

Next Generation Search Platforms: How Vendors are Searching Unstructured Content

by Rich Turner

Search 2009

All of us are very familiar with search – it’s Google, and it’s ubiquitous. Google has done a very good job at monetizing search for the masses. The company incorporates elements of many different technologies, both Boolean and conceptual; it leverages these technologies to quickly mine content; and it presents results in a very familiar, comfortable, non-threatening way. Google has even extended its platform into the enterprise search market, and Google One-Box has a commanding market share.

What is very important is to separate Google the business from Google the software. As a business, Google is an advertising company – plain and simple. It is not unlike the early TV broadcasting companies, when 1950s era soap operas were just that – engrossing melodramas written for housebound homemakers, sponsored by laundry detergent manufacturers, with the sole purpose of promoting more soap.

This business model works very well for the masses. Rail as we might that Google doesn’t provide unbiased answers, that Google searches only present a small snippet of all the information that’s out on the web, or that Google seems to be promoting one company over another, without advertising revenues there would be no possible way a company could – or would – invest the hundreds of millions of dollars Google has spent to develop and deploy its search.

The model works somewhat for business-to-consumer marketers. The Google search paradigm (its uncluttered keyword search window) is copied by most websites, and the Google advertising paradigm (presenting relevant advertisers within or as part of the results) can be seen on most consumer information sites such as maps and news services.

Rich Turner is with Content Analyst Company. He can be reached by email at rturner@contentanalyst.com

But it doesn’t work well for businesses. Even One-Box adopters struggle with the “fixed parameters” that drive Google. In business, there is no advertising revenue for search: It is a means to an end, a necessity to find critical business information and to act upon it.

Therefore, over 100 companies – a number that continues to grow rather than shrink – provide some type of search software to a variety of business customers. These companies exist because they are pioneering technologies that focus on searching *unstructured information*, which is the bulk of what we store and capture today. Searching structured data like spreadsheets and databases is fairly mature. The same holds true for keyword or Boolean search – by themselves, these mark-and-index solutions are widely deployed, and some are even available as open source software. The focus for search software companies is the *next generation* technologies that will drive past keyword search and better address the exponential explosion of unstructured information by understanding the concepts within that information.

Many enterprise implementations combine techniques – much like Google did – but focus on providing businesses with robust platforms that can address the challenge of finding actionable information. There are three technologies that are getting wide press and have found applicability in numerous business solutions. This article will look at all three, what they do and the pros and cons behind each technique. Which is best? The sad answer is that it all depends. Knowing what your options are is the first step to finding that answer.

The three technologies we’ll look at are natural language processing or NLP, Bayesian inference and latent semantic indexing and analysis. One of these is based on sentence structure, and the other two are based on mathematical algorithms, but all three try to solve the challenge of *understanding meaning* so that search can be based on concepts versus

keywords. Why is this important? Simply, because language is complicated and fluid, and there are many ways to express similar things. For cognitive beings, this complexity is a wonderful adaptation, but for computers – whose processing is binary by nature and design – it is a huge drawback. Boolean operands attempt to bridge this gap, but remembering if/and and or/else combinations, performing “fuzzy search” tricks, don’t address the problem head-on. These technologies all do, in one form or another.

Natural Language Processing

Natural language processing or NLP is the oldest and best-known computer-based language technology. The idea behind NLP – which sprang from the same Xerox-sponsored Palo Alto Research Center (PARC) think tank that brought us the mouse and ultimately Windows – is that there is structure behind language, and if you can understand the structure, you can understand the language. This assumption makes perfect sense: We communicate using sentences, and they are comprised of nouns, verbs, adjectives and the like. Noun + action = result.

NLP doesn’t work by itself; what researchers found was that within the language there was no inherent expression of meaning. Simple sentences could be expressed as “noun + action = result” but more complicated thoughts sent early NLP systems into a tailspin. What was needed – and are now available in incredible variety – were word lists. Some lists are dictionary-based – they impart meaning into the words that NLP has deconstructed from language. Others tackle language nuances called synonymy (multiple words with similar meanings) and polysemy (words with multiple meanings often based on context).

From the start NLP needs structure – that is, a taxonomy or word hierarchy because word lists are very specific to subject matter. A medical term list makes no sense to an aerospace engineer, and 20th-century synonyms might not be meaningful to Y-generation conversation and writing. So NLP correlates lists of terms and definitions (and how those terms and meanings are combined) to express meanings, concepts and context. Does NLP work? Yes, very well, and it’s deployed in many variations across a range of solutions. Are there challenges? Definitely. Because NLP depends upon

outside structures such as dictionaries and thesauri to derive ultimate meaning, those structures must be maintained – constantly, as language continually evolves. Which outside structure is also important – not all words are created equal, to paraphrase Thomas Jefferson. And finally, a good taxonomy is needed to put all this into context.

Technologies Based on Mathematical Algorithms

The two other technologies are related in that they use mathematics instead of language structure. Why math? Two reasons, actually. The first is that the structure of language – like any other repeatable structure – can be expressed in mathematical terms. The second is that by expressing language in terms of math, you can potentially avoid the need to use the outside structures discussed above. The promise of using mathematics-based search technologies is that you can create and deploy a more robust engine that won’t need to know particular information about what it’s indexing yet can still derive meaning. In theory, size would also not be as big a challenge, for example, to scalability. It shouldn’t matter how complex a sentence or thought was if you could express it mathematically. The two mathematical techniques look at different spectrums of the mathematical nuances of text.

Bayesian inference. The first we’ll discuss is Bayesian inference, which actually harks back to Rev. Thomas Bayes, who in the 17th century proposed his famous Bayesian Theorem. (Note to readers: whether Bayes actually came up with this theorem is one of those hotly contested historical debates). Bayes’ Theorem is a probability theorem, originally proposed to explain gambling outcomes. The idea is that if something occurs along with other things, and a similar thing occurs elsewhere along with other similar things, then those things are all likely related. Bayesian inference, as the practical application of the theorem has been called, has been successfully applied to a number of scientific questions, from thermal transfer to energy generation.

It wasn’t until the late 20th century that companies started looking at how Bayesian inference could apply to language. It turns out that Bayesian inference can perform quite well and is able to accurately determine conceptual similarity among documents and terms. Better still, Bayesian can handle a lot of documents – in fact, it needs a fairly large corpus in order to

establish patterns, probabilities and relationships. Bayesian inference can also be successfully used to organize documents, called categorization or clustering, based on conceptual relevance.

The biggest challenge that Bayesian inference has is dealing with very complex concepts and relationships. Here, the very elegance of Bayes' Theorem is its downfall: For each additional condition, the theorem must investigate all permutations around that condition. Complex concepts – with a lot of contingencies, conditions and facets – can cause a Bayesian engine to perform a lot of extraneous calculations, bringing throughput to a crawl. The providers of Bayesian-powered search engines compensate for this problem by linking word lists and dictionaries to their engines. While they're not technically needed, they provide a useful assist when dealing with such concepts. One other caveat for Bayesian is that in order to accurately perform categorization based on example documents, the engine needs both relevant examples and non-relevant examples – again, the provision of both types of documents speeds the engine up, enabling it to also know what it's not looking for when grouping documents.

Latent semantic indexing. The other mathematical search technology is based on word co-occurrence rather than probability. It is also a geometric model, using vector-based math across hundreds of dimensions to identify conceptual relevance. This technology is latent semantic indexing and analysis, which is not quite as old as Bayes' Theorem, but does go back to Bell Labs in the early 1980s. The scientists at Bell Labs patented LSI, but decided it had no future at AT&T because it dealt with text and needed a lot of computer horsepower.

Recently, things have changed – quite dramatically – for LSI and LSA-based solutions. First, the computer horsepower and, more specifically, the memory that these engines need for efficiency (typically they load the entire index into memory) is inexpensive and readily available. In the 1980s memory was measured in kilobytes; today, it's measured in gigabytes. Second, 21st-century mathematicians and engineers have figured out how to apply LSI across a wide range of challenges (it turns out LSI isn't limited to text; it can handle voice and video and even mixed media); as a mathematical solution it is very pure.

LSI works on the idea that words (or entities if the material is non-textual) expressing the same or similar concept will typically be used together – co-located. Like Bayesian, LSI works best with a large corpus, as this technology virtually learns conceptual relationships as it indexes documents. The benefit to this approach is that LSI does not need any outside structures like dictionaries and thesauri to work its magic. The other unique feature of LSI is that its ability to learn conceptual relationships between words and documents can also be focused on learning conceptual relationships between languages. In other words, LSI can train itself on an identical set of documents translated into different languages and it will figure out, all on its own, that “*Comment allez-vous?*” in French is the same concept as “How are you?” in English. When used in cross-lingual applications, LSI allows searchers to input queries in, say, English, yet find relevant documents in French and German *without prior translation*. One LSI vendor, Content Analyst Company, even has a training space for their LSI engine. The training space is used for things like cross-lingual indexing but is segregated from the documents being searched so that the engine is easily trained for a multitude of search tasks without polluting the search results.

LSI also comes at a price: the use of advanced math with vectors being drawn across hundreds of dimensions requires robust computing power. Today's computers – even desktops and laptops – are thankfully up to this task. The other necessity for LSI is RAM memory – lots of it. Most LSI users are running 64-bit operating systems and will load their servers up with memory to get the optimum performance from these engines. Again, the continual drops in memory cost and increasing RAM footprints in modern servers have made LSI a worthy contender in advanced search technologies.

Making Choices

So which technology should you use? The answer is that it just depends. What you are doing and what kind of outcome you need very much dictates the kind of environment you should consider. Companies that want to “index our entire two petabytes of corporate information” haven't really thought that through. What do they want to find? Who needs to find it? What is their budget? How disciplined are they? That's like saying “I want a big car, and I

TURNER, continued

want it to go fast” and then agonizing over that vehicle’s fuel consumption and mechanical intricacy because you bought a race car but you’re using it as a daily commuter.

Any search engine depends upon its index – add new information and you need, at least eventually, to rebuild that index. Multiple search engines means multiple indexes. Unlike keywords that are static and have no meaning attached in the search world, conceptual search engines find and identify concepts from the whole text being indexed. Throw in enough new terms, and new concepts will emerge. Most engines can rebuild an index while still allowing users to perform searches, but those new concepts won’t emerge until that re-index is done. Again, it comes back to what you are trying to accomplish.

Enterprises are wise to take a page from Google if they are assembling their own search environments: Use the best appropriate technology for the specific task, and don’t be afraid to have multiple technologies in your shop. One size doesn’t fit all. Vendors are increasingly tuned into this as well. Industry giant Microsoft, who has its own proprietary search technologies, integrates its FAST subsidiary’s advanced search into a variety of products. NLP vendor Cognition is as well known for its industry-specific word lists as it is for its NLP search product. And CAAT, a conceptual search engine from Content Analyst Company, is shipped to customers tightly coupled with dtSearch®, a Boolean and keyword search engine.

Which technology is better? The fact that this debate has raged for nearly a decade ultimately means that they are all worth considering for next-generation search requirements. Each has its strengths and weaknesses and nuances where each outperforms the other. What is far more significant is how the technology is implemented. A fantastic search engine can be quickly hobbled by an inadequate user interface. A simple, elegant UI (user interface) will quickly be dismissed if it lacks the expected horsepower beneath.

There are certainly areas where one technology outshines the others – and depending upon your industry, one or two of these technologies may dominate. That popularity should not be the guiding consideration, however. What is of paramount consideration is the degree to which the implementation satisfies the requirements of your user community—or if you’re creating your own interface, your development specs. Search needs to be a tool, and next-generation search is no different. The deployment of any search solution needs to be in lockstep with your users’ expectations, skill levels and work requirements. Those requirements put the onus squarely back on the vendors. Technology for its own sake rarely satisfies. Make sure your vendors understand your expectations, make sure you and your users are comfortable with the usability of their solution and make sure that deploying it will be seen as a solution, not another issue in your users’ daily work lives. ■

Resources for Further Reading

For more information about NLP (natural language processing), visit the following *Wikipedia* page: http://en.wikipedia.org/wiki/Natural_language_processing

For more information on Bayesian inference, the following *Wikipedia* page is very helpful: http://en.wikipedia.org/wiki/Bayesian_inference

For details on latent semantic indexing, the appropriate *Wikipedia* page is http://en.wikipedia.org/wiki/Latent_semantic_indexing