Recognizing Sentence Boundaries and Boilerplate

Nilani Aluthgedara
Department of Computer Science
University of Maryland, College Park
anilani@wam.umd.edu

Abstract

For many natural language processing tasks, identifying sentence boundaries is one of the most important prerequisites. Since punctuation marks are not used exclusively to mark sentence breaks, sentence boundaries are ambiguous. To achieve a reliable detection of sentence boundaries most systems use brittle, special-purpose regular expression grammar and exception rules. As an alternative, David D. Palmer and Marit A. Hearst have developed a trainable algorithm that uses a lexicon with part-of-speech probabilities and a feed-forward neural network. Their algorithm, however, still leaves some ambiguity in labeling the ends of the sentences, and it does not address the issue of boilerplate. In this paper, I discuss the work I have done to improve and modify the algorithm to perform better labeling of sentence boundaries and to recognize the boilerplate.

This work is submitted in fulfillment of the requirement for the degree of Bachelor of Science in Computer with honors at the University of Maryland.

Thesis Advisor: Professor Dianne P. O'Leary, Department of Computer Science and Institute for Advanced Computer Study (UMIACS).

Contents

Introduction

For many natural language processing (NLP) tasks, identifying sentence boundaries is one of the most important prerequisites. Yet, it has not received as much attention as it deserves. Many available natural language processing tools do not perform a reliable detection of sentence boundaries. Moreover, discarding boilerplate has not been taken into consideration in many sentence-labeling algorithms.

At first glance, it may appear that using a short list of end-of-sentence punctuation marks, such as periods, question marks, and exclamation points, is sufficient. Exclamation point and question mark are somewhat less ambiguous. However, since these forms of punctuation are not used exclusively to mark sentence breaks, sentence boundaries are ambiguous. For example a period can be used in an abbreviation, as a decimal point, in e-mail addresses and to indicate ellipsis. Some examples are shown below.

(1) She needs her car by 5 p.m. on Saturday evening.

(2) At 5 p.m. I had to go to the bank.

Identifying boilerplate in text is also a necessary prerequisite for natural language processing tasks such as automatic text summarization. However, work being done regarding this topic is not very extensive.

Related Work

Riley (Riley, 1989) determined that probabilities of a period occurring in the Tagged Brown corpus (Francis and Kucera, 1982) are about 90% at the end of a sentence, 10% at the end of an abbreviation, and about 0.5% as both abbreviation and sentence delimiters.

Many sentence boundary recognizing algorithms tokenize the text stream and apply a regular expression grammar with some amount of look-ahead, an abbreviation list, and sometimes a list of exception rules. In these algorithms tokenizing is mostly done immediately preceding and following the punctuation mark to be disambiguated. In some cases, as shown in the examples below, more context can be necessary.

(1) It was due Friday by 5 p.m. Saturday would be too late.

(2) She has an appointment at 5 p.m. Saturday to get her car fixed

There are some NLP systems designed to achieve accurate labeling of sentence boundaries. For example, at Mead Data Central, Mark Wasson and colleagues developed a system over a period of 9 staff months that could recognize special tokens (ex: non-dictionary terms, legal statute citations, etc.) as well as sentence boundaries (Palmer and Hearst, 1994). Wasson was able to build a stand-alone boundary recognizer in the form of a grammar converted into finite automata with 1419 states and 18002 transitions using his previous work. He was able to achieve nearly 99% accuracy with the upper-case legal test data yet the program did not produce accurate result for lower-case data.

Humphrey and Zhou developed a feed-forward neural network to disambiguate periods, which produced 93% accuracy over the tokenized text.

The approach Riley describes uses regression trees to classify sentence boundaries. His method uses information about one word of context on either side of the punctuation mark. Therefore, for each word in the lexicon the program required recording the probability that it occurs next to a sentence boundary. When tested against the Brown corpus the program was able to achieve 99.8% accuracy.

Palmer and Hearst developed an algorithm that uses a lexicon with part-of-speech probabilities and a feed-forward neural network. They feed the part-of-speech probabilities of the tokens surrounding a punctuation mark into a feed-forward neural network. The network's output activation value indicates the role of the punctuation mark. Their algorithm perform a better position than any of the others because of the following properties,

- The approach is robust, and does not require a hand-built grammar or specialized rules that depend on capitalization, multiple spaces between sentences, etc. Hence, the approach is easily adaptable to new text genres and new languages.
- The approach trains quickly on small training sets and does not require extra storage overhead. For example after training for less than one minute, the method

correctly labeled over 98.5% of sentence boundaries in a corpus of over 27,000 sentence-boundary marks.
- The approach is efficient and it does not noticeably slow down the NLP system's preprocessing stage.

There are still some inaccuracies in the results this program produces. For example,

Input sentence: She has an appointment at 5 p.m. Saturday afternoon.
Output produced: *She has an appointment at 5 p.m*. End of sentence
Saturday afternoon. End of sentence

My Solution

In this paper I present a method to enhance the accuracy of Hearst's and Palmer's project. In building my algorithm I took into consideration the following three features for any complete sentence.

1) Every sentence has a verb.
2) Every sentence starts with an uppercase letter.
3) All proper nouns start with an uppercase letter.

If a period indicates the end of the sentence there should be a verb before the period. This method works for marking the sentence as follows:

1) At 5 p.m. Saturday I went to the metro station.

When the program detects the period after the 'm' it has not yet found a verb. Therefore, the program recognizes that the period is not an end of sentence marker.

The period can occur after a verb. However it is not necessarily an end of sentence marker. Here is an example.

2) She has an appointment at 5 p.m. Saturday afternoon.

When the program detects the first period it has already found a verb. In this case, however, the period is not the end of sentence indicator.

For these kinds of sentences the algorithm had to be extended to follow:

1) When a period is detected check to see if there is a verb in the sentence before the period.

2) If there is a verb then check if the next word after the period starts with an upper case letter. A period followed by a word beginning with an upper case letter does not necessarily imply that it is at the end of a sentence (second example above). The reason for this is that there may be a proper noun after the period.

If the word following the period starts with an uppercase letter then check if it is a proper noun. By this test alone, however, we cannot determine if the period is an indication of the end of a sentence. A period at the end of an abbreviation can also mark the end of a sentence and the next sentence can be started with a proper noun.

Example:

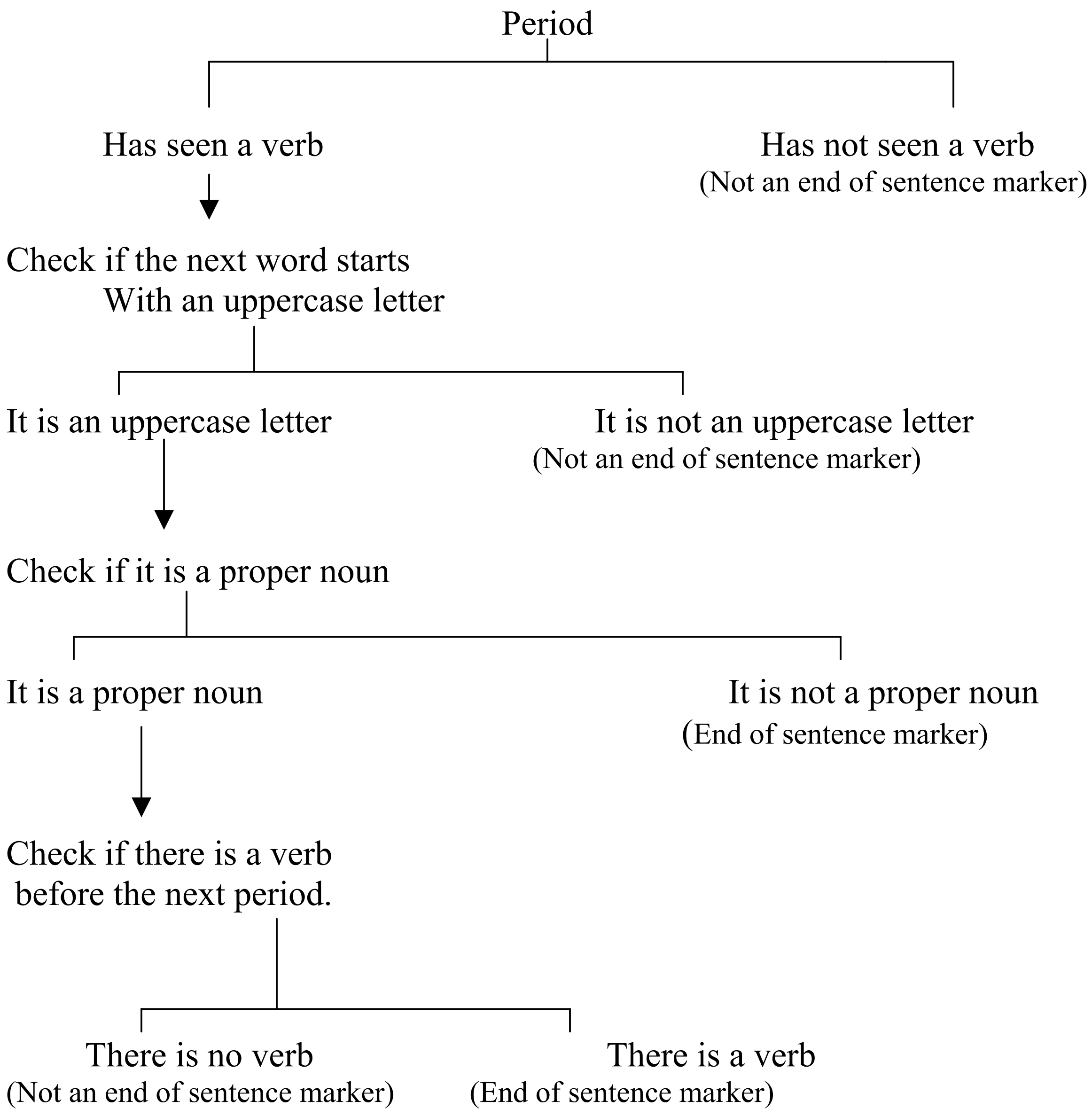
It was due Friday by 5p.m. Saturday would be too late.

3) To get rid of the above disambiguates I included the following look ahead algorithm to my method of marking sentence boundaries.

   When the program sees a period where the word following it is a proper noun it checks for a verb before the next period. If there is a verb then the first period that the program encountered is an end of sentence marker.

   The algorithm is presented in the flow chart on page 9.

The program uses a threshold value in determining the sentence boundary. The network of Palmer and Hearst produces a single value between 0 and 1 representing the confidence that a punctuation mark occurred at the end of sentence. After I determine whether the period is an end of a sentence marker or not I adjust the value so that the program produces more accurate sentence boundaries.


Period
Has seen a verb
Has not seen a verb
(Not an end of sentence marker)
Check if the next word starts
With an uppercase letter
It is an uppercase letter
It is not an uppercase letter
(Not an end of sentence marker)
Check if it is a proper noun
It is a proper noun
It is not a proper noun
(End of sentence marker)
Check if there is a verb
before the next period.
There is no verb
(Not an end of sentence marker)
There is a verb
(End of sentence marker)

Discarding Boilerplate

Recognizing boilerplate in a text is also another useful method in natural language processing. There is variety of boilerplate. One example is shown below.

Ex: <DOCNO> SJMN91-06254192 </DOCNO> <LEADPARA> Supreme Court nominee Clarence Thomas went before the Senate Judiciary Committee today, and leading Democrats said they would use his confirmation hearings to question whether he favors a "radical change" in American law including a ban on abortions.

In the algorithm I developed, my definition of boilerplate included any incomplete sentences (i.e. sentences without a verb). At first glance identifying boilerplate might look like an easy task; however, if a line contains boilerplate and a good sentence before the end of sentence marker then there should be a way to separate the boilerplate from non-boilerplate. The method I used in this algorithm is as follows.

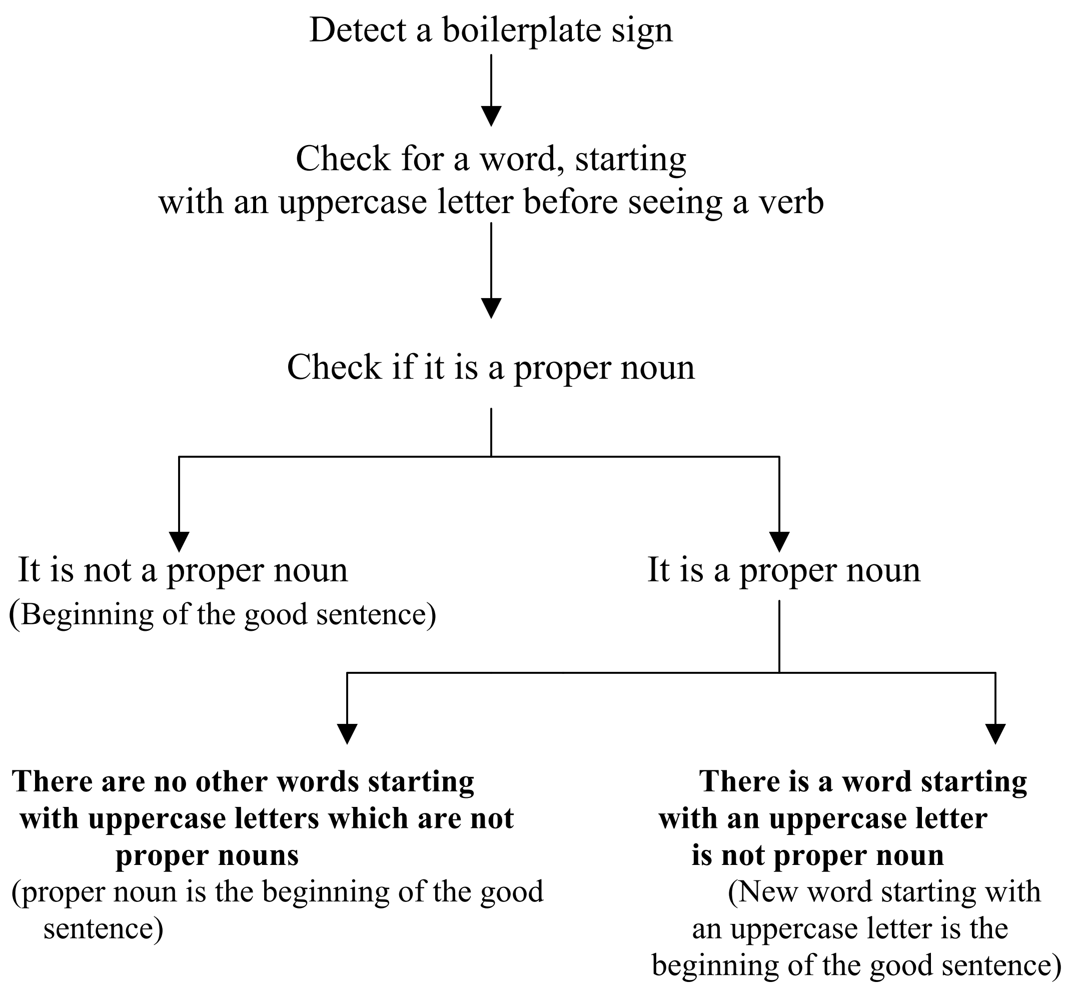
- Program detects a boilerplate sign ("<" or ">")
- Keep track of the words that start with an uppercase letter that precede a verb. The first word that the program detects is the beginning of the good sentence. This method does not always work because of the use of proper nouns.
- Therefore, another solution is to keep track of the first word, starting with an upper case letter, which is not a proper noun and is located before the verb in the sentence. However, the sentence may start with a proper noun.
- The final algorithm encompasses all of these issues.

    - Detect a boilerplate sign.
    - Excluding proper nouns check for other words starting with an uppercase letter before the verb.
    - Find the last occurrence of the word before the verb which begins with a capital letter and which is also not a proper noun. Make this the beginning of the good sentence.
    - If the first word is a proper noun and there is no word starting with an uppercase letter between the first word and the verb, consider the proper noun as the beginning of the good sentence.
    - The sentence ends when we find the end of sentence marker for the good sentence.

Output for above example:
Boilerplate:
<DOCNO> SJMN91-06254192 </DOCNO> <LEADPARA> Supreme
Good sentence:

Court nominee Clarence Thomas went before the Senate Judiciary Committee today, and leading Democrats said they would use his confirmation hearings to question whether he favors a "radical change" in American law including a ban on abortions.

The algorithm is presented in the flow chart.


Detect a boilerplate sign
Check for a word, starting with an uppercase letter before seeing a verb
Check if it is a proper noun
It is not a proper noun (Beginning of the good sentence)
It is a proper noun
There are no other words starting with uppercase letters which are not proper nouns (proper noun is the beginning of the good sentence)
There is a word starting with an uppercase letter is not proper noun (New word starting with an uppercase letter is the beginning of the good sentence)

Experimental Results

I gathered input sets for testing end-of sentence and boilerplate marking from e-mails and newspaper articles.

Marking end of sentences:

Input Set	My Algorithm	Original Algorithm
Set1 (3 sentences)		
EOS marked	3	4
Invalid	0	1
Valid	3	3
Valid/Total EOS ratio	1	0.75
Set2 (5 sentences)		
EOS marked	5	7
Invalid	0	2
Valid	5	5
Valid/Total EOS ratio	1	0.714
Set3 (174 sentences)		
EOS marked	208	259
Invalid	62	86
Valid	146	173
Valid/Total EOS ratio	0.702	0.668
Set4 (2435 sentences)		
EOS marked	2745	3271
Invalid	563	853
Valid	2182	2418
Valid/Total EOS ratio	0.795	0.739

In these experiments my algorithm always makes fewer end-of-sentences and fewer invalid end-of-sentences.

Examples of special cases that my solution produced better results:

- Having an abbreviation before a proper noun but after the verb.
  Ex: She has an appointment at 5 p.m. Saturday afternoon.

- Having an abbreviation before the verb.
  Ex: By 5 p.m. Sunday I have to be at home.

However, there are some situations (listed below) in which my algorithm does not recognize the end of sentence markers properly.

- Using proper nouns which are usually not proper nouns
  Ex: Mexico's new "debt reduction" deal with private banks falls well short of the kind of Third World relief dreamed of by theorists in the Elysee
  Palace or U.S. Senate.
  Here Senate is used as a proper noun but usually it is not categorized as a proper noun so the algorithm will mark the EOS after U.S.

- Using words, which are not in the dictionary that is used in the program.
  Ex: I send the letter at 10 a.m. Saturday is too late.
  The word "send" is not in the dictionary. Therefore, by the time the program found the period after a.m. it 'has not seen a verb'.

- Using words with ambiguous parts of speech, such as adjectives that can also be verbs.
  Ex: Adjusted value of the car by 5 p.m. Saturday was $ 9000.
  In this case when the project detects the period after 5 p.m. it acts as it has seen a verb ("adjusted") even though this word has been used as an adjective in this sentence.

As a result of the described reasons above, the threshold value for marking end of sentences is affected and hence the program marks invalid end of sentences for the sentences, which belong to these categories.

Marking Boilerplate:

Result obtained by using an e-mail as the input set:
Date: Sat, 21 Jun 2003 03:44:38 -0700 (PDT)
From: Merrick Berg <mmmmberg@yahoo.com>
To: lars@winds.gsfc.nasa.gov
Cc: Nilani <anilani@cs.umd.edu>
Subject: computer

Hi "*****GOOD***** Lars:

Call me around 4-5 p.m. if you would like to come over
this evening and configure your computer. End of Sentence
I'll be looking forward to see you tomorrow. End of Sentence
There are things that I don't understand in Treemap
that you may already know having spent so much time on it. End of Sentence
Maybe you could tell me those things right away rather than
me trying to understand spending too much time on them.

The program marked everything above "Hi" as boilerplate and inserted "*****GOOD*****" indicating that the word before it is the beginning of the non-boilerplate.

I gathered the data for following input sets from the e-mails and html documents. I used the following ratings to evaluate the results:

If the program includes zero words from the good sentence in the boilerplate - 1
If the program includes 2 or fewer words from the good sentence in the boilerplate – 0.8
If the program includes 3 - 4 words from the good sentence in the

boilerplate – 0.6
If the program includes 5 - 6 words from the good sentence in the boilerplate – 0.4
If the program more than 6 words from the good sentence in the boilerplate – 0

In similar method:
If the program includes no boilerplate in the good sentence – no deduction
If the program includes 1 - 2 boilerplate in the good sentence – -0.2
If the program includes 3 - 4 boilerplate in the good sentence – -0.4
If the program includes 5 - 6 boilerplate in the good sentence – -0.6
If the program includes more than 6 boilerplate in the good sentence – -1

At the end I add up all the points and divided by the number of sentences to get an average value for the input set.

Input Set	Performance
Set 1 (92 words)	1
Set2 (232 words)	0.89
Set3 (133 words)	0.70

Cases where my algorithm fails to mark the boilerplate accurately.

- Boilerplate followed by good sentence that starts with name containing words that are not categorized as proper noun.
  Ex: Cox News Service contributed to this report.
  The algorithm finds the last word starting with an uppercase letter and located before the verb. In this case it is "Service" since "News" and "Service" are not categorized as proper nouns usually.

- Boilerplate followed by punctuation marks (",", ":" etc.) and good sentence.
  Ex: <TEXT : Strom Thurmond, the panel's senior….
  "Strom Thurmond" is a proper noun. Therefore the program determined that the "TEXT" is the beginning of the good sentence.

Discussion of Future Work:

The algorithm I presented reduces the number of invalid end of sentence markers and also takes care of some special cases mentioned in the paper above. Furthermore, the program's marking of boilerplate is reasonably accurate. However, there are still some cases (as already described above) in which the program produces inaccurate results. In future work I am planning to examine the neural network in the original program more carefully and if possible modify it so that it will be able to take care of the cases that misled the current algorithm. Moreover, producing a more comprehensive dictionary for the program is another goal included in my future work.

References:

- David D. Palmer and Marti A. Hearst (1994). Adaptive sentence boundary disambiguation, *Report No. UCB/CSD 94/797*.
- Francis, W. and Kucera, H. (1982). Frequency Analysis of English Usage. Houghton Mifflin Co., New York. – cited in Palmer & Hearst.
- Humphrey, T. and Zhou, F. (1989). Period disambiguation using a neural network. In *IJCNN : International Joint Conference on Neural Networks*, page 606, Washington, D.C. – cited in Palmer & Hearst.
- Riley, M. D. (1989). Some applications of tree-based modeling to speech and language indexing. In *Proceeding of the DARPA Speech andn Natural Language Workshop*, pages 339 – 352. Morgan Kaufmann. – cited in Palmer & Hearst.