

Tools for Text Categorization

Herbert L. Roitblat, Ph.D.
OrcaTec

Two of the biggest problems in analyzing and classifying text are synonymy and polysemy. Synonymy means that there are lots of ways to say the same thing. Polysemy (pol-IS-seh-mee) means that the same words can mean different things in different contexts.

Thesauri

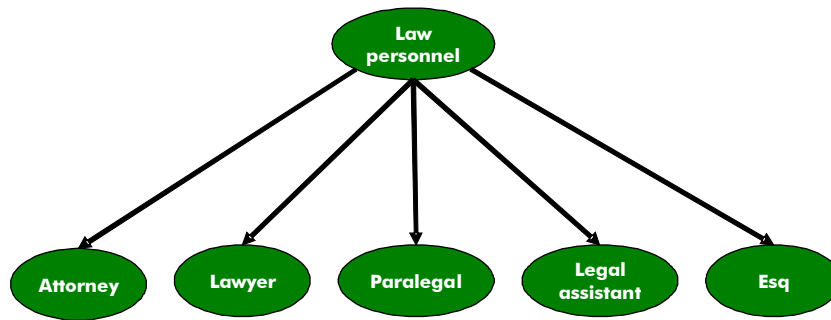
To deal with the problem of synonymy, some systems rely on a thesaurus, which lists alternative ways of expressing the same or similar ideas. The quality of the results depends on the quality of the thesaurus, which, in turn, depends on the effort expended to match the vocabulary and usage of the organization using it. Generic thesauri, which may attempt to represent the English language or are specialized for particular industries, are sometimes available to provide a starting point, but each group or organization has its own jargon and own way of talking that requires adjustment for effective data mining. In America, for example, the word “cleric” is a synonym for “minister.” In Iraq, “cleric” is a synonym for “Ayatollah.” Minister refers to a member of the government. An American thesaurus could actually interfere with appropriate categorization of documents about Iraq.

Taxonomies

Taxonomies and ontologies are also used to provide conceptual categorization. A taxonomy is a hierarchical scheme for representing classes and subclasses of concepts. The preceding figure shows a part of a taxonomy for types of personnel. Attorneys, lawyers, etc. are all kinds of law personnel. The only relations typically included in a taxonomy are inclusion relations. Items lower in the taxonomy are subclasses of items higher in the taxonomy. For example, the NAICS (North American Industry Classification System) is one generally available taxonomy, which is used to categorize businesses. In this taxonomy, the category “Information” has subclasses of “Publishing” and “Motion Picture and Sound Recording Industries” and “Broadcasting.”

You can use this kind of taxonomy to recognize the conceptual relationship among these different types of personnel. If your category includes law personnel, then any document that mentions attorney, lawyer, paralegal, etc. Should be included in that category.

Taxonomy

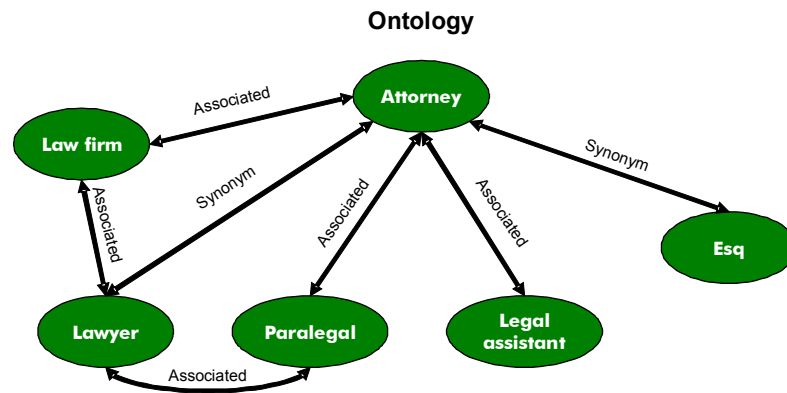


Like thesauri, there are a number of commercially available taxonomies for various kinds of industries. Adapting these taxonomies to any particular organization or matter can require substantial work. Predefined taxonomies exist for major business functions and specific industries.

An ontology is similar to a taxonomy, but it includes a variety of relationship types in addition to class inclusion. An ontology specifies the relevant set of conceptual categories and how they are related. The preceding figure shows part of an ontology with a subset of its connections. According to this ontology, if your category includes attorneys, you may also be interested in documents that mention “lawyer,” “paralegal,” or “Esq.” Like taxonomies, ontologies are most useful when they are adapted to the specific information characteristics of the organization.

Taxonomies, ontologies, and thesauri are all kinds of knowledge structures. They represent explicit knowledge about some domain. An expert in this area has to write down the specific relations that he or she can think of. Although there are tools that help the knowledge expert to create these structures, they still tend to represent only the information that the expert can describe as important.

Generic versions of these knowledge structures inevitably preserve distinction that are not useful and fail to make distinctions that are useful to specific contexts. Each knowledge structure, moreover, is a snapshot of the organization at a particular point in time. Human language and conceptual structures are constantly evolving over time, both as people become more experienced and as the environment changes. New issues emerge and old issues are resolved or subsumed. Knowledge structures require a great deal of effort to create and to maintain. Even over the course of a legal matter, the kinds of categories and relations that may be important can change, and some adaptation may be required to deal with these changes.



In actual practice, using taxonomies and ontologies also requires the development of rules for assigning text to the elements or nodes of the knowledge structure. You need rules to determine which words or documents correspond to each concept. These rules can be created explicitly, or they can be created heuristically using machine-learning techniques. Explicit rules are created by knowledge engineers. For example, one rule might include a Boolean statement like this: (acquir* or acquisition or divest* or joint venture or alliance or merg*) and (compet* or content or program*) that specifies the critical words that must appear for a document to be assigned to the "merger" category. The effectiveness of rules like these depends critically on the ability of the knowledge engineers to guess the specific words that document authors actually used. Syntactic rules may also be employed by some systems. For example, a system may only look for specific words when they are part of the noun phrase of a sentence.

Clustering

Alternatively, systems may use statistics or other machine-learning tools to recognize what information belongs to a category. The simplest of these is the use of statistical clustering. Clustering is the process of grouping together documents with similar content. There are a variety of ways to define similarity, but one way is to count the number of words that overlap between each pair of documents. The more words they have in common, the more likely they are to be about the same thing.

Many clustering tools build hierarchical clusters of documents. Some organize the documents into a fixed number of clusters. The quality of clustering is rarely as high as that obtained using custom built taxonomies, but since they require no human intervention to construct, clustering is often an economical and effective first pass at organizing the documents in a collection. Some systems improve the quality of clusters

that are produced by picking a number of clusters and adding specific documents to them. Then additional documents are added to these clusters if they are sufficiently similar. Bayesian classifiers are typically used in this way.

Bayesian classifiers

When using a Bayesian classifier, the knowledge engineer typically finds 10 – 20 documents that are characteristic of each category. The system then employs a statistical approach to discover which words are typical of each category (and not of the others). From this information, the system is able to classify additional documents into these categories by assessing the probability of each category from the words in the document. The success of such systems depends strongly on the representativeness of the example documents.

Bayesian classifiers work by computing the probability of each class in the taxonomy given the words in the document. For example, if a particular word occurs only in documents about mergers and in no other documents, then it is a good bet that any document that has that word is about a merger.

Bayesian classifiers are more subtle than this, of course. Each word in the document contributes to the classification based on the relative probabilities of the word in the sample documents for each class.

Meaning-based approaches

One reason that the knowledge structure approach is so popular is that it resembles the traditional structure of a filing cabinet, a metaphor often carried over to disk storage. Every document is in a folder, and every folder is in a drawer, etc. Everything has a place and everything is in its place. Such organizational systems made excellent sense when documents were on paper and could be stored in only one or a few places. Electronic documents, though, could be stored virtually everywhere simultaneously. They could be categorized simultaneously by every single word they contain without taking up any more storage space than if they were indexed by only a single concept. Indexing every word is the basis of word search tools.

Preplanned knowledge structures organize the data on the way into the system. They assume that the people who categorized the documents could anticipate the various ways that users of the system would seek the documents. Although taxonomies certainly have a value in information management, they are generally not sufficient because they “fight the last war.” They represent how people have used the information, not how they will need or want to use it in the future. They represent solutions to problems that have been solved, but the people who seek the information will be solving problems that may not even be anticipated at the time that the taxonomies are constructed. In addition to these knowledge structures, an organization needs a system that provides a flexible way to access information to solve future problems.

Full text search is one approach to providing this flexibility. The problem with word search is the difficulty of guessing the right words to search for. People are very flexible in their choice of words and, as a result, it is difficult to guess which words were actually used. For example, imagine that you need to write a report about the factors that cause

schedule creep. As it turns out, none of the documents in the collection actually use the phrase “schedule creep.” There is no entry in the taxonomy or ontology for “schedule creep.” Yet there are documents that talk about “schedule erosion,” “schedule problems,” “schedule risks,” “schedule delay,” “schedule impact,” “exceed schedule,” “shift schedule to the right,” “schedule growth,” “extending the schedule.” A search for “schedule” by itself, simply returns too many documents to be useful. It would be valuable to have a system that could recognize the similarity among these different phrases and let you find documents no matter which of these was actually used.

These phrases share some aspects of meaning with one another. They tend to be used in similar contexts. Fortunately, there are a number of computational techniques that are quite effective for getting at these shared meanings. These systems use neural networks, statistical procedures, or language models to derive underlying dimensions of meaning from the documents. Users can then exploit these underlying dimensions (e.g., what the schedule-creep phrases have in common) to find relevant documents even if they do not have the exact words in them that the user had in mind.

These systems derive underlying dimensions from a document collection using some variation of eigenanalysis. Eigenanalysis employs eigenvectors (and eigenvalues, but we can ignore eigenvalues here). These systems derive shared dimensions of meaning from the patterns of word usage in a document. They provide a simplified representation of the documents by combining mathematically, the parts of shared meanings the words. When a user looks for one of these words, the system automatically, then recognizes that other words are related to the query and can find these related documents.

A document that mentions lawsuits is also likely to mention lawyers, judges, attorneys, etc. In principle, we don't have to represent each of these words separately. Conversely, documents that mentioned any of these terms would be likely to be about the practice of law. These words are not synonyms, but they do share certain semantic characteristics. The eigensystems extract those relationships and exploit them to represent the concepts in the documents. A word search algorithm, in contrast, would have to represent all of these words separately in order to find documents related to any of them, and would only return documents that contained the one you actually searched for. A word search tool cannot take advantage of the conceptual relationship among these words.

Latent Semantic Indexing

The best known of these eigensystems is latent semantic analysis (also known as latent semantic indexing, LSI), but there are other systems that do similar things (e.g., DolphinSearch's neural network system and OrcaTec's language modeling system). The actual computational techniques these systems use are substantially different from one another, but the general goal of both of them is to extract and exploit the underlying dimensions of meaning. The general idea is to reduce the high degree of variability of language down to a more manageable volume and use that reduced volume (called reduced dimensionality) to manage the information needs of the users.

LSI is based on a statistical procedure that recognizes redundancy. LSI computes a way to summarize terms' shared meaning by examining the terms shared context. It places each word along one or more “meaning scales,” which are statistical summaries of the

redundancy (statisticians call these scales the eigenvectors). LSI typically computes about 300 of these summary meaning scales, but no effort is made to label them. Each word and each document can be represented as a value on these scales. Relative to word search, LSI improves the quality of search results because it reduces the effects of synonymy (words with the same meaning are likely to be located close together) and polysemy (words with different meanings are likely to be located far from one another along these scales). One way to think about LSI is that it represents the words that might be in the document, not just those that happen actually to be in the document, and it does this by computing a mathematical simplification of the documents.

Word	Dim1	Dim2	Dim 3	...
Car	1.1	.4	0	
Truck	.97	1.3	.2	
Bus	.92	.15	.01	
Wheelbarrow	.1	.7	.45	

This table shows a hypothetical set of meaning scales for a few words. In use, tables like this can be many thousands of words long with 300 or more meaning-scales / eigenvectors. Car, truck, and bus are closely related words and they are strongly related to the first scale (shown by the high number or weight). LSI does not tell us what to call that scale; the scale represents an underlying concept rather than a word of English. Still, it seems to be related somewhat to the concept of “vehicle.” The second scale might be related somewhat to the concept of “carrying stuff.” The third might be related to “dirt.” In this scheme, the best vehicle, the word most strongly related to the concept of vehicle is car, but truck and bus are also pretty strongly related as well. A search for “car” would also be likely to turn up documents with the similar words “truck” or “bus.”

LSI considers documents that have many words in common to be semantically close, relative to documents with few words in common. It also considers that words that are used in similar contexts—specifically words that co-occur in documents frequently—are also semantically similar. One of its main values, is that these similarity relations are not terribly fussy. LSI does not require exact matches the way a keyword search would. It allows fuzzy matches between documents, so that when you search for a particular query, documents that match that query exactly as well as documents that match that query only partially will also be returned. LSI can return documents that match the query to some degree, even if they do not have any of the query words in them.

LSI has been around since the 1980s. It has been studied extensively and a number of shortcomings have been identified. First, the algorithm simplifies the problem of finding the appropriate meaning scales by eliminating every word that occurs in only a single document. That does not sound like much until you remember that about half the words in a collection’s vocabulary are rare. If you count all of the documents that each word occurs in, some words will occur in almost all of the documents (words like “the,” “and,” and others. A few more words will occur in most of the documents, but some words occur only occasionally in a document, and about half occur in only one of these documents. They don’t all occur in the same document, obviously, but they occur only very rarely among the documents. If you are looking for a rare document in a collection,

one that is characterized by a unique word (e.g., a person's name), you will fail to find it using a system based on LSI. LSI is good for capturing the general meaning of documents, but it is weaker when dealing with tiny details.

LSI has also been found to be unsuitable to large document collections. To compute the statistics that form the foundation of LSI takes a lot of computer power, memory, and time. Large document sets, those containing more than about 100,000 documents may not be (depending on details of the implementation) computable at all. The statistics that are the basis of LSI have to be computed from a training set. The maximum size of these training sets is usually limited to about 10,000 documents. Larger collections can be indexed, but only about 10,000 can be used for training.

Finally, LSI has to compute its underlying dimensions in a batch. That means that the system cannot simply learn about new documents, it has to start over. It can index additional documents as they are added, but in order for the system to exploit any new knowledge in them, it must recompute its eigenvectors.

The other mentioned systems that compute underlying meaning dimensions are not subject to these same limitations. Every word in every document is indexed. Documents can be added and trained on an ongoing basis and there is no fundamental limit to the number of documents that can be used in training. These systems can be highly effective in allowing one to retrieve documents by the meaning of the words that they contain, rather than just by the words they happen to contain.

The challenge to using computers to retrieve and manage information is the dynamic characteristic of language. People could use a wide variety of terms to refer to the same thing and they can use the same terms to refer to a wide variety of things. These approaches to information management use various tools to try to resolve the problems caused by dynamic language. They all work to amplify the intelligence of their users. Users do not have to try to think of every possible way that a document's author might have said something. They don't have to know exactly where to look, but can exploit the knowledge structures and computer simplifications to help find the information that they need.

Herbert L. Roitblat, Ph.D.
Principal
OrcaTec LLC
PO Box 613
Ojai, CA 93024
herb@orcatec.com
805-212-8265