INFORMATION RETRIEVAL UTILIZING SEMANTIC REPRESENTATION OF TEXT

TECHNICAL FIELD

The present invention relates to the field of information retrieval, and, more specifically, to the field of information retrieval tokenization.

BACKGROUND OF THE INVENTION

Information retrieval refers to the process of identifying occurrences in a target document of words in a query or query document. Information retrieval can be gainfully applied in several situations, including processing explicit user search queries, identifying documents relating to a particular document, judging the similarities of two documents, extracting the features of a document, and summarizing a document.

Information retrieval typically involves a two-stage process: (1) In an indexing stage, a document is initially indexed by (a) converting each word in the document into a series of characters intelligible to and differentiable by an information retrieval engine, called a "token" (known as "tokenizing" the document) and (b) creating an index mapping from each token to the location in the document where the token occurs. (2) In a query phase, a query (or query document) is similarly tokenized and compared to the index to identify locations in the document at which tokens in the tokenized query occur.

Figure 1 is an overview data flow diagram depicting the information retrieval process. In the indexing stage, a target document 111 is submitted to a tokenizer 112. The target document is comprised of a number of strings, such as sentences, each occurring at a particular location in the target document. The strings in the target document and their word locations are passed to a tokenizer 120, which converts the words in each string into a series of tokens that are intelligible to and distinguishable by an information retrieval engine 130. An index construction portion 131 of the information retrieval engine 130 adds the tokens and their locations to an index 140. The index maps each unique token to the locations at which it occurs in the target document. This process may be repeated to add a number of different target documents to the index, if desired. If the index 140 thus represents the text in a number of target documents, the location information preferably includes an indication of, for each location, the document to which the location corresponds.

In the query phase, a textual query 112 is submitted to the tokenizer 120. The query may be a single string, or sentence, or may be an entire document comprised of a number of strings. The tokenizer 120 converts the words in the text of the query 112 into tokens in the same manner that it converted the words in the target document into tokens. The tokenizer 120 passes these tokens to an index retrieval portion 132 of the information retrieval engine 130. The index retrieval portion of the information retrieval engine searches the index 140 for occurrences of the tokens in the target document. For each of the tokens, the index retrieval portion of the information retrieval engine identifies the locations at which the token occurs in the target document. This list of locations is returned as the query result 113.

Conventional tokenizers typically involve superficial transformations of the input text, such as changing each upper-case character to lower-case, identifying the individual words in the input text, and removing suffixes from the words. For example, a conventional tokenizer might convert the input text string

The father is holding the baby.

into the following tokens:

the
father
is
hold
the
baby

This approach to tokenization tends to make searches based on it overinclusive of occurrences in which senses of words are different than the intended sense in the query text. For example, the sample input text string uses the verb "hold" in the sense that means "to support or grasp." However, the token "hold" could match uses of the word "hold" that mean "the cargo area of a ship." This approach to tokenization also tends to be overinclusive of occurrences in which the words relate to each other differently than the words in the query text. For example, the sample input text string above, in which "father" is the subject of the word "held" and "baby" is the object, might match the sentence "The father and the baby held the toy," in which "baby" is a subject, not an object. This approach is further underinclusive of occurrences that use a different, but semantically related word in place of a word of the query text. For example, the input text string above would not match the text string "The parent is holding the baby." Given these disadvantages of conventional tokenization, a tokenizer that enacts semantic relationships implicit in the tokenized text would have significant utility.

SUMMARY OF THE INVENTION

The invention is directed to performing information retrieval using an improved tokenizer that parses input text to identify logical forms, then expands the logical forms using hypernyms. The invention, when used in conjunction with conventional information retrieval index construction and querying, reduces the number of identified occurrences for which different senses were intended and in which words bear different relationships to each other, and increases the number of identified occurrences in which different but semantically related terms are used.

The invention overcomes the problems associated with conventional tokenization by parsing both indexed and query text to perform lexical, syntactic, and semantic analysis of this input text. This parsing process produces one or more logical forms, which identify words that perform primary roles in the query text and their intended senses, and that further identify the relationship between those words. The parser preferably produces logical forms that relate the deep subject, verb, and deep object of the input text. For example, for the input text "The father is holding the baby," the parser might produce the following logical form:

deep subject	verb	deep object
father	hold	baby

The parser further ascribes to these words the particular senses in which they are used in the input text.

Using a digital dictionary or thesaurus (also known as a "linguistic knowledge base") that identifies, for a particular sense of a word, senses of other words that are generic terms for the sense of the word ("hypernyms"), the invention changes the words within the logical forms produced by the parser to their hypernyms to create additional logical forms having an overall meaning that is hypernymous to the meaning of these original logical forms. For example, based on indications from the dictionary that a sense of "parent" is a hypernym of the ascribed sense of "father," a sense of "touch" is a hypernym of the ascribed sense of "hold," and a sense of "child" and sense of "person" are hypernyms of the ascribed sense of "baby," the invention might create additional logical forms as follows:

deep subject	verb	deep object
parent	hold	baby
father	touch	baby
parent	touch	baby
father	hold	child
parent	hold	child
father	touch	child
parent	touch	child
father	hold	person
parent	hold	person
father	touch	person
parent	touch	person

The invention then transforms all of the generated logical forms into tokens intelligible by the information retrieval system that compares the tokenized query to the index, and submits them to the information retrieval system.

BRIEF DESCRIPTION OF THE DRAWINGS

Figure 1 is an overview data flow diagram depicting the information retrieval process.

Figure 2 is a high-level block diagram of the general-purpose computer system upon which the facility preferably operates.

Figure 3 is an overview flow diagram showing the steps preferably performed by the facility in order to construct and access an index semantically representing the target documents.

Figure 4 is a flow diagram showing the tokenize routine used by the facility to generate tokens for an input sentence.

Figure 5 is a logical form diagram showing a sample logical form.

Figure 6 is an input text diagram showing an input text fragment for which the facility would construct the logical form shown in Figure 5.

Figure 7A is a linguistic knowledge base diagram showing sample hypernym relationships identified by a linguistic knowledge base.

Figure 7B is a linguistic knowledge base diagram showing the selection of hypernyms of the deep subject of the primary logical form, man (sense 2).

Figure 8 is a linguistic knowledge base diagram showing the selection of hypernyms of the verb of the primary logical form, kiss (sense 1).

Figures 9 and 10 are linguistic knowledge base diagrams showing the selection of hypernyms of the deep object of the primary logical form, pig (sense 2).

Figure 11 is a logical form diagram showing the expanded logical form.

Figure 12 is a chart diagram showing the derivative logical forms created by permuting the expanded primary logical form.

Figure 13 is an index diagram showing sample contents of the index.

Figure 14 is a logical form diagram showing the logical form preferably constructed by the facility for the query "man kissing horse."

DOCKET ALARM

Explore Litigation Insights

Docket Alarm provides insights to develop a more informed litigation strategy and the peace of mind of knowing you're on top of things.

Real-Time Litigation Alerts

Keep your litigation team up-to-date with **real-time alerts** and advanced team management tools built for the enterprise, all while greatly reducing PACER spend.

Our comprehensive service means we can handle Federal, State, and Administrative courts across the country.

Advanced Docket Research

With over 230 million records, Docket Alarm's cloud-native docket research platform finds what other services can't. Coverage includes Federal, State, plus PTAB, TTAB, ITC and NLRB decisions, all in one place.

Identify arguments that have been successful in the past with full text, pinpoint searching. Link to case law cited within any court document via Fastcase.

Analytics At Your Fingertips

Learn what happened the last time a particular judge, opposing counsel or company faced cases similar to yours.

Advanced out-of-the-box PTAB and TTAB analytics are always at your fingertips.

API

Docket Alarm offers a powerful API (application programming interface) to developers that want to integrate case filings into their apps.

LAW FIRMS

Build custom dashboards for your attorneys and clients with live data direct from the court.

Automate many repetitive legal tasks like conflict checks, document management, and marketing.

FINANCIAL INSTITUTIONS

Litigation and bankruptcy checks for companies and debtors.

E-DISCOVERY AND LEGAL VENDORS

Sync your system to PACER to automate legal marketing.

WHAT WILL YOU BUILD? | **sales@docketalarm.com** | **1-866-77-FASTCASE**

fastcase®
Smarter legal research.