

SCIENTIO

ConceptMap White Paper Comparing and indexing large sets of documents

Andrew Edmonds PhD (andy@scientio.com)

Scientio LLC, July 2, 2006, Version 10

Introduction

Scientio have extensive experience of text mining and have been researching concept based text mining for several years. We've developed techniques that merge elements of the main threads of text mining research, combining natural language processing with statistical techniques. This research has resulted in our new product, ConceptMap, which is an infrastructure software component that can be used to efficiently and rapidly create a signature identifying the concepts used in a document. Because these signatures are numeric, they can be effectively 'plotted on a map' and documents containing similar concepts, which are thus similar in topic and content, can be grouped and indexed efficiently.

Key features are:

- Documents are represented by a small, fixed length numeric vector used as a signature generated from the concepts in a document.
- Signatures form a space in which similar documents can be detected with great computational efficiency.
- ConceptMap includes support classes to create, index and locate signatures, and thus similar documents in $\log(N)$ time.
- The measure of similarity can be adapted to have a range of characteristics including finding near duplicates and variants of documents, or finding documents with similar topics.
- Unlike other algorithms, the similarity measure ranks documents for similarity in a way that corresponds closely to human responses.
- The algorithm used is parallelizable and scalable. It has been used on a production database of 2.5 million documents.
- For anti-plagiarism applications, the software, working as it does on the concepts in a document, is immune to thesaurus based substitution.
- Although the software is language-specific it can be easily ported to new languages. Off the shelf languages are currently English and Spanish.
- The software is used in production and has been verified extensively for accuracy in finding duplicate and near duplicate documents.

Background

The research work underpinning ConceptMap dates from 2003, but it wasn't until 2005 that we were contacted by a customer with a difficult problem that presented an opportunity for making use of this research. This was a problem the customers had expected to be able to solve easily, and were surprised to find no off-the-shelf solutions for. They wanted to compare new documents being loaded into a database with existing ones; to make sure the new documents were not

duplicates or near duplicates of those already in the database. The customer wanted the comparison to ignore document formatting, typos and minor changes.

If you want to find two or more documents that are exactly the same, there are many simple 'hashing' algorithms that create a signature number from a document. Documents with the same hash code are likely to be identical. However two documents with slight differences will have dramatically different hash codes and so hash codes cannot be used to locate similar documents.

There are several comparison algorithms that one might use. You might, for instance, perform a word frequency analysis and then perform some kind of similarity measure between the word-vectors produced. However, with a database of a million documents this kind of pair-wise comparison would have to be done perhaps half a million times every time you wanted to add a new document!

Ideally, to create a practical solution, one would try to locate each document in some kind of Euclidean space, based on content, and then locate any incoming document in the same space. This would enable the user to be sure there were not any near neighbors that might be duplicates. One common solution is to use the 'word space', i.e. to use the space of word frequencies, to compare documents in. Unfortunately the English language contains more than 40,000 words, so this word space has enormous dimensionality, and is thus computationally difficult when comparing large documents. In the light of this difficulty, algorithms (e.g. Microsoft's 'shingling' algorithm) have been developed that use statistical sampling to create a range of characterizing measures for documents, and these measures are then used to form a smaller set of dimensions that are more computationally tractable. However these measures make use of features of the documents that are not necessarily related to its meaning and content; so, for instance, two drafts of the same letter that contain different wording but convey the same ideas might not be matched.

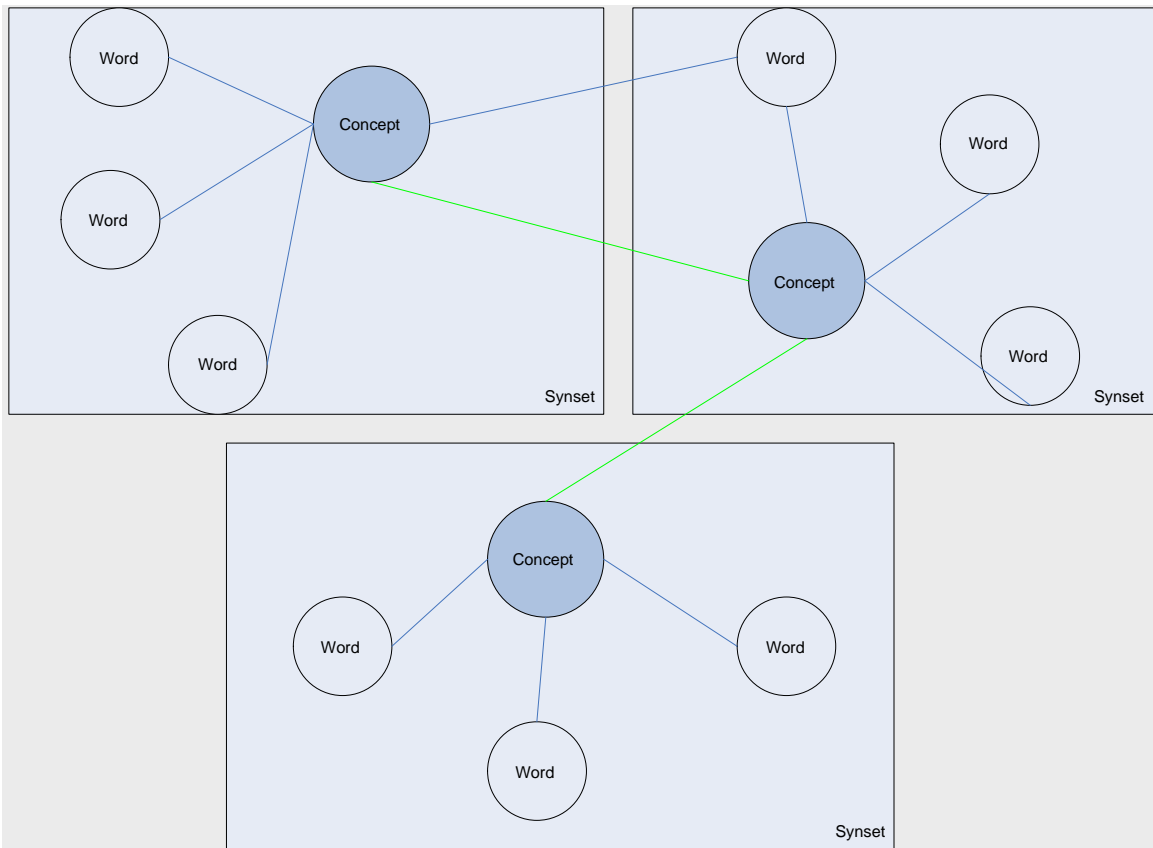
As in statistics, different distance measures have different characteristics. Measures that use word frequency statistics may not correspond well with how a human would rank documents for similarity. In almost every application we can think of, the most useful measure, and the one a human user would find most intuitive, would involve generating a distance measure based on the similarity of the *concepts* within a document, and not the choice of words.

Our research and the availability of computer models of the world's major languages offer a new and novel solution.

Concept mining

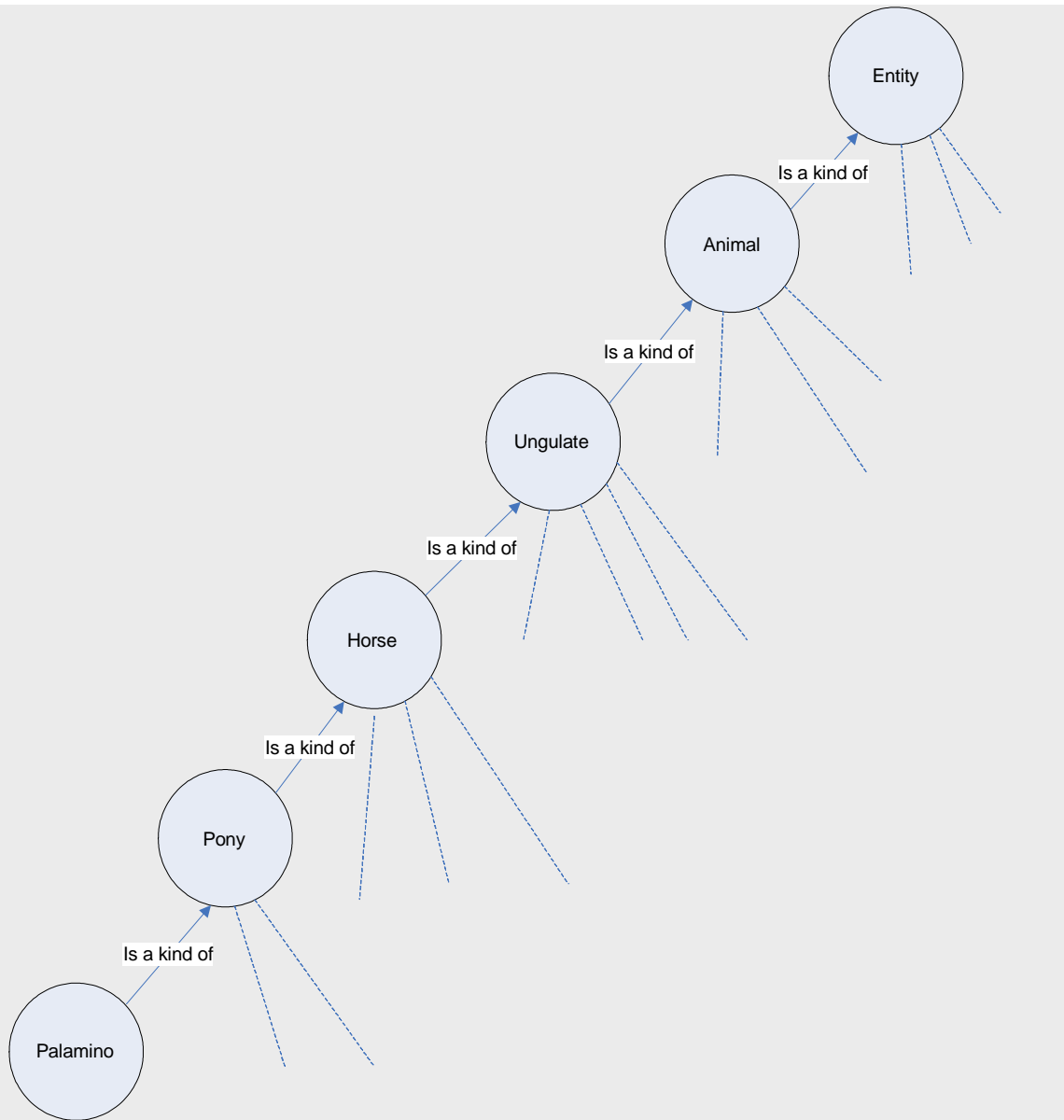
Human communication is about, in its simplest form, transferring concepts from one person to another. Although it's not possible for us to get inside each other's minds and verify this, in general it seems that there is a common core set of concepts used by all humans.

In recent years models of individual languages, called WordNets have been created. Princeton University created the first of these for English. A WordNet combines the functions of both a dictionary and a thesaurus, and annotates this with copious cross referencing of various relationships.



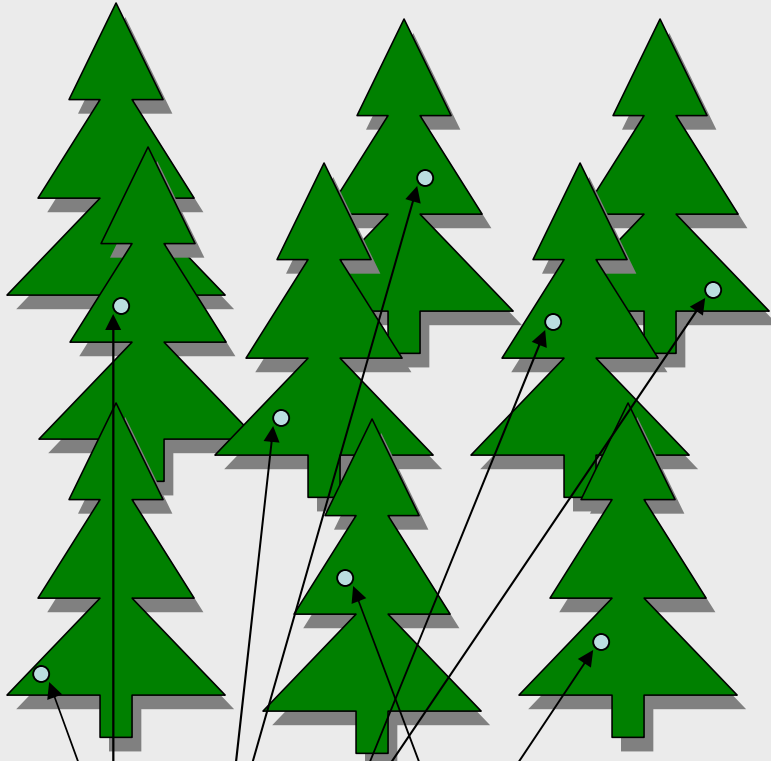
The fundamental unit of a WordNet is a *synset*, a collection of words of the same grammatical type linked by a common concept. Broadly speaking we can say that synsets embody a particular concept. Words are frequently ambiguous, possessing different senses, and can be associated with more than one synset. Synsets are connected to other synsets by a variety of relationships. Some of these relationships form hierarchies. One concept can be part of another; for instance the concept 'finger' is part of the concept 'hand' (this is known as meronymy). Also one concept can be a specialization, or kind, of another concept, for instance a car is a kind of vehicle (this is known as hyponymy).

All noun concepts form part of one of surprisingly few hyponymy hierarchies. English, for instance, according to the WordNet created by Princeton, has only nine hierarchies. In the diagram below we show how a section of one of these hierarchies for the word/concept *Palomino*.



The presence of these hierarchies is very useful. We can use them to simplify dramatically the concept information in a document. This enables us to create numeric measures, or signatures, of a piece of text that enable us to position it like a map reference in what we call 'Concept space'.

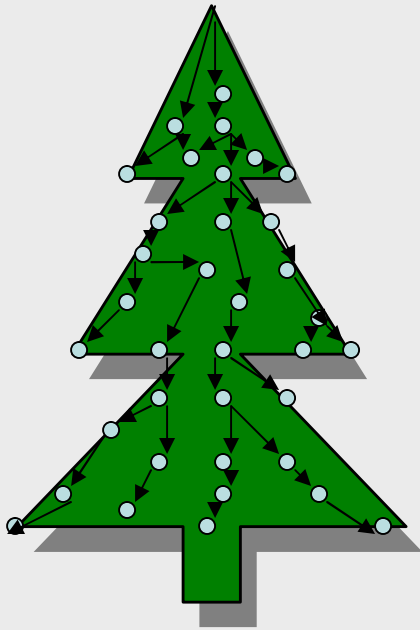
In this process we take words in a document and associate counters corresponding to each word on the concept hierarchies, represented in this diagram as a *set of trees*.



The quick brown fox jumped over the...

Typically each word can be associated with more than one concept, and so there is ambiguity in converting from words to concepts, but our experience is that the process of creating a statistical measure over the many concepts in a document tends to average out these ambiguities.

A single 'concept intensity value' can be created for each tree recursively:



And from these we can create a numeric vector signature that can be used to characterize an individual document.

Our ConceptMap software can rapidly create signatures for English and Spanish language documents of any size. These signatures have the following characteristics:

- Signatures consist of 9-16 floating point values, depending on language.
- Signature size is independent of document length.
- Signatures are not influenced by formatting or white space.
- Since signatures are numeric, calculating the distance between two signatures, or plotting them in space to identify clusters is computationally simple.
- Documents that are close in concept space tend to be close in meaning and topic.
- Signatures can be easily extended to add weight to other document characteristics, such as size or number content.
- For anti-plagiarism applications, documents that have been modified using thesaurus based substitution of words, or by the re-arrangement of sentence or paragraph order will still generate near or identical signatures to the originals.
- Various words, including proper names and misspellings will not be recognized in the processing of the document for a given language. ConceptMap can output a list of these words found in each document, and the possibility exists to create a hybrid system combining conventional word search/word frequency and concept based techniques.

Practical example 1 – Finding similar documents in large corpora

The process of finding two similar or identical documents out of a large database is greatly simplified by using signatures. Assuming we create a signature of each document once, when it is added to the database, and of course again if it is ever modified, we can compare the signatures for similarity, instead of comparing the documents themselves.

We might want to do something more sophisticated, and find, given a new document, the other 'n' most similar documents in the database.

There is a very efficient algorithm to do this based on structures called KD trees. These enable the user to find near neighbors to any given numeric vector in logarithmic time.

This means that as the size of your database increases, the time taken to find the neighbors, of course, increases too, but at a very slow rate. Each time you double the size of the database the processing time goes up only by a constant amount.

ConceptMap contains both the signature generating code, and a full and efficient implementation of KD trees. Some databases, such as those from Oracle, include native support for indexing vectors of numbers, so the KD tree code may not be required in every application.

Performance

The software has been tested extensively on large amounts of text in both English and Spanish. One of the tests we tried was using a Reuter's news items data set. We use this data set frequently in text mining testing. It contains 8599 samples of text, each around a paragraph long, containing news stories from the mid 80s, averaging 150 words each. We started by extracting the stories, and then running each through our software, transferring them to "concept space".

We then used our KD tree implementation to construct an index of these documents.

Now, in order to test finding corrupted documents similar to a source document, we removed 5% of each document, first from the front, and then the rear, and in each case asked the system to find the nearest neighbors to these corrupted documents.

The table below shows where in the ranking of nearest neighbors the correct document occurred.

Rank in search by which the correct document has been found.	No truncation count	No truncation %	Truncation at rear count	Truncation at rear %	Truncation at start count	Truncation at start%
1	8599	100	7398	86.03	6118	71.14
2			7884	91.68	6875	79.95
3			8054	93.66	7182	83.52
4			8160	94.89	7401	86.06
5			8228	95.68	7532	87.59
6			8266	96.12	7640	88.84

We also looked at the time requirements for processing these documents.

Timings on a 2Ghz AMD Athlon 64 with 1Gb RAM running XP and .Net 2.0

Constructing language model	3.4 Seconds
Time to create 8599 signatures	76.71 Seconds
Time to load the KDTree with 8599 records	0.046 Seconds

Commercial application

Legis Editores, the largest legal publisher in the Spanish speaking world has, through their subsidiary IconoMultimedia SA, created a database of the entire constitutional and legal codices of the Spanish-speaking countries of South America. This consists of some 2.5 million documents. The source used contained duplication and near duplication, i.e. two or more otherwise identical documents with different formatting and spelling, or multiple near-identical documents differing only in small revisions. They have successfully used our software to create signatures for the entire database and to identify these similar documents rapidly and efficiently. In trials they have found that they can locate the nearest 10 documents in the database to any given document in less than 3 seconds. At the time of writing they have been using the software in a production environment for nine months without problems.

Practical example 2 – Detecting Web log plagiarism

Web logs are extremely popular, and well liked weblogs or *blogs* can generate substantial advertising revenue. The main effort involved in creating a weblog is the creation of content, and this has encouraged the unscrupulous to plagiarize successful weblogs and reuse the content. This, like general plagiarism of internet content can be hard to detect without very efficient algorithms. In association with the weblog registry Syndic8.com we have created a pilot application that accesses thousands of weblogs looking for plagiarized content.

Synic8 receives 'pings' or signals whenever weblogs are updated. It is a simple matter then to read the weblogs and create signatures for each content item. A plagiarism detecting hub does not have to store the content, just the signature of each item. Thus any weblog owner wishing to detect possible plagiarizers of his or her intellectual property need only register with the hub and be returned a list of similar items from other blogs to investigate.

A working example of this can be viewed on:

<http://www.scientio.com/blogplagiarism.aspx>

More general applications – adapting the signatures

One of the important virtues of ConceptMap is its ability to summarize a large document in a very small and fixed-length signature. In order to adapt ConceptMap for different applications we can modify and adapt this signature generation process.

As well as detecting similar or identical documents, we can also identify documents that are similar in *topic*. This requires a differently constructed signature, using more of the lower level elements in the concept hierarchies, and thus longer, but still very small signatures.

This could form the basis of a recommendation system, or an auto-categorizer in a topic map generation system.

Scientio has performed internal research in this area with useful results which we would be happy to discuss with interested parties.

Language support

The system we've described relies on the presence of a model of the language used in the analysis. We use the WordNet model of the English language created at Princeton University. Various other groups of linguists around the world have created WordNets for most common languages. For instance the EU has funded the creation of WordNets for most of the European languages. Our experience of porting to Spanish suggests that converting this system to a new language is relatively straightforward, taking 1-2 weeks. Different licensing terms exist for each WordNet, so we suggest you contact our sales department at sales@scientio.com to discuss your requirements.

Technology

ConceptMap is available as a .Net class library for simple integration into applications on Microsoft platforms, or via Mono on Linux/Unix/Mac platforms. A Java version is also under development.

ConceptMap V1.1 supports version 1.1 of the .Net CLR, whereas version 1.2 supports version 2.0 of the .Net CLR, and makes use of generics for improved processing speed (17% improvement).

Contacting Scientio

More details of ConceptMap can be found at <http://www.scientio.com/conceptmap.aspx>
You can contact Scientio sales on (302)-351-5663 in the US or +44-1908-766151 for our UK office. You can reach us by email at sales@scientio.com
We would be delighted to discuss OEM type applications on an individual basis.