March 29, 2011

Morphological Analysis Searches For You—Intelligently

Comparing the effectiveness of n-gram indexing vs. morphological analysis

by Steve Cohen, EVP/COO

BASIS TECHNOLOGY

We put the World in the World Wide Web®

ABOUT BASIS TECHNOLOGY

Basis Technology provides software solutions for text analytics, information retrieval, digital forensics, and identity resolution in over forty languages. Our Rosette® linguistics platform is a widely used suite of interoperable components that power search, business intelligence, e-discovery, social media monitoring, financial compliance, and other enterprise applications. Our linguistics team is at the forefront of applied natural language processing using a combination of statistical modeling, expert rules, and corpus-derived data. Our forensics team pioneers better, faster, and cheaper techniques to extract forensic evidence, keeping government and law enforcement ahead of exponential growth of data storage volumes.

Software vendors, content providers, financial institutions, and government agencies worldwide rely on Basis Technology's solutions for Unicode compliance, language identification, multilingual search, entity extraction, name indexing, and name translation. Our products and services are used by over 250 major firms, including Cisco, EMC, Exalead/Dassault Systems, Hewlett-Packard, Microsoft, Oracle, and Symantec. Our text analysis products are widely used in the U.S. defense and intelligence industry by such firms as CACI, Lockheed Martin, Northrop Grumman, SAIC, and SRI. We are the top provider of multilingual technology to web and e-commerce search engines, including Amazon.com, Bing, Google, and Yahoo!.

Company headquarters are in Cambridge, Massachusetts, with branch offices in San Francisco, Washington, London, and Tokyo. For more information, visit www.basistech.com.

BASIS TECHNOLOGY

Yoshio is meeting a friend in Tokyo to see the first showing of the new “Lupin” movie and he needs to get the show time quickly. He types two phrases into a Japanese web search engine: 東京都 (“Tokyo Capital”) and ルパン上映時間 (“Lupin show times”). For some reason, though, half of his search results are for bakeries in Kyoto!? Yoshio probably won’t be using this search engine again!

What happened? How did “Lupin show times in Tokyo” become “bakeries in Kyoto”? Let’s look at how search engines match keywords with Japanese webpages to understand how this confusion can occur, and why processing Japanese text requires sophisticated linguistic analysis.

Japanese text, like our example “東京都 ルパン上映時間”, normally uses no spaces to separate words; readers know how to separate the words based on their understanding of the grammar and vocabulary of the language. Without spaces though, how can a search engine find individual words to store in its index? There are two common ways to isolate (segment) these words: N-gram and morphological analysis.

N-GRAM SEGMENTATION

The “N-gram” technique is one approach to languages such as Japanese, Chinese, and Korean which use no spaces between words. It requires less programming work on the back end and requires no dictionary or linguistic rules, but makes for less gratifying front-end results for the user.

N-gram segmentation works by chopping text into segments n-characters long, usually two to three characters. When “n” is two, a sliding window isolates strings two-characters long from the input text.

To illustrate this, consider the English input text:

When in Rome

Using a value of N=2, the n-gram system produces the strings at right, which are stored in the search index.

A query for any of the words in “When in Rome” will also be segmented using the N-gram segmenter, so it will match these strings. However since they don’t match the actual words in the original text, many other unrelated strings might accidentally match them as well, and a search becomes a statistical process: Do enough of these two-character segments appear in a given document to flag it as a “match,” while not accidentally matching the wrong strings?

This approach also produces many more entries in the index than words, unnecessarily increasing its size. Processing a 100-character buffer (about 13-17 words), adds 99 entries to the index.

Wh	he	en	n□	□i	in	n□	□R	Ro	om	me

Now consider a Japanese language example:

東京都 ルパン上映時間

N-gram segmentation produces the following tokens:

東京	京都	都□	□ル	ルパ	パン	ン上	上映	映時	時間

As a whole, Japanese words contain fewer characters per word, so the number of "false words" created by the segmentation process can be relatively high. In the example, there are four genuine words:

東京都	(Tokyo)
ルパン	(Lupin, the name of a movie character)
上映	(playing/screening)
時間	(time)

N-gram segmentation produced three false words (and just two of the four actual words):

都	(capital city)
京都	(Kyoto)
パン	(bread)

The net result of searching with n-gram is many results and a poor match for the user's expectations.

MORPHOLOGICAL ANALYSIS SEGMENTATION

Morphological analysis requires a bit more work upfront, a dictionary, and the intelligence to recognize features of the language: punctuation, actual words, word forms and affixes (prefixes and endings). By using a rule-set specific to the language, morphological analysis will find the same words that a human would see in the text.

Morphologically analyzed, our English example adds three entries to the index (instead of 11 n-grams):

When	in	Rome

Similarly for Japanese, just the four recognized words are indexed (instead of 10 n-grams):

東京都	ルパン	上映	時間

And with its linguistic smarts, morphological analysis can return linguistically similar terms such as plural/singular and conjugated forms. A search for directions to "install air conditioner" finds articles on "installing air conditioners."

The net result of using morphological analysis is:

- accurate matches that meet user expectations
- faster performance with a smaller index
- ability to match linguistically similar terms

... and no more false hits on “Kyoto” or “bread” when you want to know what time “Lupin” is playing in Tokyo.

N-gram segmentation is often the quickest way to add limited Asian language support to a search or text analytics system, but the counterintuitive behavior of the end product is generally not satisfactory in the home markets for these languages. Customers in these locales are demanding and have high expectations for how their languages will be processed—major web search engines all use morphology—so it is almost always better to integrate a morphological analysis engine at the beginning of an internationalization project.

NOTES FOR LUCENE, SOLR AND NUTCH DEVELOPERS

Users of the Lucene and Solr open-source search engine components will be familiar with the Standard Tokenizer or the Standard Analyzer built into these applications. These analyzers are primarily designed for English and work acceptably for European languages where a space separates words. For Chinese and Japanese, however, these analyzers treat each Chinese and Japanese character as one token. Thus Lucene uses a uni-gram (N-gram where N=1) method for Chinese and Japanese indexing. In this case the city name “東京都” ("Tokyo Capital") would wind up with each character indexed: “東”, “京”, and “都”. A search for the character “東” (“East”) would return the full word “東京都” and any other word which happens to include this character.

Lucene also includes what is commonly known as CJK Tokenizer and Analyzer, and Solr has the corresponding factory class. These are based on the bi-gram method and have the same limitations described earlier in this document.

Nutch (http://lucene.apache.org/nutch/) is another open-source search engine based on Lucene. It has a plug-in mechanism with which language analyzers can be added at run time. Nutch comes with language analyzer plug-ins for German and French. Developers will have no difficulty making a Rosette Linguistics Platform-based language analyzer plug-in, using the sample Lucene integration code and the source code for GermanAnalyzer found in the Nutch distribution. Nutch does not offer the same plug-in mechanism for the analyzer used for queries, however. This makes it necessary to make small modifications of Nutch source code.

Basis Technology's Rosette linguistics platform includes a code sample that can be used to integrate the morphological analyzers of various languages with Lucene and Solr easily.

EXPLORE FURTHER

For more information or to request an evaluation, please call us at 617-386-2090 or 800-697-2062, or write to info@basistech.com. We will be happy to assist you in evaluating the performance of our products on your data.

ABOUT THE AUTHOR

Steve Cohen (stevec@basistech.com) is Executive Vice President and Chief Operating Officer of Basis Technology, where he is responsible for worldwide sales and the planning and operations of the company's linguistic product research and development. Before starting Basis Technology with Carl Hoffman, Steve was engineering manager for Cognex Corporation's Tokyo office and development manager for SMT device inspection. He has also consulted on software internationalization engineering and developed software for embedded systems and electronic test equipment. Steve earned a bachelor's degree in electrical engineering from MIT and studied at Waseda University in Tokyo.