Understanding How Search by Meaning Finds What Really Matters

How semantic similarity can augment search applications with human-like intuition

BABELSTREET

Introduction

Searching a large collection of documents — or the internet itself — by keywords remains a uniquely human operation. Search engines return so many results that human judgment is required to identify the ones that are most relevant to the search query.

Until recently, keyword search was the only option, even though it returns many irrelevant results and misses important ones. This is because keyword search only looks for word matches without considering the meaning of those words.

In contrast, searching by meaning — also known as semantic search — uses the actual meaning of words to return relevant results in situations such as eDiscovery, government intelligence, adverse media screening, and any search that needs to find what matters most from a mountain of data.

Semantic search eliminates off-topic results that happen to include the keyword and instead finds results that match the meaning — and intent — of the searcher.

Integrating semantic similarity capabilities, such as those found in Babel Street Text Analytics, into an application increases the effectiveness of:

- Adverse media screening
- Data triage for government intelligence
- Cross-lingual search
- eDiscovery and search for relevant documents
- Reducing the pool of data with document clustering and deduplication

This technical brief describes the approach that Text Analytics uses for word embeddings — the technology that powers semantic similarity for semantic search — to retrieve the intent and context suggested by the keywords. You will follow the steps Text Analytics takes to encode the semantic meaning of words as mathematical expressions (vectors) and see how to integrate semantic similarity into your search efforts.

Semantic search eliminates off-topic results that happen to include the keyword and instead finds results that match the meaning — and intent — of the searcher.

Going beyond keyword search

Suppose you are a compliance officer with a financial institution and a politically exposed person (PEP) — usually wealthy or well-known thus susceptible to corruption — wants to open a new account. Doing business with someone who has committed a predicate offense may exceed your institution's risk appetite, so you turn to your screening tools to find any adverse media coverage on the individual.

In an era of text-based search engines, you perform a keyword (Boolean) search on the person's name and the words "money laundering." Then, as you study the search results, you include and exclude other keywords to refine your search.

Over time, your keyword search phrase can easily grow quite long. For example:

```
"NAME" AND (launder! or criminal
or crime or fraud or breach or
illegal or civil or negligen! or
murder or complaint or charge! or
accus! or acquit! or alleg! or
arrest! or guilty or testif! or
bribe or conspir! or raid or FBI or
investigation or violat! or indict!
or litigat! or default or seize or
foreclo! or lawsuit or terror! or
porno! or threat! or plead or plea
or scandal! or bankruptcy or scam or
tort or misconduct or corrupt! or
misrepresent! or deceptive or scheme
or evasion or arraign!) NOT laundry
```

The problem with this approach is that it depends on matching your search keywords to individual words in the documents. It also depends on your ability to determine and add keywords to the query as you go.

The search engine is designed to compare the words in each document to your query keywords. However, you would prefer that it compare the meaning of each document to the keywords. Those results would show high-priority, semantically similar documents, with two important differences from Boolean search results:

- Some results would not contain the keywords you had entered
- Some results would contain keywords you had not thought to look for

Nevertheless, those results would still be useful.

It's easy to determine which words a document contains, then identify documents with the same words. It's more difficult to determine what a document means and identify documents that are semantically similar. To do this, you must first convert text (words and sentences) to numerical representations. You can then perform calculations on those numbers to objectively compare how close they are in value. The closer they are, the more similar they are in meaning.

Word embeddings: A path to semantic similarity

Word embeddings transform words into vectors: numerical representations that approximate the conceptual distance of one word's meaning from another. They pave the way to identify semantic similarity and to perform semantic search, where a user can search on meaning rather than on a keyword.

The idea of expressing the meaning of words as mathematical vectors appeared in the 1960s with Cornell University's SMART system.[1] Most text information retrieval models since the 1980s have been based on vector models of text. However, it was not until 2013, with the introduction of new algorithms, that the task of creating word vectors became much more scalable and practical.[2] A dense/distributed vector representation can encode word semantics in ways that more naturally generalize to words that have not been seen before.[3]

An example of calculating vectors

At the heart of word embeddings is the mathematical concept of a vector. Vectors are commonly used in representing relationships, like locations on the earth's surface expressed as latitude and longitude:

- Paris [48.86, 2.21]
- Marseille [43.28, 5.24]
- Lyon [45.76, 4.76]

Each of those three cities can be represented by a vector containing two dimensions (latitude in degrees from Greenwich, longitude in degrees from the equator). As shown in Figure 1, simple algorithms can use the vectors to calculate their proximity (that is, geographic similarity) to one another:

- Paris-Marseille: 660 km
- Lyon-Marseille: 276 km
- Paris-Lyon: 393 km

Vectors also serve in calculating the distance from (dissimilarity to) a geographically unrelated place, like New York City [40.70, -74.26]:

- Paris-New York City: 5,851 km
- Lyon-New York City: 6,160 km
- Marseille-New York City: 6,318 km

Figure 1: Using vectors to represent geographical points

How word embeddings use vectors to represent words

Take the example of the words "writing," "pen," "pencil," and "fox." It's possible to graphically represent similarity and dissimilarity among those words, as shown in Figure 2.

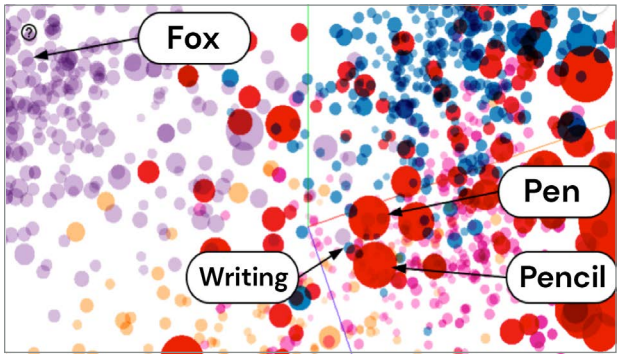

Figure 2: Graphical, two-dimensional representation of vector space showing relationships among words

Given that semantic similarity/dissimilarity can be depicted, how can it be made objective?

Word embedding is a way to quantify semantic similarity and dissimilarity. As in the example of representing cities by vectors, word embedding converts the meaning of words into mathematical vectors. Algorithms can then calculate semantic "distance" among them in that vector space. Word embeddings make it possible to quantify how "pen," "pencil," and "writing" are semantically similar, and how "fox" is semantically dissimilar to all of them.

Whereas each vector in the geographic example above comprises two dimensions (latitude and longitude), each vector in the Text Analytics approach to word embedding mathematically describes where a word lies relative to the "word universe" (the set of words upon which the word embeddings model was trained).

For the best vector representations, the "word universe" must be in the same topical domain as the text you want to process. If you are processing text about microchip manufacturing, the word universe must be too. Product reviews of potato chips won't do.

Calculating vectors for word embeddings

The traditional way of representing words mathematically is to analyze an extremely large corpus of textual documents to see the context in which each word frequently appears. The result of the analysis is a co-occurrence matrix. For the tens of thousands of words in the vocabulary of the corpus, each word has a row and a column in the matrix. At the intersection of row and column is the number of times the words occur near others in the corpus.

Table 1 shows a very small detail from a typical co-occurrence matrix: statistics for words related to "noisy" in a corpus of text. "Noise" and "loud" occurred near each other 14 times; "noise" and "driver" did not co-occur.

Word embeddings transform words into vectors: numerical representations that approximate the conceptual distance of one word's meaning from another.

	very	too	loud	low	room	night	engine	driver	...
noisy	10	12	10	11	15	7	10	0	
noise	11	4	14	12	3	2	16	0	
sound	2	5	13	12	6	5	10	0	
drills	5	3	9	7	6	1	9	0	
cake	4	2	1	1	2	3	0	3	
table	1	2	0	3	4	2	0	0	
car	2	3	10	2	1	0	9	12	
...									

Table 1: Co-occurrence matrix (partial) for words related to "noisy"

Software tools can generate a co-occurrence matrix relatively easily from a large corpus of documents. Each row of the co-occurrence matrix in Table 1 corresponds to a basic word vector. For instance, the vector of "noisy" (10, 12, 10, 11, 15, 7, 10, 0, and so on) represents its meaning within the entire vocabulary of this corpus. The vector for "noisy" is more similar to the vector for "noise" than it is to the vectors for "cake" and "table." Words with similar meanings have similar vectors because, at scale, they often appear in the same context (near the same words).

A vector, then, is a list of values, with each value representing one dimension in a multi-dimensional space. In the latitude-longitude example above, each value represented one dimension in a two-dimensional space. But to exhaustively model the vector space of a vocabulary, or even one specific vocabulary within a corpus, would take tens of thousands of dimensions. Each vector would comprise tens of thousands of values representing the relationship of every word to every other word.

That would be computationally prohibitive and require a corpus so large as to be unwieldy.

Distributed representations: Word embeddings in 300 dimensions

Instead of struggling with vectors with millions of dimensions (the size of the entire vocabulary), Text Analytics uses techniques to compress those vectors to 300 dimensions, which turns out to be the best balance between reducing computing requirements and still making meaningful semantic comparisons. Collapsing several dimensions into one is done by groupings such as gender, color, or aspects of objects (e.g., motorized objects or objects used in cooking). At that threshold there is enough information to make semantically valid comparisons with reasonable computing requirements.

The result is that, in the product's word embeddings, each dimension of the 300-dimension vector is a combination of many of the original dimensions. That makes each vector a compact, powerful representation of every word.

The vectorization of text is based on training models using a huge collection of raw documents from topics and domains *that match the text it will process*. Word embeddings are accurate only within the context of that training corpus — mechanical, medical, financial, popular culture — which defines the meanings of the words in it.

Text Analytics then assigns vectors, such that the words with similar meanings ("pen," "pencil," "writing") have similar values. The degree to which two words are similar is represented by the similarity of their corresponding vectors, as in Table 1.

That is how Babel Street Text Analytics detects similarity/dissimilarity among individual words. How does it handle documents and long text?

Text embeddings: Semantic similarity among entire documents and articles

Text embedding is the computational technique Text Analytics uses to represent the meaning of an entire document, so that a software application can detect similarity/dissimilarity to other documents. Text embeddings are based on the idea that words and phrases whose vectors are close in vector space will also be close in meaning, regardless of the language they are written in.

How Text Analytics measures similarity

Babel Street Text Analytics uses the cosine similarity between word vectors to measure semantic similarity.

In a mathematical context, cosine similarity measures the extent to which two vectors point in the same direction in a vector space. As a concept in information retrieval, it measures the similarity of two pieces of text (such as words, queries, sentences, and documents). Text Analytics first transforms the text into vectors in its 300-dimension space, then applies cosine similarity to determine how similar (or different) they are.

Text Analytics returns the vectors used for calculating cosine similarity, which can be reported in a range from, say, -100 (terms are exact opposites) to 100 (terms are identical). In the midpoint of such a range, a score of 0 indicates that the terms are not related.

As shown in Figure 2, word embedding highlights how "writing" is similar to "pencil," but dissimilar to "fox." Text embedding can show that the Gospel of John is similar to the Gospels of Luke and Mark, and that *Great Expectations* is dissimilar to all the gospels.

How Text Analytics handles text embedding

Text embeddings are the mathematical representations of text as vectors. The semantic vectors endpoint in Text Analytics works on any length of text by first breaking it into words, then calculating a 300-dimension vector for each word. It assigns a weight to each resulting vector based on frequency and an average per dimension. A single, final vector represents the entire text, which the product then compares to its 300-dimension vector space for semantics.

Inside the vector space, themes and topics are grouped by similarity. For instance, one area is mapped to automotive terms, another to financial terms, another to engineering terms, and so on. Text Analytics takes the input text and assigns a vector to it, which represents where the text lies semantically in the vector space.

When integrated with a software application designed for searching text, Text Analytics can cut the time required to find semantically similar documents and boost search accuracy.

Semantic similarity across languages

Text embeddings and semantic similarity also play a role in searching for similar documents and articles across languages, in processes like multilingual eDiscovery.

In the Text Analytics vector space, terms and concepts in each language's vector space are aligned so that the English words "writing," "pen," and "pencil" are in the same "space" as the French "écriture" ("writing"), "stylo" ("pen"), and "crayon" ("pencil").

The vector space is currently available in all major strategic languages, based on models trained in each language. Even across languages, the vectors for individual terms and concepts with similar meanings have similar values. In this way, Text Analytics is able to map semantic similarities and dissimilarities across languages, as depicted in Figure 3.

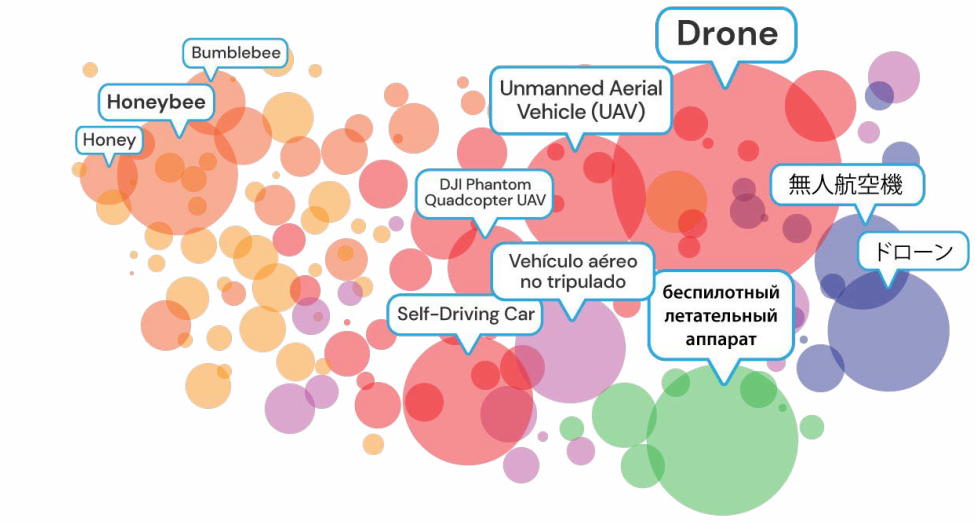

Figure 3: Multilingual, graphical, 2D representation of "drone" and related terms in the Text Analytics vector space

How to integrate semantic similarity into keyword search

Through RESTful endpoints, Babel Street Text Analytics provides tools for generating and using text vectors to identify semantically similar words, in a single language and across languages. Its three use cases are described below with sample application use cases.

1. Finding related terms

This endpoint takes an input term, such as "money," and returns semantically similar terms formatted in JSON. The sample below shows the first 10 results in the vector space that are similar to "money," illustrating the broad context that semantic search encompasses:


```
{
 "similarTerms": {
   "eng": [
   { "term": "money", "similarity": 1 },
   { "term": "funds", "similarity": 0.70577776 },
   { "term": "cash", "similarity": 0.68805909 },
   { "term": "monies", "similarity": 0.6602034 },
   { "term": "donations", "similarity": 0.57579613 },
   { "term": "billions", "similarity": 0.54842752 },
   { "term": "savings", "similarity": 0.53408724 },
   { "term": "financing", "similarity": 0.5261488 },
   { "term": "largesse", "similarity": 0.52415174 },
   { "term": "proceeds", "similarity": 0.51997113 }
    ]
  }
}
```


Your software application can store those results in its search index and use them to expand queries or offer alternative keywords.

You can also submit arguments for multiple result-languages. For the English input term "spy," the example below shows the first related term in German, Japanese, and Spanish:


```
{
  "similarTerms": {
    "deu": [
      {
        "term": "Spion",
        "similarity": 0.50051737
      },
    ],
    "jpn": [
      {
        "term": "スパイ",
        "similarity": 0.5544399
      },
    ],
    "spa": [
      {
        "term": "espía",
        "similarity": 0.61295485
      },

    ]
```


Use case for related terms

Searching on related terms is helpful in matching company and product names, which tend to be used ambiguously, both orally and in print. Semantic similarity aids in detecting near matches because, for example, the company name "PennyLuck Drugs" is more similar to "PennyLuck Pharmaceuticals" than it is to "PennyLuck Landscaping" (see Figure 4).

Likewise, the product name "Vacation Chocolate Bar" is more similar to "Vacation Candy Bar" than it is to "Vacation Estates Cocoa Butter Soap."

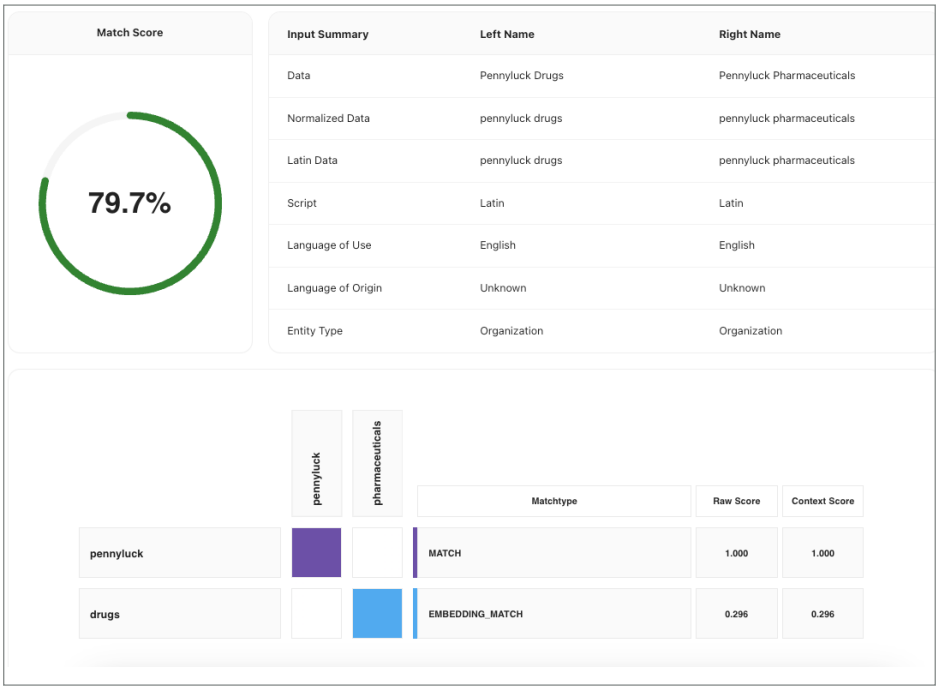

Figure 4: Semantic similarity in name matching. Screenshots are taken from Match Studio

Word embeddings and event extraction

Text Analytics leverages word vectors to extract events from natural language. Event extraction is a subset of entity extraction, which identifies important data points (entities) in unstructured text, like news, webpages, and text fields. Text Analytics is trained to detect patterns of events, with building blocks referring to specific roles in the event. Figure 5 shows two sentences of the same pattern. Text Analytics identifies actor roles: "husband," "woman," and "by Springfield police," "by U.S. Marshals." The text highlighted in blue refers to the keyphrase used to detect the event described: arrests.

Using text embeddings, Text Analytics looks for sentences that are semantically similar to examples of events it saw in the human-annotated training data.

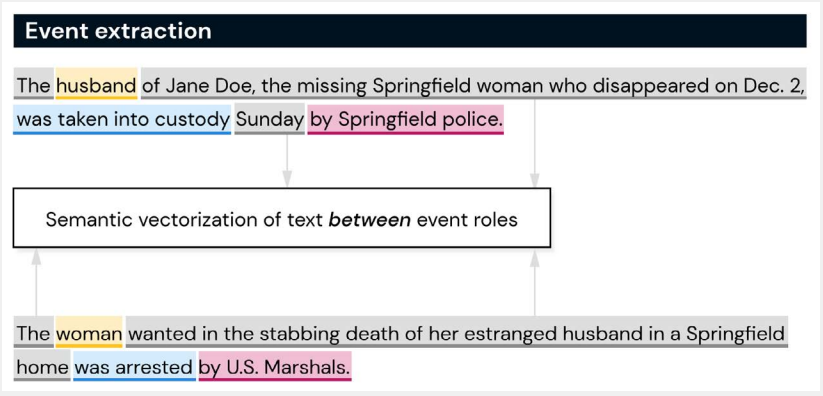

Figure 5: Event extraction leverages word embeddings to find events even when worded differently than in the training data.

Use case: Adverse media screening

Adverse media screening is an area of financial compliance that deals with detecting politically exposed persons and their close associates who hold prominent public positions and thus are more susceptible to bribery or corruption. Financial regulations worldwide require financial institutions to search through news and media about potential or existing PEP customers for signs of wrongdoing.[4]

The difficulty is that since PEPs are prominent, they appear frequently in the news, and often not for wrongdoing. Semantic similarity combined with other NLP can reduce the number of false positive matches.

- Semantic similarity can limit articles *to those about wrongdoing*.
- Entity extraction and linking check that the person in the article is the PEP being screened and not another person sharing the same name.
- Event extraction further confirms that the PEP is the one committing the crime, rather than an incidental figure.

Remember that the latter two technologies also leverage word embeddings!

2. Finding similar documents

Imagine that you must review thousands of electronically stored documents and identify the ones relevant to a given news article, legal case, or industrial process. In that kind of eDiscovery, the ability to detect documents related by theme or topic can save you weeks of expensive human labor. For several use cases, Babel Street Text Analytics offers business-ready workflows for detecting documents that you may include in (or exclude from) the queue of documents requiring human review.

Searching on query terms

You do not need to know the exact word or phrase on which to search; you merely need to know terms that describe the concept. In the demo user interface below, Text Analytics queries a collection of documents for "economy currency money" and returns similar sentences and paragraphs (see Figure 6).

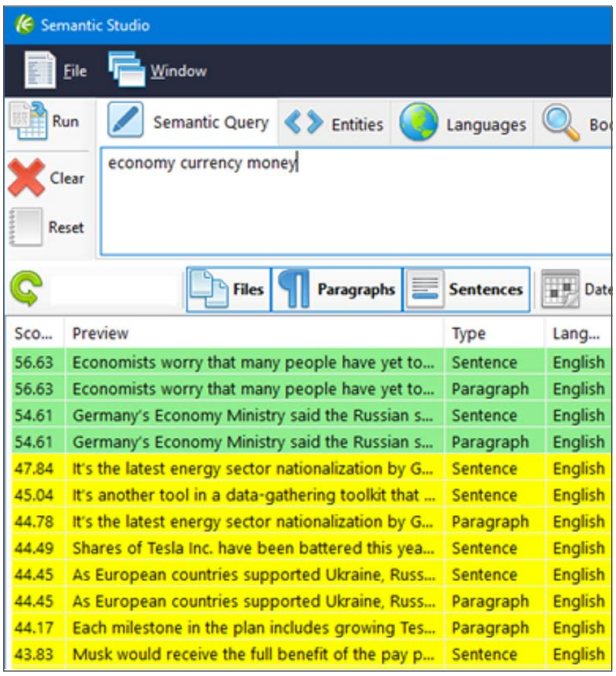

Figure 6: Using semantic search to find content relevant to the keywords "economy currency money"

Note that the results from this semantic search do not necessarily *contain* the terms "economy," "currency," and "money," but they *share meanings* with those terms. Semantic search finds documents with relevant content, even if the documents do not contain the exact keywords used in the query.

Searching using a document as the query

You can input a document to Text Analytics and semantically compare it to a collection of documents (for example, in the same directory). The resulting score and ranking enable you to quickly see near or exact duplicates and gauge the semantic similarity of other documents.

The left side of Figure 7 shows that the input document (selected in blue) contains the text "Blackouts worsen…." The right side of the figure shows the similarity scores for other documents in the same directory.

Naturally, the input document itself returns a score of 100. Other documents in the directory return similarity scores of 92, 71, 43, and so on, down to negative scores that indicate dissimilarity to the input document.

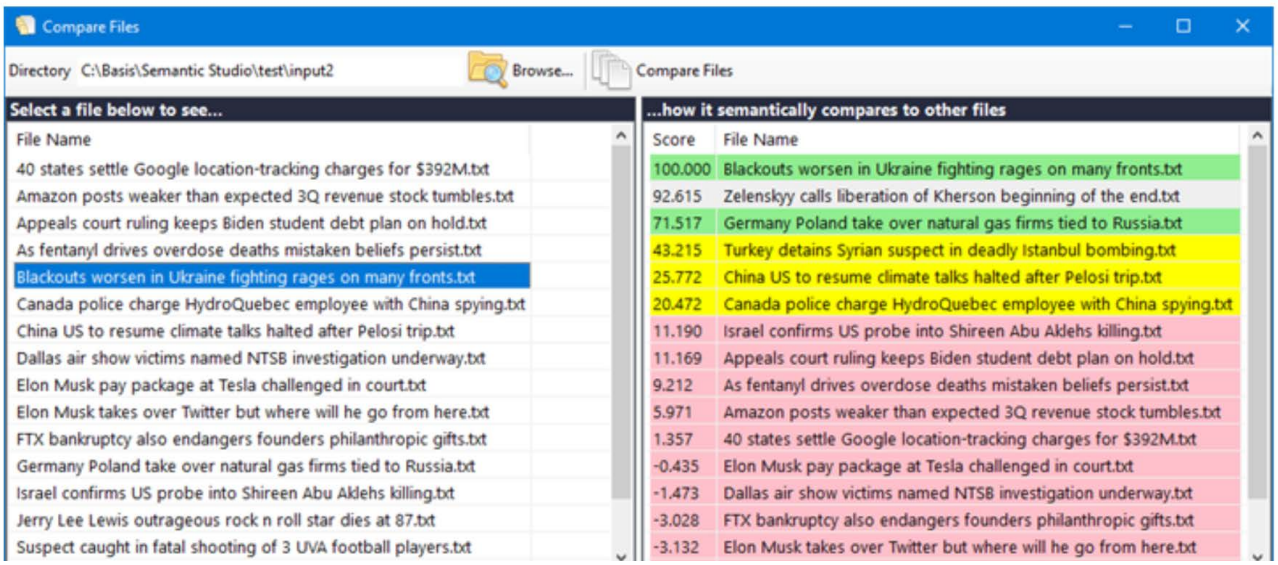

Figure 7: Calculating similarity of an input document to a collection of documents

Use case for finding similar documents

The ability of Text Analytics to calculate a similarity score and rank documents in a collection is also useful when you're searching for duplicate documents. In another application of text embeddings and semantic search, Text Analytics can detect plagiarism, even when superficial changes to sentences would foil typical plagiarism checkers.

Finding related terms and documents across languages

Text Analytics maps semantic similarities and dissimilarities across all major strategic languages. That is because terms and concepts with similar meanings are aligned across languages so they are located in the same areas of vector space. Figure 8 illustrates the cross-language results generated by Text Analytics against a one-word query — "missile" — supplemented with a contextual sentence, "But Kwon said…,":

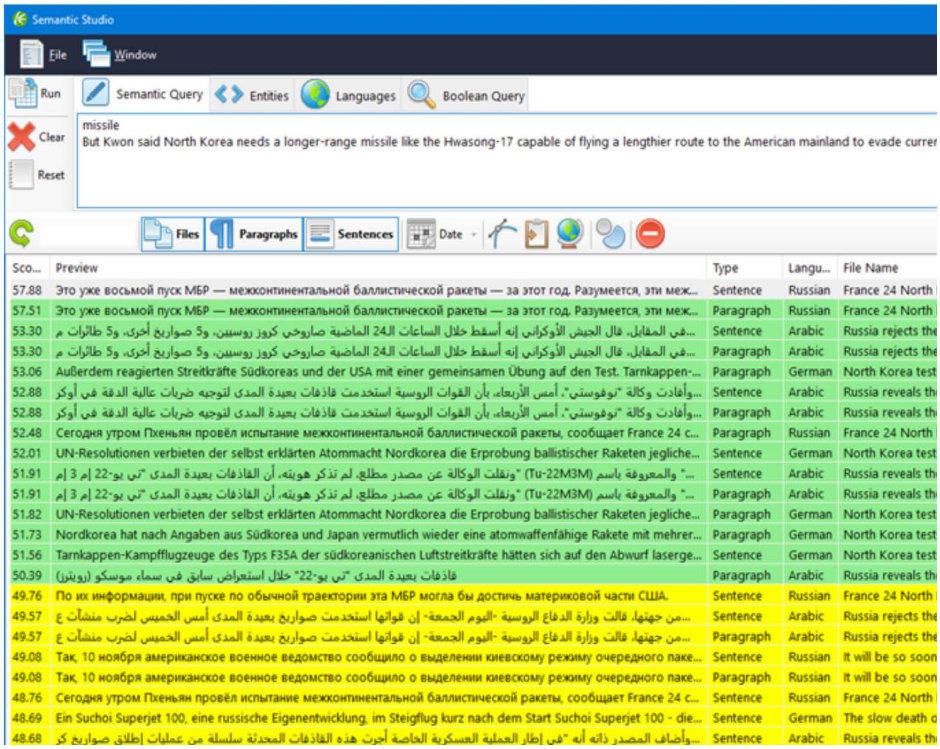

Figure 8: Using semantic search to find relevant content across languages

The demo user interface shown in Figure 8 highlights several advantages of using semantic search with Text Analytics:

- Even without English-language results (excluded for this query), Text Analytics identifies numerous selections in Russian, Arabic, and German.
- It retains the similarity scores.
- Because Boolean search engines match words instead of meanings, they would not have found the exact query text in these documents and would have omitted them from the results.
- Text Analytics also includes a Language Identification REST endpoint. Here, its result indicates the source language of each search result.

Use case for finding related terms and documents across languages

Now suppose you must review thousands of documents in multiple languages and identify the documents relevant to a particular case. It's a common problem in government intelligence, where a monolingual analyst is tasked with triage of information in multiple languages and not enough analysts are available with fluency in every language. It's also common in legal eDiscovery, where a monolingual attorney wants to reserve expensive translation expertise for only the highest-priority documents.

Multilingual eDiscovery requires either a staff of analysts/attorneys competent in all target languages, or language professionals who can translate the documents (or search queries) into

the target languages. In the case of search, a linguist must also review and vet search results. But as described above, ordinary search is designed to match text rather than meaning, so even with linguistic expertise, the process could easily miss important documents.

With Babel Street Text Analytics, monolingual users can vet large quantities of documents in multiple languages and identify the ones relevant to a given search term. It relieves the bottleneck of multilingual expertise in government and industry.

Semantic search is transforming mission-critical search today

Boolean search has been the king of search for a very long time, and semantic similarity is its first heir apparent, capable of delivering search results significantly closer to human intuition. Word embeddings address the weaknesses of keyword search where:

- Irrelevant results occur because keywords contain too little context and can have multiple meanings.
- Missed results occur because an exhaustive list of keywords is rarely possible.
- Searching with phrases may provide additional context, but yield fewer results because there's less likelihood of an exact word-for-word match.

Adopters of semantic similarity are building a competitive edge by enhancing mission-critical search functions in domains such as multilingual eDiscovery, cross-lingual search, and data triage of government intelligence.

Endnotes

[1] Wikipedia, retrieved Nov. 28, 2022. https://en.wikipedia.org/wiki/SMART_Information_Retrieval_System

[2] Wikipedia, retrieved Nov. 28, 2022. https://en.wikipedia.org/wiki/Word_embedding

[3] Hoffman, Garrett "How neural networks learn distributed representations" Feb. 13, 2018, O'Reilly. https://www.oreilly.com/content/how-neural-networks-learn-distributed-representations

[4] Comply Advantage "PEP Screening Requirements Around The World," Jan. 16, 2023. https://complyadvantage.com/insights/politically-exposed-persons-peps-screening-requirements-around-the-world