Given a document, select a relevant snippet

Asked 9 years, 10 months ago Active 9 years, 10 months ago Viewed 3k times

10

10

When I ask a question here, the tool tips for the question returned by the auto search given the first little bit of the question, but a decent percentage of them don't give any text that is any more useful for understanding the question than the title. Does anyone have an idea about how to make a filter to trim out useless bits of a question?

My first idea is to trim any leading sentences that contain only words in some list (for instance, stop words, plus words from the title, plus words from the SO corpus that have very weak correlation with tags, that is that are equally likely to occur in any question regardless of it's tags)

statistics nlp text-processing heuristics

edited May 14 '10 at 0:40

dmcer
7,808 31 41

asked May 13 '10 at 18:30

BCS
61.1k 58 169 268

Possible duplicate of: **C# Finding Relevant Document Snippets for Search Result Display** – hippietrail Oct 20 '12 at 17:56

1 Answer

16

Automatic Text Summarization

It sounds like you're interested in **automatic text summarization**. For a nice overview of the problem, issues involved, and available algorithms, take a look at Das and Martin's paper **A Survey on Automatic Text Summarization** (2007).

Simple Algorithm

A simple but reasonably effective summarization algorithm is to just select a limited number of sentences from the original text that contain the most frequent content words (i.e., the most frequent ones not including **stop list** words).

```
Summarizer(originalText, maxSummarySize):
   // start with the raw freqs, e.g. [(10,'the'), (3,'language'), (8,'code')...]
   wordFrequences = getWordCounts(originalText)
   // filter, e.g. [(3, 'language'), (8, 'code')...]
   contentWordFrequences = filtStopWords(wordFrequences)
   // sort by freq & drop counts, e.g. ['code', 'language'...]
   contentWordsSortbyFreq = sortByFreqThenDropFreq(contentWordFrequences)

   // Split Sentences
   sentences = getSentences(originalText)
```

```
    firstMatchingSentence = search(sentences, word)
    setSummarySentences.add(firstMatchingSentence)
    if setSummarySentences.size() = maxSummarySize:
       break

// construct summary out of select sentences, preserving original ordering
summary = ""
foreach sentence in sentences:
  if sentence in setSummarySentences:
     summary = summary + " " + sentence

return summary
```

Some open source packages that do summarization using this algorithm are:

Classifier4J (Java)

If you're using Java, you can use **Classifier4J**'s module SimpleSummarizer.

Using the example found here, let's assume the original text is:

> Classifier4J is a java package for working with text. Classifier4J includes a summariser. A Summariser allows the summary of text. A Summariser is really cool. I don't think there are any other java summarisers.

As seen in the following snippet, you can easily create a simple one sentence summary:

```
// Request a 1 sentence summary
String summary = summariser.summarise(longOriginalText, 1);
```

Using the algorithm above, this will produce `Classifier4J includes a summariser. .`

NClassifier (C#)

If you're using C#, there's a port of Classifier4J to C# called **NClassifier**

Tristan Havelick's Summarizer for NLTK (Python)

There's a work-in-progress Python port of Classifier4J's summarizer built with Python's Natural Language Toolkit (NLTK) available **here**.

edited May 14 '10 at 3:57

answered May 14 '10 at 0:40

dmcer
7,808 31 41

I wonder if the C# version is fast enough to be used for this site? – BCS May 14 '10 at 0:55

The algorithm is **dead simple**, so it really should be fast enough. It first determines the **most frequent content words** in the original text. It then iterates over them and selects the **earliest sentence** in the original string that contains each word. This continues until the desired number of N many sentences are selected. – dmcer May 14 '10 at 3:01

I recently used this algorithm and believe me its really easy to implement in C# and it does give good results. I needed to play around with a few settings here and there such as getting rid of empty spaces or return key spaces. It takes a couple tries. Thank you. – Karim O. Mar 29 '14 at 5:31