Correct*** *Those boys are fighting.*

**Sentence 4.** *I am hearing a noise outside. (PRP VBP VBG DT NN IN .)*

- ***Error*** *I am hearing a*
- ***Correct*** *I hear a noise outside.*

### 5.3 Article Misuse

**Sentence 5.** *Are you in hurry? (VBP PRP IN NN .)*

- ***Error*** *you in hurry*
- ***Correct*** *Are you in a hurry?*

### 5.4 Wrong Pronouns

**Sentence 6.** *Who of the two girls look good. (WP IN DT CD NNS VBP JJ .)*

- ***Error*** *Who of the two*
- ***Correct*** *Which of the two girls look good?*

### 5.5 Wrong Numbers

**Sentence 7.** *I lost my baggages in the train. (PRP VBD PRP NNS IN DT NN .)*

- ***Error*** *I lost my baggages*
- ***Correct*** *I lost my baggage in the train.*

### 5.6 Wrong Adverbs

**Sentence 8.** *We reached station timely. (PRP VBD NN JJ .)*

- ***Error*** *reached station timely*
- ***Correct*** *We reached station in time.*

### 5.7 Wrong Adjectives

**Sentence 9.** *This is most unique institution. (DT VBZ RBS JJ NN .)*

- ***Error*** *is most unique institution*
- ***Correct*** *This is unique institution.*

### 5.8 Missing Verb

**Sentence 10.** *I going home. (PRP VBG NN .)*

- ***Error*** *I going*
- ***Correct*** *I am going home.*

## 6 Limitations

Although our grammar checker catches a variety of constructs, experimentally we found out that many ungrammatical constructs are passed undetected. This deficiency can be traced to the limited tag-set being used in this work. Being small, this tag set is incapable of modeling the entire underlying English grammar. However this is not a shortcoming of our approach. If we use a more enhanced tag-set like the C7 tag-set used to annotate the British National Corpus then we can identify the different usage of words like *I* and *He* (treated same in current approach) because these two are assigned different tags PPIS1 and PPHS1 respectively.

The current system can not detect differences between singular and plural forms of a word (mostly pronoun). For example word *they* and *He* are given same tag PRP. Again this difficulty can be solved by using a more enhanced tagset.

In addition we still need to have some connection between the actual words used in the corpus and the part of speech tags. Currently our grammar checker only uses the N-gram sequences. To make a checker with high precision we would also have to incorporate these words in our error checking process.

## 7 Conclusion and Future Work

The motivation for our approach comes from the human immune system which is able to identify almost every external harmful entity that encroaches the human body. We have modeled our grammar checker on such an approach so that it is able to identify any grammatical construct outside the corpus and flag it as an error. The corpus constitutes the self of our system. Although this grammar checker is not able to trap all the errors, it may be inferred that the algorithms innately used by the human immune system are definitely effective as one level of detection.

Being language independent is one major advantage of our approach. Grammar rules for the language need not be provided *a priori*. Thus during the error detection phase no grammar rules are used to generate the antibodies or error detectors. The generation of antibodies is dependent only on the *self* set.

A major improvement can be the addition of the ability to record the antibodies or error detector usage information. This information would help us to identify those antibodies which are used repeatedly and in a way indicative of the most common mistakes committed by a person. This knowledge about a language learner's mistakes can provide valuable tips in many scenarios of pedagogy. Consequently our system can also be used as a plug-in for an intelligent language tutoring system for Non-Native speakers of English or for that matter other languages. Once a user interacts with the system for a sufficient amount of time the detector usage information can easily bring out the weaknesses of that person.

## References


1. Bolioli, A., Dini, I., Mahmti, G.: Jdii: Parsing italian with a robust constraint grammar. In: Proceedings of the 15th International Conference on Computational Linguistics, pp. 1003–1007 (1992)
2. Dasgupta, D., KrishnaKumar, K., Wong, D., Berry, M.: Negative selection algorithm for aircraft fault detection. In: Nicosia, G., Cutello, V., Bentley, P.J., Timmis, J. (eds.) ICARIS 2004. LNCS, vol. 3239, pp. 21–35. Springer, Heidelberg (2004)
3. de Castro, L.N., Timmis, J.: Artificial immune systems: A new computational intelligence approach. Springer, London (1992)
4. Flora, B., Fernando, R.: Gramcheck: A grammar and style checker. In: Proceedings of the International Conference on Computational Linguistics (COLING), pp. 365–370 (1996)
5. Forrest, S., Perelson, A., Allen, L., Cherukuri, R.: Selfnonself discrimination in a computer. In: Proceedings of the IEEE Symposium on Security and Privacy, IEEE Computer Society Press, Los Alamitos (1994)
6. Genthial, D., Courtin, J.: From detection/correction to computer aided writing. In: Proceedings of the International Conference on Computational Linguistics (COLING), pp. 1013–1018 (1992)
7. Liu, H.: Montylingua: An end-to-end natural language processor with common sense (2004)
8. Misra, A.K.: Avoid errors (1994)
9. Sompayrac, L.: How the immune system works. Blackwell Science, Oxford (1999)
10. Thurmair, G.: Parsing for grammar and style checking. In: Proceedings of the International Conference on Computational Linguistics (COLING), pp. 365–370 (1990)
11. Tschichold, C., et al.: Developing a new grammar checker for english as a second language. In: Technical Report Laboratoire de traitement du langage et de la parole