Assignment-1

CS6370: Natural Language Processing

Name: Abhishek Kumar
Roll no: cs21m002
Team no: 36

Q.1. *What is the simplest and obvious top-down approach to sentence segmentation for English texts?*

Ans: The most obvious top down version to solve sentence segmentation is to split a sentence at end of full-stop , question mark or exclamation mark (‘.’ or ‘?’ or ‘!’) also called Naïve top-down approach.

Text: Dhoni play cricket. Is he also play football? What a shot!
Segmented Sentence: “Dhoni play cricket.”, “Is he also play football?”, “What a shot!”

Q.2.*Does the top-down approach (your answer to the above question) always do correct sentence segmentation? If Yes, justify. If No, substantiate it with a counter example.*

Ans: No,
Sometimes english texts used abover marker as part of abbreviations.
e.g. → the short form for example,
Dr. → abbreviation for doctor,
A.M. → as in ante meridiem (10 A.M.).
With such occurrences of the period in a given piece of text, the top-down approach would end up producing erroneous and meaningless segments.

Text: Dr. Abhishek appointment is on 9 A.M.
Segmented Sentences: “Dr.”, “Abhishek appointment is on 9 A.”, “M”

The above top-down method also fails when grammar rules are not followed strictly.

Q.3. *Python NLTK is one of the most commonly used packages for Natural Language Processing. What does the Punkt Sentence Tokenizer in NLTK do differently from the simple top-down approach?*

Ans: The Punkt Sentence Tokenizer uses a bottom-up approach for sentence segmentation. A bottom-up approach to parsing a given piece of text works by first recognizing smaller portions of text, identifying their syntactic classes followed by combining results obtained for the smaller objects to identify bigger objects composed of those smaller objects until some kind of language entity (sentences in this case) is recognized.

Punkt tokenizer divides a text into a list of sentences by using an unsupervised algorithm to build a model for abbreviation words, collocations, and words that start sentences. It must be trained on a large collection of plain text in the target language before it can be used.

The specific technique used by the Punkt Sentence Tokenizer is called sentence boundary detection and it works by counting punctuation and tokens that commonly end a sentence, such as a period or newline, then using the resulting frequencies to decide what the sentence boundaries should actually look like.

The NLTK data package includes a pre-trained Punkt tokenizer for English.

Q.4. *Perform sentence segmentation on the documents in the Cranfield dataset using:*
(a) The top-down method stated above
(b) The pre-trained Punkt Tokenizer for English

State a possible scenario along with an example where:
(a) the first method performs better than the second one (if any)
(b) the second method performs better than the first one (if any).

Ans:
part a:
Naïve top-dowm approach perform better than Punkt when text is not properly punctuated as per standard English language. One best example is not give space after full stops. For Punkt, its like abbreviation ,but here naïve work well.

Text: Honesty is good.Abhishek is honest.Abhishek is gentle.

Naïve segmented Text: “Honesty is good.”, “Abhishek is honest.”, “Abhishek is gentle.”

Punkt Segmented text: “Honesty is good.Abhishek is honest.Abhishek is gentle.”
As “good.Abhishek” and “honest.Abhishek” as abbreviation because of missiong space after full stop.

part b:
Example mentioned in question 2 and "a. ferri's vortical layer is brought into evidence ."
In this sentence if we follow naive approach we have to split at a. which is wrong.
Q.5. *What is the simplest top-down approach to word tokenization for English texts?*

Ans: Word tokenization refers to the process of breaking down a piece of text into its constituent words. Simplest way to tokenize words is to split the words when they are separated by white spaces(White Space Tokenizer) or commas.

Q.6. *Study about NLTK's Penn Treebank tokenizer here. What type of knowledge does it use - Top-down or Bottom-up?*

Ans: The Treebank tokenizer uses regular expressions to tokenize text as in Penn Treebank. It is a top-down approach based on some fixed rules.

Q.7. *Perform word tokenization of the sentence-segmented documents using*
(a) The simple method stated above
(b) Penn Treebank Tokenizer
State a possible scenario along with an example where:
(a) the first method performs better than the second one (if any)
(b) the second method performs better than the first one (if any)

Ans:
part a

No example for now to say white space tokenizer is better than Treebank tokenizer. But white space tokenizer is fast, so it works better than the Penn Treebank Tokenizer.

part b
White space word tokenizer does not consider other punctuation marks,
so for this example `how do interference-free longitudinal stability measurements (made using free-flight models) compare with similar measurements made in a low-blockage wind tunnel .'
here brackets are ignored by white space tokenizer model but not by Treebank tokenizer.

Q.8. *What is the difference between stemming and lemmatization?*

Ans: Stemming and Lemmatization, both techniques are used to reduce the words to normalize form or root words. These techniques reduce unique words in the vocabulary and help to reduce the size of the dictionary.

Difference between Stemming and Lemmatization is:

Stemming:

(i) It generally operates on each word separately without considering the context of words in the sentence.

(ii) Part of Speech tags are not considered generally in stemming.

(iii) It may lead to words without meaning as Stemming is getting the base word after removing suffixes and prefixes
eg: Dancing may reduced to danc, Moving may get reduced to mov, which does not have any meaning.

(iv) Stemming done by using a top-down approach using some fixed rules.

(v) Stemming operation in the preprocessing step performs really well on tasks like spam classification and basic sentiment analysis.

Lemmatization:

(i) It considers words as well as context of words in the sentence.

(ii) Part of Speech tags are considered in lemmatization.

(iii) Lemmatization is getting the meaningful root word of the words.

Example: Ate reduced to eat. Moving will get reduced to move.

(iv) It takes morphological analysis of the words into account.

(v) Lemmatization in pre-processing step will be useful in language generation tasks like chatbot and question-answering applications as to generate something meaningful, the words should have meaningful representation.

Q.9. *For the search engine application, which is better? Give a proper justification to your answer.*

Ans: For search engines stemming is better than lemmatization The downsides of using lemmatization over stemming are as follow:

1) Lemmatization is slower than Stemming. So for fast search engine stemming is best option.
2) Lemmatization also gives less no of related words as it focuses on precision.eg: operate, operation and operational even though they are related they give different root words in lemmatization(themselves) while stemming gives same result(oper).This means we can get more related documents while using stemming over lemmatization which is what required for a general search engine.
3) Without pos tags to the word it is assumed as noun by wordnet lemmatizer. This means it is not accurate for some cases like meeting as as noun vs meeting as a verb. This can be solved by feeding sentences to spacy and lemmatizing during which spacy automatically finds pos tags.

but there are some downsides of using stemming as lemmatization does full morphological analysis to accurately identify the lemma for each word instead of following fixed rules
eg: for \`leaves' and \`leaf' stemming gives \`leav' and \`leaf' while lemmatization gives \`leaf' and \`leaf'.
Another advantage is we can improve lemmatization performance by updating dictionary and also results are interpretable.

Q.12. *In the above question, the list of stopwords denotes top-down knowledge. Can you think of a bottom-up approach for stopword removal?*

Ans: The bottom-up approach for stopword removal can be done by two methods.

1) Stop words have high frequency of occurrence
2) Stop words are general words and not used specifically in a certain domain, hence they occur in documents of every domain.

a) Statistical approach: In this approach, we find out how frequently a word appears in documents. We can compute measures like term frequency, document frequency or inverse document frequency and get to know the overall relevance of a word in any document. Generally stopwords are very less relevant in a document and the meaning of the document can be captured even if the stopwords are removed. After identifying them, we can remove all the stopwords using this approach.

b) Semantic approach : In this approach, we try to calculate the information gained from a stop word. Generally, a stop word carries very little meaning and the information gained from it will also be small. Information gained from a word is inversely proportional to the probability of word occurrence. Hence, the information gained from a word stopword will be very low as the probability of occurrence of a stopword in a document is very high. We can also say that the stopword will have very low discrimination value.