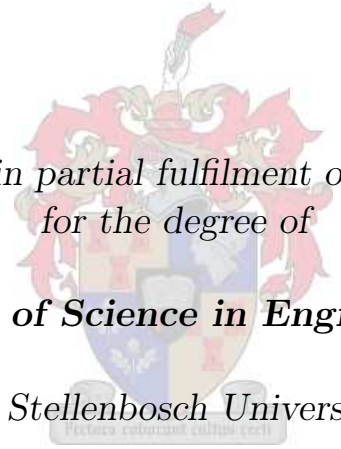
# Evaluation and Development of Conceptual Document Similarity Metrics with Content-based Recommender Applications

by

Stephan Gouws



*Thesis presented in partial fulfilment of the requirements for the degree of*

***Master of Science in Engineering***

*at Stellenbosch University*

Supervisors:

Dr. G-J. van Rooyen  Dr. H.A. Engelbrecht

December 2010

# Declaration

By submitting this thesis electronically, I declare that the entirety of the work contained therein is my own, original work, that I am the owner of the copyright thereof (unless to the extent explicitly otherwise stated) and that I have not previously in its entirety or in part submitted it for obtaining any qualification.

December 2010

# Abstract

The World Wide Web brought with it an unprecedented level of information overload. Computers are very effective at processing and clustering numerical and binary data, however, the automated conceptual clustering of natural-language data is considerably harder to automate. Most past techniques rely on simple keyword-matching techniques or probabilistic methods to measure semantic relatedness. However, these approaches do not always accurately capture conceptual relatedness as measured by humans.

In this thesis we propose and evaluate the use of novel Spreading Activation (SA) techniques for computing semantic relatedness, by modelling the article hyperlink structure of Wikipedia as an associative network structure for knowledge representation. The SA technique is adapted and several problems are addressed for it to function over the Wikipedia hyperlink structure. Inter-concept and inter-document similarity metrics are developed which make use of SA to compute the conceptual similarity between two concepts and between two natural-language documents. We evaluate these approaches over two document similarity datasets and achieve results which compare favourably with the state of the art.

Furthermore, document preprocessing techniques are evaluated in terms of the performance gain these techniques can have on the well-known cosine document similarity metric and the Normalised Compression Distance (NCD) metric. Results indicate that a near two-fold increase in accuracy can be achieved for NCD by applying simple preprocessing techniques. Nonetheless, the cosine similarity metric still significantly outperforms NCD.

Finally, we show that using our Wikipedia-based method to augment the cosine vector space model provides superior results to either in isolation. Combining the two methods leads to an increased correlation of Pearson $\rho = 0.72$ over the Lee (2005) document similarity dataset, which matches the reported result for the state-of-the-art Explicit Semantic Analysis (ESA) technique, while requiring less than 10% of the Wikipedia database as required by ESA.

As a use case for document similarity techniques, a purely content-based news-article recommender system is designed and implemented for a large online media company. This system is used to gather additional human-generated relevance ratings which we use to evaluate the performance of three state-of-the-art document similarity metrics for providing content-based document recommendations.

# Uittreksel

Die Wêreldwye-Web het 'n vlak van inligting-oorbelading tot gevolg gehad soos nog nooit tevore. Rekenaars is baie effektief met die verwerking en groepering van numeriese en binêre data, maar die konsepsuele groepering van natuurlike-taal data is aansienlik moeiliker om te outomatiseer. Tradisioneel berus sulke algoritmes op eenvoudige sleutelwoordherkenningstegnieke of waarskynlikheidsmetodes om semantiese verwantskappe te bereken, maar hierdie benaderings modelleer nie konsepsuele verwantskappe, soos gemeet deur die mens, baie akkuraat nie.

In hierdie tesis stel ons die gebruik van 'n nuwe aktiverings-verspreidingstrategie (AV) voor waarmee inter-konsep verwantskappe bereken kan word, deur die artikel skakelstruktuur van Wikipedia te modelleer as 'n assosiatiewe netwerk. Die AV tegniek word aangepas om te funksioneer oor die Wikipedia skakelstruktuur, en verskeie probleme wat hiermee gepaard gaan word aangespreek. Inter-konsep en inter-dokument verwantskapsmaatstawwe word ontwikkel wat gebruik maak van AV om die konsepsuele verwantskap tussen twee konsepte en twee natuurlike-taal dokumente te bereken. Ons evalueer hierdie benadering oor twee dokument-verwantskap datastelle en die resultate vergelyk goed met die van ander toonaangewende metodes.

Verder word teks-voorverwerkingstegnieke ondersoek in terme van die moontlike verbetering wat dit tot gevolg kan hê op die werksverrigting van die bekende kosinus vektorruimtemaatstaf en die genormaliseerde kompressie-afstandmaatstaf (GKA). Resultate dui daarop dat GKA se akkuraatheid byna verdubbel kan word deur gebruik te maak van eenvoudige voorverwerkingstegnieke, maar dat die kosinus vektorruimtemaatstaf steeds aansienlike beter resultate lewer.

Laastens wys ons dat die Wikipedia-gebasseerde metode gebruik kan word om die vektorruimtemaatstaf aan te vul tot 'n gekombineerde maatstaf wat beter resultate lewer as enige van die twee metodes afsonderlik. Deur die twee metodes te kombineer lei tot 'n verhoogde korrelasie van Pearson $\rho = 0.72$ oor die Lee dokument-verwantskap datastel. Dit is gelyk aan die gerapporteerde resultaat vir Explicit Semantic Analysis (ESA), die huidige beste Wikipedia-gebasseerde tegniek. Ons benadering benodig egter minder as 10% van die Wikipedia databasis wat benodig word vir ESA.

As 'n toetstoepassing vir dokument-verwantskaptegnieke ontwerp en implementeer ons 'n stelsel vir 'n aanlyn media-maatskappy wat nuusartikels aanbeveel vir gebruikers, slegs op grond van die artikels se inhoud. Joernaliste wat die stelsel gebruik ken 'n punt toe aan elke aanbeveling en ons gebruik hierdie data om die akkuraatheid van drie toonaangewende maatstawwe vir dokument-verwantskap te evalueer in die konteks van inhoud-gebasseerde nuus-artikel aanbevelings.

# Acknowledgements

I would like to express my sincere gratitude to the following people and organisations:

- My study leaders, Dr. Gert-Jan van Rooyen and Dr. Herman Engelbrecht for their expert guidance over the last two years,
- MIH for financial support during the course of this study,
- JP Strauss from 24.com for his contribution and help in sorting out the numerous integration difficulties towards implementing the news article recommender system in the Health24.com CMS,
- Prof. Michael D. Lee for kindly providing the human ratings for his dataset and for answering numerous queries about their experimental protocol,
- Dr. Steve Kroon for help with the graph-related work and optimisation,
- All my friends in the Media Lab for making the long hours worth it and the dull moments interesting,
- Anita van der Spuy for helping with some of the tongue-twisters in the Afrikaans abstract, and finally
- My parents and my girlfriend for all their much needed support and understanding.

# Contents

# List of Figures

# List of Tables

# Nomenclature

## Acronyms

| | |
|---|---|
| **AA** | Agglomerative Approach spreading strategy |
| **AA-cos** | cosine agglomerative spreading strategy |
| **AA-wlm** | WLM-based agglomerative spreading strategy |
| **AN** | associative network |
| **API** | Application Programming Interface |
| **BIM** | Binary Independence Model |
| **BOW** | bag-of-words |
| **CB** | content-based |
| **CF** | collaborative-filtering |
| **CMS** | content management system |
| **DB** | database |
| **ED** | pure Energy Distribution weighting scheme |
| **ESA** | Explicit Semantic Analysis |
| **ILF** | Inverse Link Frequency weighting scheme |
| **IR** | Information Retrieval |
| **KB** | knowledge base |
| **LM** | language model |
| **LSA** | Latent Semantic Analysis |
| **MAP** | mean average uninterpolated precision |
| **NCD** | Normalised Compression Distance |
| **NILF** | Normalised Inverse Link Frequency |
| **NID** | Normalised Information Distance |

| | |
|---|---|
| **NGD** | Normalised Google Distance |
| **NLTK** | Natural Language Toolkit |
| **POS** | part-of-speech |
| **RDBMS** | relational database management system |
| **SA** | Spreading Activation |
| **SOAP** | Simple Object Access Protocol |
| **SVD** | Singular Value Decomposition |
| **TAA** | Target Activation Approach spreading strategy |
| **tf-idf** | term-frequency inverse document-frequency |
| **TSA** | Term Space Augmentation |
| **TSS** | Term Space Substitution |
| **UML** | Unified Modelling Language |
| **VSM** | Vector Space Model |
| **WLM** | Wikipedia Link-based Measure |
| **WSDL** | Web Services Description Language |
| **XML** | eXtensible Markup Language |
| **XML-RPC** | XML Remote Procedure Call |

# Notation

| | |
|---|---|
| $\rho$ | correlation coefficient |
| $\lambda$ | mixture weight between cosine VSM and Wikipedia-based methods |
| $a_{j,\text{in}}$ | activation energy input for node $v_j$ |
| $a_{i,\text{out}}$ | activation output of a node $v_i$ connected to a node $v_j$ |
| $A, B$ | objects of discussion |
| $\mathbf{A}_i$ | real-valued activation vector $\{a_{i1}, \ldots, a_{iN}\}$ for node $v_i$ |
| $b$ | Okapi BM25 tuning parameter |
| $c_{t,d}$ | theoretical BIM value of relatedness between query term $t$ and document $d$ |
| $C(\cdot)$ | some arbitrary compression function |
| $\mathcal{C}$ | document-concept vector of the form $\{(v_1, p_1), \ldots, (v_{\lvert\mathcal{C}\rvert}, p_{\lvert\mathcal{C}\rvert})\}$ |
| $d$ | document |
| $\text{df}_t$ | document-frequency of term $t$ (number of documents in $\mathcal{D}$ in which $t$ occurs) |
| $\mathcal{D}$ | collection of indexed documents |
| $E(d_i, d_j)$ | information distance between the text of documents $d_i$ and $d_j$ |
| $G, \lvert G\rvert$ | directed graph, number of nodes in $G$ |
| $\text{idf}_t$ | inverse document-frequency of term $t$, $\text{idf}_t = \log \frac{N}{\text{df}_t}$ |
| $i, j, k$ | indexes for objects in a collection |
| $I(\cdot)$ | the information content of an event, defined as $I(A) = -\log P(A)$ |
| $K_1$ | constant Okapi BM25 tuning parameter |
| $K_{\text{init}}$ | constant initial activation value (we used $K_{\text{init}} = 1.0$) |
| $K_{\text{decay}}$ | constant global network decay spreading parameter $0 \leq K_{\text{decay}} \leq 1$ |
| $K(d)$ | the Kolmogorov-complexity of the text of document $d$ |
| $L_d$ | length (number of words) of document $d$ |
| $L_{\text{ave}}$ | average length of all indexed documents |
| $L_{p,\text{max}}$ | maximum path length allowed to spread to |
| $M_d$ | generative language model of document $d$ |
| $N$ | total items in a set or collection |
| $\mathcal{N}(v_i)$ | set of neighbour nodes for node $v_i$ |
| $O(p)$ | the odds-ratio $\frac{p}{1-p}$ of an event with probability $p$ |
| $p_t$ | probability of term $t$ appearing in a relevant document |
| $P(A)$ | the probability of event $A$ occurring |
| $q$ | a user query in information retrieval |
| $r$ | relevance score |
| $s_{ik}$ | maximum similarity between some concept $v_i \in \mathcal{C}_i$ and all $v_j \in \mathcal{C}_j$ |
| $\text{sim}(A, B)$ | function for computing the real-valued similarity between $A$ and $B$ |
| $t$ | terms in document |
| $\text{tf}_{t,d}$ | term-frequency (count) of term $t$ in document $d$ |
| $T$ | spreading activation threshold value |
| $\mathcal{T}_{d,q}$ | intersection of terms contained in document $d$ and query $q$ |
| $u_t$ | probability of term $t$ appearing in a non-relevant document |
| $v_i \in \mathbf{V}$ | element $v_i$ from vector $\mathbf{V}$ |
| $\mathbf{V}$ | vector of objects |
| $\mathcal{V}$ | set of objects |
| $\lvert\mathbf{V}\rvert$ or $\lvert\mathcal{V}\rvert$ | number of elements in $\mathbf{V}$ or $\mathcal{V}$ |
| $\lvert\lvert\mathbf{V}\rvert\rvert$ | magnitude of vector $\mathbf{V}$ |
| $w_{ij}$ | weight of link connecting nodes $v_i$ and $v_j$ |

# Chapter 1

# Introduction

## 1.1 Motivation

The volume of information available to users on the World Wide Web is growing at an exponential rate [4]. Due to this increasing level of information overload, it is becoming more difficult for users to find documents that are relevant to their information needs. One main reason for this is that the semantics in documents are not adequately recognised by existing keyword-based systems [5].

Information retrieval has come a long way from the boolean models of relevance, where a document either contains a user's query terms, and is thus considered relevant – to vector space models and probabilistic models of relevance, where the distribution of keywords in the corpus of documents are used to estimate how important these keywords are for computing relevance [6].

Google [7] and Bing [8] are some of the most popular Web search engines in use today [9]. Although improvements such as Google's PageRank algorithm [10] can lead to improved relevance of results, these systems still, however, fundamentally rely on the bag-of-words keyword-matching model, in which word-order is ignored and semantics play a secondary role [11].

Such keyword-matching systems suffer from several limitations, the most notable being an inability to accurately model the ambiguities in natural language, such as synonymy (different words having the same meaning) and polysemy (one word having multiple different meanings). Choosing which sense a polysemous word refers to is largely governed by the context in which a word appears, which is ignored by these simpler models [11].

In recent years, much research attention has therefore been given to semantic techniques of information retrieval, whereby the system attempts to rank and return documents which relate to the semantic meaning of the user's search query. Such systems allow for sophisticated semantic search, however, they require the use of a more difficult-to-understand query-syntax [12]. Furthermore, these methods of semantic search rely on specifically encoded *ontologies* to describe the particular domain knowledge in which they operate, and the specific interrelations of concepts within that domain, using formal languages such as OWL [13] or RDF [14], which are costly to produce.

In this thesis, we focus on the problem of computationally estimating similarity or relatedness between two natural-language documents. We want to evaluate to what extent existing algorithmic approaches for computing similarity agree with how humans perceive

and rate similarity. We also develop and test several new measures for computing similarity between single concepts (inter-concept similarity), as well as between documents (inter-document similarity).

A natural-language processing system is developed with the purpose of computing the semantic relatedness or similarity between two natural-language documents. This is useful both in a document-recommender context, where recommendations are made by finding and relating other similar documents to a user, as well as in a retrieval context, where a user enters a search phrase or query and the system presents documents from its index that are estimated to be most related to that query. This system makes use of a novel technique (developed in this thesis) for computing semantic similarity, by spreading activation over the hyperlink structure of Wikipedia, the largest free online encyclopaedia [15].

It will be demonstrated that the proposed technique can achieve a Pearson linear product-moment correlation[1] of $\rho = 0.68$ with human ratings over the well-known Lee document-similarity dataset [16], which compares favourably with the state-of-the-art Explicit Semantic Analysis (ESA) technique [1], which achieves a correlation of $\rho = 0.72$ over the same dataset. However, our approach only makes use of the anchor text and hyperlink structure of Wikipedia (1GB as of 2009), whereas the ESA approach requires an indexed copy of the entire Wikipedia database (21GB as of 2009). In addition, it will also be shown that combining our proposed technique with the well-known cosine similarity technique [17] used in the Vector Space Model, leads to an improved correlation of 0.72 on the same dataset.

In the past, a major barrier to using such knowledge-based approaches for computing document similarity was the time, skill and effort required to create a knowledge base of sufficient breadth and quality to capture all concepts and their interrelationships as they might appear in the documents under comparison. Crestani [18] notes:

> *"... the problem of building a network which effectively represents the useful relations (in terms of the IR's aims) has always been the critical point of many of the attempts to use Spreading Activation in IR. These networks are very difficult to build, to maintain, and keep up to date. Their construction requires in-depth application domain knowledge that only experts in the application domain can provide."*

Since our techniques make use of the freely available Wikipedia as a knowledge base, however, we do not incur the additional cost of creating and maintaining such a specialised domain ontology, a task normally performed by a lexicographer or professional knowledge engineer. Even so, additional strategies are required for translating the hyperlink structure of Wikipedia into a suitable associative network format, and for this new techniques are proposed and tested.

Document similarity metrics has applications in information retrieval, conceptual document clustering, and document recommendation systems. As a proof-of-concept of the application of automatic document similarity measurement for use in recommending related documents, a selection of these algorithms was integrated into the content management system of a media company for use by professional journalists. This implementation included a rating feedback system which provided a platform for collecting additional human-generated data which was used in evaluating the utility of such a system.

[1] Also referred to as 'Pearson's $r$'.

## 1.2 Problem statement

This study is concerned with the problem of computationally estimating the relatedness or similarity between two natural-language documents. This problem is significant since it is used in information retrieval to compute the relatedness of a document to a user's query, and also in document recommendation, where a user is presented with additional documents which are related to or similar to a given document.

## 1.3 Research objectives

The objectives of this study are to:

- Develop a conceptual document similarity metric which can be used to estimate conceptual similarity between natural-language documents.
- Evaluate the major available document similarity metrics by correlation with human ratings obtained from conducting controlled psycholinguistic experiments.
- Evaluate and quantify the effects of document preprocessing techniques on the performance of 'simpler' similarity metrics, such as cosine similarity and Normalised Compression Distance (NCD) (introduced later in this chapter).
- Research ways in which man-made lexical and ontological resources can be used to augment these simpler metrics for increased performance.
- Develop a content-based news-article recommender system for Health24.com. Use this system to collect data from their journalists and to evaluate the performance of different document similarity algorithms in providing such content-based recommendations.

## 1.4 Background information

In this section we provide a quick overview of the major approaches that exist for computing document similarity. This is only meant as a general introduction to these methods, as a much more in-depth and thorough survey of the literature will follow in Chapter 2.

Computational approaches for quantifying semantic similarity between documents fall into three categories (our classification):

I Purely algorithmic approaches which require no background information;

II Approaches which require unstructured background information, such as a collection of text documents; and

III Approaches which require structured knowledge bases and formal domain ontologies.

The problem we are focusing on in this thesis can be summarised as computing the semantic similarity between natural-language documents in order to cluster texts conceptually and in order to recommend related articles. This problem touches on several larger problems in the field of artificial intelligence, such as knowledge representation and

automated reasoning. However, for our purposes we do not require the same level of 'understanding'.

Given two strings, the simplest approach to measure similarity between them is at the lexical (letters of words) level. One can obtain an exact match (two strings are equal), a phrasal match (one string is a substring of the other, e.g. 'car' and 'sports **car** magazine'), or one string can contain a subset of the terms in the other string, e.g. 'Seattle mariners tickets' and 'tickets mariners' [19].

This can be measured using for instance the Levenshtein string edit distance [20], or by breaking up text into sets of $n$-grams (with $n$ the number of letters per group), and measuring the amount of overlap between these sets of $n$-grams [21]. For instance 'verify' can be broken up into the following trigrams: {'ver', 'eri', 'rif', 'ify'}.

However, such a simplistic approach has two main problems: It does not take context into account, and it also contains no background information. Context is important to distinguish between, for instance, ambiguous polysemous words, such as 'river **bank**' and '**Bank** of America'. Background knowledge is important for the reasons mentioned above, and for instance to know that 'sheep' and 'goat' are more related (or more similar) than for instance 'car' and 'cat', even though the latter are lexically more similar (differing only in their final letters).

In the following paragraphs, we present an overview of the major approaches that exist for computing semantic similarity between two strings or texts. We start with purely algorithmic approaches which measure similarity as the amount of information overlap between two strings. We then introduce the major retrieval models used in Information Retrieval (IR), such as the vector space model or the probabilistic model, as examples of approaches utilising collections of unstructured information to compute semantic relatedness. Finally, we introduce recent works which aim to extract semantic relationships from collaboratively-edited resources such as Wikipedia, as a starting point to the work contained in this thesis. Each of these methods will be evaluated later on in this thesis, by comparing them to similarity ratings collected from human participants in Web-based experiments.

## Normalised Compression Distance (NCD)

The NCD defines the distance between two strings as the normalised amount of unique information between the strings, by borrowing concepts from information theory [22]. This is based on the non-computable Kolmogorov-complexity and views two strings as streams of information. Kolmogorov-complexity is a theoretical concept, and can be seen as the ultimate compressed version of a piece of data from which the original data can be recovered. Since this function is non-computable, it is approximated by using standard compression techniques, such as gzip [23] or bzip2 [24].

## Similarity in information retrieval systems

Information retrieval (IR) systems, such as Web search engines and database query systems, use mathematical retrieval models to define and quantify *relatedness* between a user's submitted *query* and the documents or Web-pages the system has in its index. In this section we present an overview of the main approaches to measuring similarity in use by IR systems, as a background to the large body of available approaches.

In IR, a user submits a query $q$ which the system matches against its list of $N$ indexed documents $\mathcal{D} = \{d_1, ..., d_N\}$, to return some list of documents $\{d_1, ..., d_j\}$ in decreasing order of relevance to the query, with relevance rated by die specific retrieval model in use.

Zipf [25] noticed that the distribution of most words in text corpora follow a power distribution (a so-called 'long tail') where the frequency of a word is inversely proportional to its rank in the frequency table (i.e. the most frequent word occurs two times more frequently than the second most frequent word, etc., see Figure 1.1). This hypothesis has empirically been shown to be accurate, and is known as *Zipf's Law*.

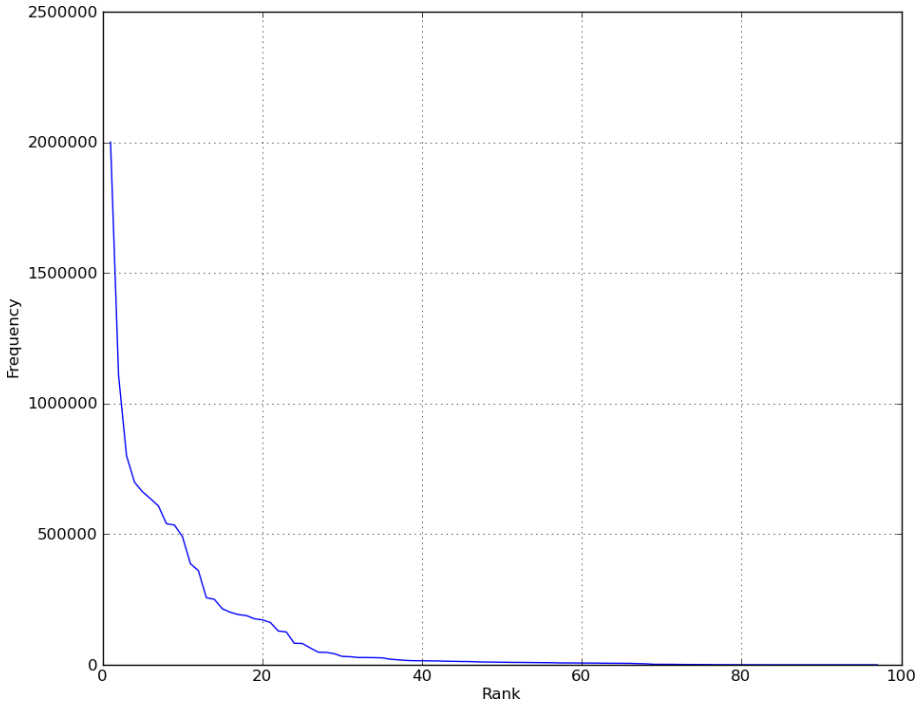



**Figure 1.1:** Word rank versus word frequency over the Gutenberg corpus, illustrating the concept known as Zipf's law, where the frequency of a word is roughly inversely proportional to its rank.

This inspired the notion of using the distributional statistics of words in a corpus to rank the importance or *discriminatory power* of individual terms, and to use some ranking function to determine the rank order of relevant documents to a given query.

Different *retrieval models*, each having their own ranking functions, exist for modelling relevance. We will focus on the three main models of relevance found in the literature, namely the vector space model, the probabilistic model and query-likelihood statistical language models.

### Vector space model (VSM)

In the VSM, documents are represented as vectors containing the terms in the document with their respective weights of importance in the document as defined by some weighting function. It is based on the *bag-of-words (BOA)* assumption, which simply means that documents are viewed as a bag of words in which word order is ignored. Therefore the documents *'John loves cycling'*, *'loves John cycling'*, *'cycling John loves'*, etc. are viewed as identical. It is also based on the assumption that the more often a query term appears

in a document, the more relevant that document is to the given query. This is referred to as the *term frequency* of a term.

The problem with this approach, however, is that certain words naturally appear more frequently than others. Consider for instance the prevalence of the article 'the', as opposed to, for instance, the noun 'bifurcation'. To address this problem, a term's weight is also reduced by its *inverse document-frequency* – the ratio of documents in the entire corpus that contains a specific term. The more often some term appears in all documents across the corpus, the less significance it carries in this model.

In order to compute the similarity between query and document, their vectors are compared using standard correlation metrics, such as the cosine similarity which we will introduce in Equation 2.4.3 later on in this work.

### Probabilistic retrieval and Okapi BM25

In an IR system, users start with an *information need*, which they translate into a *query* to express that need. The IR system, on the other hand, consists of a collection of documents. Each document is translated into some *document representation* which, as an approximation, differs from the documents in certain ways – at the very least as a result of how the text was tokenised, non-important words removed, term-positions in the text discarded, etc. Both these processes of translation introduce a degree of uncertainty into the process of judging relevance of a document to a user's original information need. Probability theory provides a principled foundation for such reasoning under uncertainty.

In a probabilistic model, a document's relevance to a user's query is estimated as a probability and documents are returned in decreasing order of their estimated probability of relevance. This obvious ordering is captured by the Probability Ranking Principle, which forms the fundamental basis of the probabilistic model, and can be stated as:

> *'If a reference system's response*[2] *to each request is a ranking of the documents in the collection in order of decreasing probability of relevance to the user who submitted the request, where the probabilities are estimated as accurately as possible on the basis of whatever data have been made available to the system for this purpose, the overall effectiveness of the system to its user will be the best that is obtainable on the basis of those data'* [26]

However, since it is very difficult in practice to obtain the necessary data to calculate these probabilities accurately, several simplifying assumptions are made in the Binary Independence Model (BIM) [27] which make calculating these estimations easier. For short texts, the BIM works reasonably well, however, the assumptions on which it is based do not generally hold as well for longer pieces of text and for pieces of irregular length. The Okapi BM25 heuristic weighting scheme [28] addresses these problems and builds on the BIM by incorporating these crucial statistics into the basic BIM. The result is a combination of the probabilistic model and the vector space model – a retrieval model which has been shown to perform very well in several works in the literature [29].

[2] A 'reference system' would nowadays be called a 'search engine'.

### Statistical language models

The use of language models (LMs) in the IR problem was largely introduced by Ponte and Croft in their pivotal paper *'A Language Modelling Approach to Information Retrieval'* [30] and is the most recent of the three models introduced so far. Query-likelihood statistical language models construct a generative language model which attempts to describe the language of a given document. It uses this to answer the simple question: 'What is the likelihood that the current document $d$ in the collection of indexed documents $\mathcal{D}$ can generate the given query $q$?'. This probability is estimated for each document in the collection with regard to the given query, and documents are ranked in their order of decreasing probabilities.

Models can vary in their complexity based on the number of previous terms that are used to calculate the probability of observing the present term. If only the present term is used, the model is referred to as a *unigram language model*, if the present and previous term is used, it is called a *bigram language model*, etc.

The LM approach has been welcomed by the IR community, since it provides a conceptually simple way of viewing the problem of relevance ranking. This approach has also been shown to achieve very competitive scores in text retrieval competitions.

## Extracting relationships from Wikipedia

As was mentioned in the introduction to this section, computational approaches for calculating similarity between documents or fragments of text can be divided into three categories, and the approaches introduced so far belong to the following:

I purely algorithmic approaches, requiring no background data – for instance NCD;

II approaches requiring unstructured background information, such as text documents – for instance IR retrieval models.

Although these approaches can successfully extract 'meaningful' relationships between terms, phrases, etc., from large quantities of text, these relationships are purely statistical – i.e. they lack any common-sense roots. To solve this problem, methods from category III are required, namely approaches requiring structured knowledge bases, or ontologies, which define the relationships between concepts in a given domain. The drawback is that the creation of formal ontologies must be done by professional knowledge engineers, which therefore make them very costly to create.

Recently, several approaches have been suggested in the literature that make use of Wikipedia [15], the largest online collaboratively-edited and free encyclopaedia that is available. These approaches view each of the encyclopaedia's articles as representing one specific concept, for instance `Human`, `House`, or `Language`. However, the approaches differ in how they compute relatedness, or similarity, between individual concepts.

**Explicit Semantic Analysis (ESA)** [1] views Wikipedia's articles as a *concept space*. ESA indexes the full body-text of Wikipedia articles using the vector space information retrieval model discussed above. Given a piece of text, an IR lookup is performed and the set of Wikipedia article titles (i.e. document titles in the VSM) that best represent that piece of text is retrieved – what they call *document concept vectors*. To compute similarity, a cosine similarity operation between the two document concept vectors of two pieces of text is performed.

**Wikipedia Link-based Measure (WLM)** [2] views the different anchor text descriptions in hyperlinks[3] pointing to pages as *synonyms* for the pages (or concepts) they link to. Relatedness is measured between these concepts by using the sets of all other pages that link to that specific page as a description vector for that specific concept. WLM therefore describes a concept by the set of all other concepts that link directly to that concept. Finally, similarity is computed between these sets, and thus between the concepts they represent, using a measure to compute information overlap, derived from the Normalised Compression Distance metric introduced above (we will discuss this more thoroughly in the next chapter).

### Associative networks and spreading activation

An **associative network (AN)** is a data structure for knowledge representation which is a more general form of a *semantic network*, as introduced by Quillian [31]. In this network, information items or concepts are represented by nodes, and links between nodes represent the level of association between nodes, usually as some real-valued weight.

**Spreading activation (SA)** is a graph search technique for searching such associative networks, based on the supposed mechanisms of human memory operations [32]. It starts by supplying activation energy to one or more nodes in an AN. This energy is then iteratively propagated to associated nodes until certain termination conditions are met. The real-valued level of activations of related nodes then indicate to what extend they are related to the initial nodes.

## 1.5 Thesis statements

The work in this thesis is based on two primary hypotheses which were used to steer the direction of research, and which we set out to either prove or disprove, namely:

1. The Wikipedia hyperlink structure can be modelled as an associative network (AN), and can therefore be processed by spreading activation between its articles (which are viewed as concept nodes) via the hyperlinks connecting them (which are viewed as links of association in the associative network model), to accurately estimate the relatedness or similarity between these concepts.

2. A statistical, keyword-based measure of semantic relatedness in combination with a knowledge-based approach captures document similarity ratings better than either in isolation.

The first thesis statement is related to the fact that users tend to create hyperlinks from one page to another when the two pages contain concepts which are related to one another. Since links relate one article to its neighbours and by extension relate these articles to their neighbours, we hypothesise that we can extract and process this hyperlink structure as an associative network of concepts and links relating them to each other. This hypothesis led us to adapt spreading activation, a technique for processing associative networks, and to apply it to the hyperlink structure of Wikipedia in order to measure relatedness between

[3] I.e. the text between the `<a href...>` and `</a>` HTML tags.

concepts and documents. We test this hypothesis by conducting several tests to evaluate its performance.

The second thesis statement is related to the fact that keyword-matching approaches, such as the vector space model (VSM) introduced earlier, measures relatedness between two documents based on the keywords contained in both documents and some weighting function. This is based on the observation that when two documents share words, they are generally related. However, words can be ambiguous and there are many different ways of stating the same idea without using the same words, and this is difficult to account for in the VSM.

On the other hand, knowledge-based approaches make use of data structures such as associative networks, which relate concepts to one another according to how humans rate these concepts. Such an approach can generally capture much more complicated relationships between concepts, even when no keywords are shared – for instance the relatedness between *'kids' books'* and *'children's literary works'*. However, a knowledge-based approach can only capture relationships between concepts actually defined in its lexicon. If it is not contained in the knowledge base, it knows nothing about it, even if it is the same word.

We therefore hypothesise that a combination of the two approaches will lead to improved accuracy in rating document similarity.

## 1.6 Contributions

The following contributions are made:

- The spreading activation model is adapted and extended for processing the Wikipedia hyperlink graph structure. Strategies are developed for estimating the connection-weights between two articles, and also for translating activations into a measure of relatedness.
- An inter-concept similarity metric is developed which uses spreading activation over the hyperlink structure of Wikipedia and achieves results of Pearson's $\rho = 0.70$ over the WordSimilarity-353 [33] dataset which compares favourably with state-of-the-art methods, such as WLM [2] ($\rho = 0.69$), and ESA [1] ($\rho = 0.75$).
- Two inter-document similarity metrics are developed which combine Wikipedia topics extracted from documents into inter-document similarity scores using spreading activation.
- A combined document similarity algorithm is developed which relies on the cosine Vector Space Model (VSM) with term-frequency inverse document-frequency (tf-idf) and simple document preprocessing, and the Wikipedia-based metric mentioned above. It is shown that using the combined model improves performance, and achieves a correlation of $\rho = 0.72$ over the standard Lee document-similarity dataset (Lee [16] describes this dataset; also discussed in detail in Chapter 7). This compares favourably with ESA ($\rho = 0.72$ on the same dataset), while requiring less than 10% of the Wikipedia database required for ESA (1GB required, versus 21GB for ESA).
- A stand-alone, Web-based news-article recommender system is developed for Health24. This is used as a platform for evaluating the performance of three state-of-the-art

information retrieval algorithms in a news-article recommender setting, and it is concluded that pure content-based recommendations, based simply on document similarity, can achieve high performance ratings as measured by the mean average uninterpolated precision (MAP) scores as rated by professional journalists using the service.

- Cosine similarity with tf-idf term weighting, Okapi BM25 and statistical language models are evaluated in the context of providing accurate news-article recommendations and it is found that all three methods score highly, with Okapi achieving the highest ratings as measured by MAP.

- The potential for increasing the accuracy of simple vector space document similarity metrics and the NCD metric, by applying document preprocessing procedures on the input text is evaluated over two datasets. It is concluded that cosine similarity improves 9% and 28% respectively over the baseline (using raw text), as measured by Pearson correlations with human-assigned ratings. NCD correlation improves 54% and 80% respectively by applying simple preprocessing steps, and converting terms into bigrams, but is still significantly outperformed by the cosine VSM.

## 1.7 Thesis overview

**Chapter 2** provides an in-depth study of the necessary background information and literature related to this work.

**Chapter 3** presents a thorough overview of the spreading activation process and discusses how to model the Wikipedia hyperlink structure as an associative network. It explains how the model of spreading activation is extended and adapted specifically for the Wikipedia graph, and which problems had to be addressed in order to do so. Strategies are required for interpreting the results of the spreading activation process, for which we introduce new approaches.

**Chapter 4** discusses some of the available preprocessing techniques that can be applied to text, prior to measuring relatedness using simple vector space methods or the NCD. The problem of computing inter-document similarity is further explored and discussed within the Spreading Activation (SA) framework developed in Chapter 3. Two new document-similarity metrics, making use of Wikipedia and SA, are presented. Finally, we propose a combined method of our Wikipedia-based method and the cosine VSM in order to evaluate our hypothesis that such a combined method would lead to increased accuracy in measuring document similarity.

**Chapter 5** documents a study which was conducted in parallel with the main study, regarding a specific use case for document similarity metrics, namely providing news-article recommendations for Health24.com, an online South African media company. This provided a platform for testing several state-of-the-art document similarity algorithms and their performance in providing purely content-based recommendations, by collecting ratings from the professional journalists who used the system.

**Chapter 6** discusses the design and implementation of Web-based experiments to obtain a suitable 'gold standard' sub-corpus from the full Health24 document-corpus. This sub-corpus is used as benchmark for comparing the performance of different document similarity measures. The construction of such a sub-corpus is discussed and motivated.

**Chapter 7** contains all the experiments that were conducted for this thesis. We first test the accuracy of the proposed model for spreading activation (Chapter 3). Next we evaluate what effect different techniques of preprocessing document text has on the performance of NCD and the cosine VSM (Chapter 4). We then test the accuracy of all document similarity metrics discussed in this thesis by making use of a standard document similarity dataset and our own Health24 dataset and human ratings (Chapter 6). Finally we analyse the ratings obtained from journalists who used the document recommender (Chapter 5), in order to see which metrics provide the most accurate news-article recommendations.

**Chapter 8** presents a conclusion to the work conducted during the course of this thesis by briefly reiterating and discussing the main results that were obtained. We present concluding thoughts for our initial thesis statements (see Section 1.5). Finally, recommendations for future research are made.

# Chapter 2

# Literature Review

In this chapter, related works in the literature are presented to provide the necessary background for the work in the rest of this thesis.

## 2.1 Introduction

The literature on available approaches for computing document similarity can be divided into three main categories (our classification), namely:

I purely algorithmic approaches requiring no background information;

II approaches requiring unstructured background information, such as a collection of text documents; and

III approaches requiring structured knowledge bases or formal domain ontologies.

From the first category, we present Lin's Similarity Theorem [21] which provides a conceptual framework for quantifying 'similarity'. We also present the Normalised Compression Distance (NCD) [22] – a feature-free measure of relatedness between streams of information, such as text documents, based on information theory and compression algorithms.

From the second category, we present three of the main retrieval models used by Information Retrieval (IR) systems, such as Web search engines. These approaches use statistical approaches to extract relationships from vast quantities of unstructured text data to determine for instance which terms generally occur together. This information is used within the framework of the specific retrieval model to estimate which documents in the index are most related to a user's query. In this section we introduce the Vector Space Model (VSM) [17], the probabilistic retrieval model [34] (and specifically the Okapi BM25 algorithm [28]), and finally the query-likelihood statistical language modelling approach [30]. We will evaluate how accurately these models correlate with human ratings later in this thesis.

From the final category, we present recent approaches such as the Wikipedia Link-based Measure (WLM) [2] and Explicit Semantic Analysis (ESA) [1]. These approaches utilise the online encyclopaedia Wikipedia [15], to extract indications of semantic or conceptual relatedness between words and documents. The work in the rest of this thesis builds on and extends the ideas presented in these works.

## 2.2 Notation and terminology

An attempt was made to make use of a consistent notation throughout this document to provide as little confusion as possible. In cases where this was more difficult, we tried to make the meaning of each symbol as clear as possible to the reader from the context, or immediately following its use.

In general, $A$ and $B$ are used to refer to two general objects of discussion, the meaning of which will be made clear in the text. $d$ is used to refer to documents, and $t$ is used to refer to the terms in documents. $w$ represents a weight associated with some object. Bold notation, such as $\mathbf{A}$ or $\mathbf{V}$, is used to represent a vector of objects. Cursive notation, such as $\mathcal{V}$ or $\mathcal{C}$, is used to denote a set of objects.

The variable $N$ refers to the total number of items in a collection, for instance words in a document, or documents in an index. We often use indexes as subscripts to differentiate specific objects, for example $v_i$ is node $i$. $i$ and $j$ are generally indexes for two objects from different sets or collections which will be made clear in the text. $k$ is used as index for an arbitrary different object compared to $i$ and $j$.

The function $\text{sim}(A, B)$ is used to denote a similarity function, which computes the similarity between objects $A$ and $B$ and returns a real-valued result. The probability of an event $A$ is denoted using the standard notation $P(A)$.

## 2.3 Purely algorithmic approaches

In this section we introduce two purely algorithmic approaches to computing document similarity pertaining to category I as identified above. These approaches use only the text contained in each document to compute similarity.

### 2.3.1 Similarity in information theory terms

In the following discussion, we look at work in the literature that deals with similarity on a very general psychological level and work towards more specific computational models as they are applied to the problem of computing semantic similarity between texts. We generally use 'object' to refer to some concept (such as a 'cow' or an 'apple') which can be compared to some other concept. Naturally, concepts can be either tangible and fully perceptible, or intangible and abstract. We do not consider or debate the acquisition of such a shared collection of concepts, but assume that these concepts can be fully described by their features or attributes.

Tversky [35] developed three models for how humans evaluate similarity between objects as a function of the presence or absence of features between the objects. As noted above, one can characterise objects by their observable features or attributes, e.g. some features of `Human` might be 'breathing', 'alive', 'biped', 'mammal', 'intelligent', etc. Lee, Pincombe and Welsh [16] evaluated these models and obtained their best results with what is called the 'ratio model', or measuring similarity between concepts as a ratio of the common features they share to the total common and distinctive features that describe objects.

Lin [21] draws from Tversky's work and systematically develops an information-theoretic and domain-agnostic model of similarity between some object $A$ and object $B$, based on three main intuitions (taken from [21]):

1. **Intuition 1**. The similarity between $A$ and $B$ is related to their commonality. The more commonality they share, the more similar they are.

2. **Intuition 2**. The similarity between $A$ and $B$ is related to the differences between them. The more differences they have, the less similar they are.

3. **Intuition 3**. The maximum similarity between $A$ and $B$ is reached when $A$ and $B$ are identical, no matter how much commonality they share.

Lin further quantifies his three main intuitions by laying out six additional assumptions, stipulating among others that the amount of information contained in a proposition, or $I(\cdot)$, is measured in information-theoretic terms as the negative logarithm of the probability of the proposition being true.

For instance, consider two objects, $A$ (an orange) and $B$ (an apple), which are represented by sets of features $\mathcal{V}_A$ and $\mathcal{V}_B$ which attempt to describe them. Features may include for instance {'fruit', 'orange-colour', 'sweet-taste', 'round-shape', etc.}. We refer to such a set of features as the 'description' of the object. Each of these features or attributes, such as 'fruit', has an associated probability of being true (as best we can observe) for that specific object. The information content, $I(\cdot)$, of some $i$'th feature, $v_i$ is defined in information theory terminology as the negative logarithm of the proposition it contains, $I(v_i) = -\log P(v_i)$, where $P(v_i)$ denotes the probability of feature $v_i$ being true.

The result of these considerations is the Similarity Theorem [21], which measures the similarity between two objects as the ratio of the information contained in their common features ($\mathcal{V}_A \cap \mathcal{V}_B$), to the information contained in the total set of all features between the two objects ($\mathcal{V}_A \cup \mathcal{V}_B$):

$$\text{sim}(\mathcal{V}_A, \mathcal{V}_B) = \frac{I(\mathcal{V}_A \cap \mathcal{V}_B)}{I(\mathcal{V}_A \cup \mathcal{V}_B)}. \tag{2.3.1}$$

### 2.3.2 Normalised Compression Distance (NCD)

The Normalised Compression Distance is a similarity metric based purely on information theory [22]. It is based on the non-computable Kolmogorov complexity (also called descriptive complexity or algorithmic entropy [36]), which denotes a measure of the computational resources needed to completely describe an object. More formally, the *Kolmogorov complexity* of some string is the length in bits of the shortest computer program in a fixed reference universal computing system that produces that string as its output.

For some document $d$ (which is simply a text string), the Kolmogorov complexity $K(d)$ can be likened to the ultimate compressed version from which $d$ can be recovered by a general decompression algorithm.

Using for instance *gzip* [23] as compression algorithm $C(\cdot)$ on $d$ produces $C_{\text{gzip}}(d)$. Using a compressor with a higher compression rate, such as *bzip2* [24] produces $C_{\text{bzip2}}(d)$. If we use $|\cdot|$ to represent the size of the compressor's output, it usually holds that $|C_{\text{bzip2}}(d)| < |C_{\text{gzip}}(d)|$. Using some fictitious algorithm *fict*, with an even higher compression rate, produces $C_{\text{fict}}(d)$ with $|C_{\text{fict}}(d)| < |C_{\text{bzip2}}(d)|$, etc. However, the Kolmogorov complexity $K(d)$ gives a lower bound for the compressed length of $d$; for every known or unknown compression algorithm, $K(d)$ is less than or equal to the compressed version of $d$.

Given two text documents $d_i$ and $d_j$, the *information distance* $E(d_i, d_j)$ is defined as the length of the shortest binary program in a reference universal computing system such that the program computes output $d_j$ from input $d_i$, and also output $d_i$ from input $d_j$ [37]. This is given as

$$E(d_i, d_j) = K(d_i + d_j) - \min\{K(d_i), K(d_j)\} \tag{2.3.2}$$

where $d_i + d_j$ denotes the concatenation of the text of documents $d_i$ and $d_j$. This measure turns out to be a distance metric[1] which distorts *similarity*, since long documents with a short distance between them are much more similar than short documents with the same distance between them, which this distance metric does not consider. Hence, Cilibrasi and Vitanyi [22] normalise by the maximum information contained in either of the two items, which produces the Normalised Information Distance,

$$\text{NID}(d_i, d_j) = \frac{K(d_i + d_j) - \min\{K(d_i), K(d_j)\}}{\max\{K(d_i), K(d_j)\}} \tag{2.3.3}$$

However, since the Kolmogorov complexity is non-computable, they approximate it with an arbitrary computable compression function which serves as an upper bound to the Kolmogorov complexity of that string. The **Normalised Compression Distance** is thus defined as

$$\text{NCD}(d_i, d_j) = \frac{C(d_i + d_j) - \min\{C(d_i), C(d_j)\}}{\max\{C(d_i), C(d_j)\}} \tag{2.3.4}$$

with $C(\cdot)$ defined as some arbitrary compression function.

NCD is *feature-free* since it does not analyse items for particular features. Instead it tries to analyse all features simultaneously and compute the similarity between two items based on the dominant features identified.

Since this is a normalised distance metric, a distance of 0 indicates two identical strings and 1.0 indicates two completely dissimilar strings. Therefore, subtracting the NCD from 1.0 produces a similarity score.

## 2.4 Approaches which require unstructured background data

In this section we introduce approaches from category II as identified in the beginning of this chapter, which require a collection of unstructured documents for background data. We discuss the three main models in use in Information Retrieval (IR) systems for computing the relevance of a query $q$ that a user submits to the system, to an indexed collection of documents $\mathcal{D} = \{d_1, ..., d_n\}$.

### 2.4.1 Vector space model: Cosine tf-idf & LSA

Given a query $q$ and some document $d$ from the collection of indexed documents $\mathcal{D}$, the problem in this indexing paradigm is to assign to each term occurring in the query a score

[1] $E(d_i, d_i) = 0$, $E(d_i, d_j) > 0$ for $d_i \neq d_j$, $E(d_i, d_j) = E(d_j, d_i)$ (symmetrical), and $E(d_i, d_j) \leq E(d_i, d_k) + E(d_k, d_j)$ for all $d_i, d_j, d_k$ (Triangle Inequality).

which reflects its relevance to $d$. For each document, the query term scores are summed into a relevance score $r$. It is then assumed that the most relevant document would have the highest $r$. The problem, then, is to find a suitable *term weighting scheme* which can compute the weight for a term $t$ given $d$ and $\mathcal{D}$.

The first assumption in this model is that the more frequently a query term appears in a document, the more relevant that document is to the query. This is modelled by a term $t$'s *term frequency*, or the count of $t$ in $d$, given as $\text{tf}_{t,d}$.

In this model, any document can be reduced to a vector which contains the terms of the document and their respective weights as defined by the weighting function being used, such as $\text{tf}_{t,d}$. Furthermore, in this model the exact ordering of terms is ignored and we are only left with information on the number of occurrences of each term. Thus the document *'James is taller than Peter'* is identical to the document *'Peter is taller than James'*. This approach is referred to as the bag-of-words model (BOW).

However, using only the raw term frequencies suffers from a critical problem: All terms are considered equally important when it comes to assessing relevance of a query to a document. This assumption generally does not hold, since words which only occur in a small subset of $\mathcal{D}$ can much better discriminate the respective documents in which they occur, than words which occur in every document in $\mathcal{D}$ [17, 38].

To address this problem, we weight a term $t$'s term frequency count $\text{tf}_{t,d}$ by the number of documents in which it occurs in the entire collection $\mathcal{D}$, called the *document frequency* $\text{df}_t$. In a collection of $N$ documents, we define a measure called the *inverse document frequency* of $t$ as

$$\text{idf}_t = \log \frac{N}{\text{df}_t} \tag{2.4.1}$$

One of the most widely-used term weighting heuristics is called the term-frequency inverse document-frequency (tf-idf) introduced by Salton and McGill in 1983 [17]. Tf-idf combines these two measures into a single term weighting scheme, i.e.

$$\text{tf-idf}_{t,d} = \text{tf}_{t,d} \cdot \text{idf}_t. \tag{2.4.2}$$

The intuition is that the more a term occurs within a document (term frequency), the more weight it should receive because the more it is deemed to 'characterise' that document. However, the more a term appears in the entire corpus (document frequency), the less weight it should carry since its ability to discriminate becomes weaker. For instance if all documents contain the term 'the', it does not discriminate any document very well; however, if only 0,001% of documents contain the word 'parliament', its discriminating power for those documents is much higher.

In the VSM, documents and queries are represented as $k$-dimensional vectors (with $k$ the total number of unique terms in $\mathcal{D}$), with each element in the vector denoting the tf-idf weighting associated with that term. Queries and documents are thus represented as vectors in the same vector space.

These vectors are normally interpreted geometrically, with relatedness between query and document viewed as measuring the angle between their respective vector representations. Many such similarity measures exist, the most prominent and successful being the **cosine correlation**. If we represent the VSM vector representation of query $q$ and document $d$ as $\mathbf{V}_q$ and $\mathbf{V}_d$ respectively, with $v_{xi}$ element $i$ of vector $\mathbf{V}_x$, then the relatedness of $q$ to $d$ can be computed as the cosine correlation between their vectors, defined as

$$\text{sim}_{\cos}(\mathbf{V}_q, \mathbf{V}_d) = \frac{\sum_{i=1}^{k} v_{qi} v_{di}}{\sqrt{\sum_{i=1}^{k} v_{qi}^2 \sum_{i=1}^{k} v_{di}^2}}. \tag{2.4.3}$$

In other words, the cosine similarity computes the normalised dot product (also called the inner product) between $\mathbf{V}_q$ and $\mathbf{V}_d$. It measures the cosine of the angle between the two vectors being compared. The cosine of the angle between two identical vectors will be 1 (the angle is 0), and 0 between completely dissimilar vectors.

Given a query $q$ and a set of documents $\mathcal{D}$, the cosine correlations between their vector representations can be used to produce a list of documents and their ratings of relevance to the query as output. By arranging this list of documents from highest to lowest relevance rating, this list of documents will be ranked according to each document's estimated relevance to the query.

However, the VSM makes two simplifying assumptions which put an upper bound on the accuracy that can be achieved using it:

1. semantics and word order are ignored and text is treated as a bag-of-words (BOW); and
2. the potential semantic relations between words such as synonyms, homonyms, etc. are discarded. In other words, cosine similarity will rank two very similar phrases such as *'kids' books'* and *'children's literary works'* as completely dissimilar since they share no words.

**Latent Semantic Analysis (LSA)** [39] partly addresses assumption (2). LSA uses a dimensionality reduction procedure such as Singular Value Decomposition (SVD) in order to reduce the sparse term-document matrix, which can be high-dimensional, to a lower-order matrix approximating the original. The reasons for performing this step of dimensionality reduction are firstly to make computational processes more tractable due to the term-document's reduced rank. However, most importantly, it is done to reduce the 'noise' in the text and to find hidden or 'latent' connections between words *related* to the given documents, instead of words only *contained in* each document.

This is in a sense the mathematical analogue of recognising synonyms and automatic query expansion in order to find documents that are related, but do not necessarily contain the exact same keywords (Section 4.2 briefly mentions query expansion).

LSA produces state-of-the-art similarity ratings and derivatives of the basic technique is widely in use [40]. However, latent concepts are very hard to understand for humans. In other words, it is not always apparent *why* LSA groups certain concepts together since it is purely a mathematical procedure (as opposed to an intuitive semantic procedure). This is not necessarily a drawback, however it can make the process hard to improve upon, since its working is not very intuitive.

### 2.4.2 Probabilistic retrieval: Okapi BM25

The probabilistic model relies fundamentally on the probability ranking principle introduced in Section 1.4. In short, the problem is to estimate the probability that a given document in the system is relevant to a user's query. Documents are ranked and presented to users in decreasing order of their estimated probability of relevance.

If we adopt a binary notion of relevance, then a document $d$ is either relevant to a query $q$ (relevance $r = 1$), or it is non-relevant ($r = 0$). The basic problem in the probabilistic model is to estimate the probability of $P(r = 1|d, q)$.

The Binary Independence Model (BIM) [27] introduces several (quite severe) simplifying assumptions which make estimating this probability practical. These can be summarised as [6, p. 231]:

1. A Boolean representation is used for documents, queries, and relevance.
2. Terms are considered to be independent (i.e. the BIM assumes words are not context-dependent).
3. Terms that are not in the query do not affect the outcome.
4. Document relevance values are independent.

Given a query $q$ and document $d$, consider the problem of estimating the probability $P(r|d, q)$, i.e. the probability that a given document $d$ is relevant ($r = 1$) to some query $q$. This can be separated into the two probabilities, $P(r = 1|d, q)$, i.e. $d$ is relevant to $q$, and $P(r = 0|d, q)$, i.e. $d$ is not relevant to $q$. Since a document is either relevant or non-relevant to a query, we have $P(r = 1|d, q) = 1 - P(r = 0|d, q)$.

Using Bayes' rule, the basic problem of estimating $P(r|d, q)$ presented above can be written as:

$$P(r = 1|d, q) = \frac{P(d|r = 1, q)P(r = 1|q)}{P(d|q)}, \text{ and}$$
$$P(r = 0|d, q) = \frac{P(d|r = 0, q)P(r = 0|q)}{P(d|q)}.$$

However, these probabilities are difficult to estimate in practice. In the probabilistic model, documents are ranked based on their decreasing probability of relevance. We are normally more interested in the rank order of relevance for documents given a query, than the exact probability of relevance. Thus, to eliminate the common denominator $P(d|q)$ in the above two equations, we can combine them into an odds ratio which is monotonic with the probability of relevance, and therefore preserves the rank order. The odds $O(p)$ of an event with a probability $p$ is computed as $\frac{p}{1-p}$. By recognising that $P(r = 1|d, q) = 1 - P(r = 0|d, q)$, we can write

$$O(r|d, q) = \frac{P(r = 1|q)}{P(r = 0|q)} \cdot \frac{P(d|r = 1, q)}{P(d|r = 0, q)} \tag{2.4.4}$$

Using the above equation as starting point, it can be shown [6, pp. 224-225] that one can derive a rank-preserving approximation (the Retrieval Status Value (RSV)) of the equation given above, which estimates the odds of relevance of a document to a query, as a summation of values computed over the query terms:

If we define $\mathcal{T}_{d,q}$ as the set of terms common to both document $d$ and query $q$, $p_t$ as the probability of term $t$ appearing in a *relevant* document, and $u_t$ as the probability of term $t$ appearing in a *non-relevant* document, then we can write the RSV for $d$ and $q$ as follows:

$$\mathrm{RSV}_{d,q} = \sum_{t \in \mathcal{T}_{d,q}} \log \frac{p_t(1-u_t)}{u_t(1-p_t)} \tag{2.4.5}$$

From the above, for the rest of this discussion, let

$$c_{t,d} = \log \frac{p_t(1-u_t)}{u_t(1-p_t)} = \log \frac{p_t}{(1-p_t)} + \log \frac{(1-u_t)}{u_t} = \log O(p_t) - \log O(u_t). \tag{2.4.6}$$

$c_{t,d}$ represents the theoretical BIM value of relatedness between query term $t$ and document $d$. Therefore, $c_{t,d}$ summed over all terms in $\mathcal{T}_{d,q}$ represents the relatedness of document $d$ to query $q$.

Next we consider how to estimate $c_{t,d}$.

In IR, relevance feedback (feedback from users on which of the documents that were returned are relevant to their query) is used to make the retrieval models more accurate. Under normal circumstances, these ratings of relevance are hard to obtain. Therefore Web search engines make use of implicit feedback, such as which of the returned links users choose to click on, and so forth.

In the absence of relevance feedback, we can make the assumption that relevant documents are a very small percentage of the document collection. Thus $u_t$ (the probability of a term occurring in a non-relevant document) can be approximated as the ratio of documents in the corpus that contain the term.

If we use $\mathrm{df}_t$ for the document frequency of term $t$, and $N$ for the number of documents in the corpus, we can write this ratio as $u_t = \mathrm{df}_t/N$, and the second term in Equation 2.4.6 can be approximated as

$$\log \frac{(1-u_t)}{u_t} = \log \frac{(N-\mathrm{df}_t)}{\mathrm{df}_t} \approx \log \frac{N}{\mathrm{df}_t}$$

for large $N$. Croft and Harper [41] furthermore make the assumption that $p_t$ (the probability of a term $t$ occurring in a relevant document) is constant, and $p_t = 0.5$. This reduces the first term in Equation 2.4.6 to 0, since $\log \frac{p_t}{1-p_t} = \log \frac{0.5}{1-0.5} = \log 1 = 0$. This reduces $c_t$ to the well known term-frequency inverse document-frequency (tf-idf) [17] heuristic,

$$c_t = \log \frac{N}{\mathrm{df}_t}. \tag{2.4.7}$$

The BIM was originally designed, and works reasonably well, for short pieces of text [6]. However, for longer pieces of text, statistics like term frequency and document length play an important role. The **Okapi BM25** weighting scheme [28] aims to incorporate these statistics by factoring them into the approximate equation given by Equation 2.4.7 to yield

$$RSV_{d,q} = \sum_{t \in \mathcal{T}_{d,q}} \log \left[\frac{N}{\mathrm{df}_t}\right] \cdot \frac{(K_1+1)\mathrm{tf}_{t,d}}{K_1((1-b)+b(L_d/L_{\mathrm{ave}}))+\mathrm{tf}_{t,d}}, \tag{2.4.8}$$

where $\mathrm{tf_{t,d}}$ is the frequency of term $t$ in document $d$, and $L_d$ and $L_{\mathrm{ave}}$ are the length of document $d$ and the average document length for the whole collection. $K_1$ is a positive tuning parameter that calibrates the document term frequency scaling. $b$ is another tuning parameter ($0 \leq b \leq 1$) which determines the scaling by document length.

### 2.4.3 Query-likelihood statistical language models

Query-likelihood statistical language models (LMs) attempt to estimate the likelihood that the current document $d$ can generate the given query $q$, given some generative language model $M_d$ which describes $d$. These probabilities are estimated and a ranked list of documents is returned in decreasing order of their probabilities [30].

In essence, a statistical language model (LM) is a probability distribution over a word sequence which tries to describe the 'language' of the domain in question by predicting the probability of observing a term $t_i$, given a set of the $n$ preceding terms, $\{t_{i-1}, t_{i-2}, ..., t_{i-n}\}$. We call such a model an $n$-gram language model. We can use the chain rule to decompose the probability of a sequence of events into the probability of each successive event conditioned on earlier events:

$$P(t_1 t_2 t_3 \cdots t_n) = P(t_1)P(t_2|t_1)P(t_3|t_1 t_2) \cdots P(t_n|t_1 t_2 \cdots t_{n-1}) \tag{2.4.9}$$

The simplest form of language model, called the *unigram language model*, simply discards all conditioning context and estimates each term independently, i.e.

$$P_{\text{uni}}(t_1 t_2 t_3 \cdots t_n) = P(t_1)P(t_2)P(t_3) \cdots P(t_n) \tag{2.4.10}$$

Similarly, a bigram language model discards all context but the direct preceding word:

$$P_{\text{bi}}(t_1 t_2 t_3 \cdots t_n) = P(t_1)P(t_2|t_1)P(t_3|t_2) \cdots P(t_n|t_{n-1}). \tag{2.4.11}$$

Such a LM can capture any potential local dependency between two adjacent terms – i.e. if it were trained on a corpus in which the phrase *'New York'* appeared more often than expected by chance, it would typically be able to assign a much higher probability to the next term in the phrase *'New _'* being *'York'* than the unigram model shown above, which discards any contextual information.

In the simplest case, these probabilites are estimated using some parameter estimation function, e.g. a maximum-likelihood estimator, which seeks a model which gives the observed data the highest likelihood (and is thus prone to over-fitting). Maximum-likelihood estimators in the LM sense typically reduce to term counting [42],

$$P(t|M_d) = \frac{\text{tf}_{t,d}}{L_d}, \tag{2.4.12}$$

where $M_d$ is the language model to estimate, $\text{tf}_{t,d}$ is the count of term $t$ in document $d$, and $L_d$ is the length of document $d$ (number of words in the document).

Furthermore, since maximum-likelihood estimators cannot assign probabilities to terms it has not observed, one typically has to account for the zero probability problem where an unobserved word receives a likelihood of 0 in the LM. However, since it was only trained on a small subset of data, such an assumption is usually invalid and 'smoothing' techniques are used to spread the probabilities more evenly and to avoid zero probabilities for words [42].

Language models therefore consider documents and queries as one and the same, as opposed to treating them as two separate entities as in the probabilistic model discussed in Section 2.4.2. This results in a conceptually simple, computationally tractable (in the case of unigram and bigram LMs), and intuitively appealing model. However, the computational cost of more complex LMs is a concern for large-scale retrieval applications [43].

Computational cost tends to increase near-exponentially with the complexity of the LM, and simple unigram language models are unable to allow language mismatch between the query language and the document language (e.g. using synonyms or equivalent phrases). Nonetheless, applications still rely mostly on simple unigram and bigram models which are computationally more tractable [44].

## 2.5 Approaches using knowledge bases and lexical resources

In this section we introduce approaches that rely on more explicitly coded sources of background knowledge (from category III, as identified in the beginning of this chapter). The approaches outlined in the previous section all attempt to model or extract relatedness between terms via their statistical distributional properties found in unstructured sources of text. The approaches in this section make use of resources which are specifically created to reflect these relationships (such as the electronic thesaurus WordNet [45]), or in which these relationships develop as a natural result of the organisation of the resource (as with the online encyclopaedia Wikipedia [15]).

The term 'ontology' refers to an explicit specification of the entities in a specific domain and the relationships between those entities. A typical form of encoding knowledge in such a domain is called a semantic network. A semantic network is a graph where vertices represent concepts and where edges represent relations between concepts. **WordNet** is an example of a taxonomic lexical resource, where lexical relations are used to show the relationships between words. Lexical relations denote the patterns of association between lexical items in a language [46]. These may include:

1. Synonymy, which can be seen as the identity lexical relation. Nonetheless, it ranges in specificity from absolute synonymy where two words are identical, to words which can be described as having an *is-a* or *is-a-kind-of* relationship (hypernymy and hyponymy). For instance 'scarlet', 'vermilion', 'carmine' and 'crimson' can be seen as hyponyms (more specific instances) of 'red'; 'red' being their hypernym (a more general 'class' word);
2. Antonymy, the relation of having the opposite meaning; and
3. Polysemy, defined as the same word having multiple, distinct senses. 'Word sense disambiguation (WSD) systems' are faced with distinguishing the intended sense in the given context.

Several algorithms have been proposed for quantifying the relatedness between two concepts, given their representation in some taxonomical resource such as WordNet [47, 48, 49, 21]. A knowledge-based approach mines the strength of the relationship between two concepts based on how they are encoded in the underlying knowledge base or ontology. Therefore, it is evident that, regardless of the metric used on the ontology, the efficiency of the metric depends on the quality and the completeness of the underlying ontology: If there is no representation for a concept in the ontology, nothing can be said about its relationship to any other concept.

WordNet has long been the *de facto* lexical resource, however, it is considered to be rather pedantic in its classification scheme [50]. For instance, a `Dog` is encoded as a `Canine`

which is a `Carnivore` which is a `Placental Mammal`, however WordNet does not contain the relationship that a dog is a `Pet`.

This lack of common-sense knowledge is addressed in **ConceptNet** [50], which tries to bridge the gap between explicit knowledge such as 'Paris is the capital of France', and common-sense knowledge, such as 'when you stub your toe it hurts', and 'it is not nice when it hurts'. From these it attempts to draw common-sense conclusions, such as 'therefore try not to stub your toe'. However, this added generality comes at the expense of specificity, such as its lack of several named entities and domain-specific knowledge. For instance ConceptNet might capture the relationship between `Actor` and `Person`, but not between a named entity such as `Marlon Brando` and `Actor`.

One resource which provides a good balance between concept coverage, generality and specificity is the online encyclopaedia **Wikipedia** [15]. It is based on the 'wiki' concept where anyone can edit and revise the encyclopaedia. It is freely available, constantly evolving to reflect what is important for a large subset of society and tends to provide a more neutral focus since people with vastly different opinions can update its definitions until it reaches an equilibrium where most contributors are happy with it. However, this added freedom amounts to less well-defined semantic relations between concepts.

Wikipedia represents an ever-growing network denoting relationships between concepts closely modelled on how humans model the world. Thus, if properly mined, one can use this resource to differentiate among ambiguous concepts (word sense disambiguation), find the similarity between concepts, find concepts related to certain concepts, etc.

There are mainly three significant approaches in the literature which deal specifically with computing semantic relatedness using Wikipedia. These are: WikiRelate! [51], Explicit Semantic Analysis [1], and the Wikipedia-Link based Measure [2]. All three approaches assume that an article in Wikipedia represents a single concept, for instance the article on 'Dogs' represents the concept of a `Dog`. We will now briefly look at each of these approaches.

### 2.5.1 WikiRelate!

**WikiRelate!** [51], was one of the first attempts to compute semantic relatedness between concepts using Wikipedia. Wikipedia has a taxonomic structure since 2004 by providing categories with which to tag pertaining articles. Categorical relations in Wikipedia, however, cannot simply be seen as *is-a* links in a taxonomy, since they can denote other lexical relations such as meronymy (a *part-whole* relation, e.g. 'wheel' is a meronym of 'wagon') as well.

Strube and Ponzetto [51] liken Wikipedia's structure to that of a *folksonomy*, namely a collaboratively-edited knowledge source that allows users to categorise the content of the encyclopaedic entries. Broadly speaking, their approach first resolves words to candidate articles. Then, these articles are linked to one another via Wikipedia's category structure. For instance, if `Dog` and `Cat` are categorised under `MAMMALS`, they are linked as `Dog`→`MAMMALS`←`Cat`. Finally their relatedness is computed using path-based metrics originally developed for WordNet [47, 48, 49, 21].

As a first attempt, their methods were fairly successful and scored better than WordNet-based methods on representative datasets [51]. However, WikiRelate! is significantly outperformed by ESA and WLM, which we introduce next.

### 2.5.2 Explicit Semantic Analysis (ESA)

Gabrilovich and Markovitch proposed a new method in 2007, called **Explicit Semantic Analysis (ESA)** (see Figure 2.1). It is inspired by the *'desire to augment text representation with massive amounts of world knowledge'* [1].

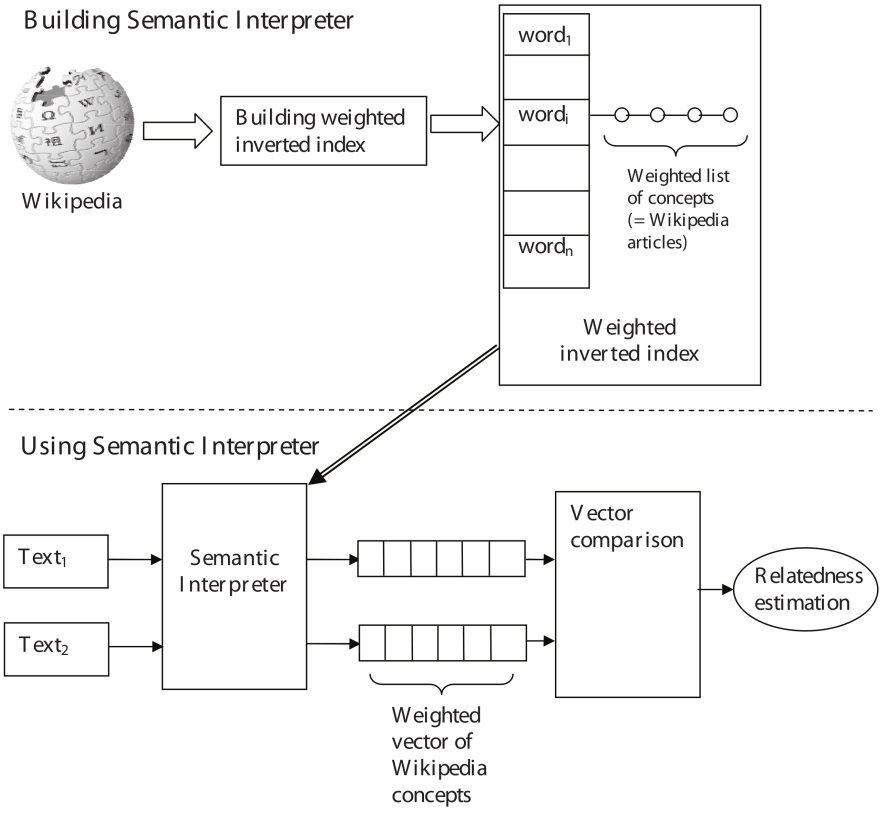



**Figure 2.1:** The ESA semantic interpretation process (reproduced from [1]).

Concepts are represented by their associated articles in Wikipedia. Essentially, they use a vector space model from IR to map the terms occurring in the various Wikipedia articles to the concepts (articles) they refer to. Given two input texts to relate to one another (this can be two words, phrases or documents), an *interpretation vector* is constructed for each text by performing an information lookup using the terms it contains. This produces a vector of article identifiers from Wikipedia and a weight denoting how strongly the input text belongs to the content of each article – what they call a weighted vector of 'Wikipedia concepts' [1]. Computing semantic relatedness of text then becomes comparing their respective interpretation vectors in this Wikipedia-based concept space, using some vector comparison technique, such as cosine correlation (see Section 2.4.1).

### 2.5.3 Wikipedia Link-based Measure (WLM)

The third Wikipedia-based method is the **Wikipedia Link-based Measure (WLM)**. The main difference between WLM and other Wikipedia-based approaches is its use of Wikipedia's hyperlink structure. Using the hyperlinks in and out of candidate concept articles, they link one article (concept) to all other concepts it is linked to. WLM uses

this hyperlink structure to define relatedness, something they argue makes their measure cheaper (since one can ignore the vast textual content of Wikipedia's articles) and more accurate than ESA (since, they argue, it is more closely tied to the manually defined semantics of the resource) [2].

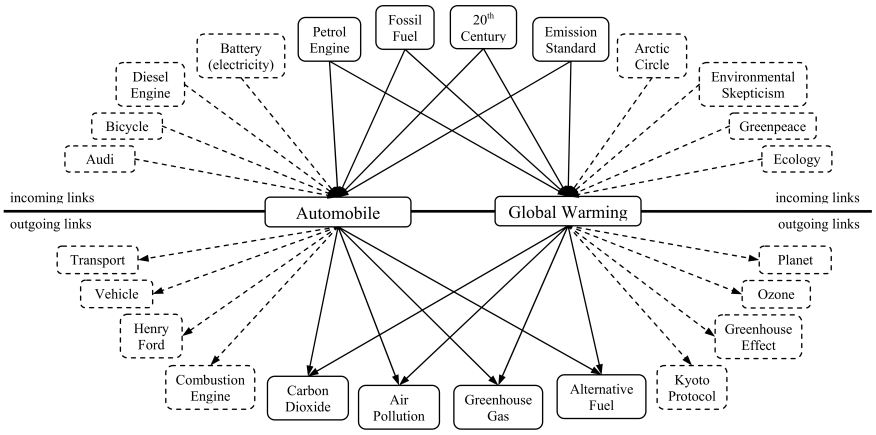



**Figure 2.2:** Obtaining a semantic relatedness measure between `Automobile` and `Global Warming` from Wikipedia links using WLM (reproduced from [2]).

In Figure 2.2, Milne and Witten graphically illustrate their WLM approach by showing a small percentage of the links relating `Automobile` and `Global Warming`.

In their approach to measure semantic relatedness between two concepts they first identify the articles which best represent them. In doing this, one is always faced with two problems, namely *polysemy* (one word representing different concepts) and *synonymy* (different words referring to the same concept).

They use *anchors*, the terms or phrases in the text to which links are attached (i.e. the text between an `<a...>` and `</a>` HTML tag), to differentiate candidate articles. To measure relatedness between article pairs, they use a metric inspired by the Normalised Google Distance (NGD) (see [22]; related to the Normalised Compression Distance (NCD), see Section 2.3.2).

If we define $G$ as the entire Wikipedia hyperlink graph structure, $v_i$ and $v_j$ as two articles of interest (i.e. two 'concepts'), and $\mathcal{V}_i$ and $\mathcal{V}_j$ as two sets containing the identifiers of all articles that link to $v_i$ and $v_j$ respectively, then the similarity between $v_i$ and $v_j$ using the WLM can be defined as

$$\text{sim}_{\text{WLM}}(\mathcal{V}_i, \mathcal{V}_j) := \frac{\log\left(\max(|\mathcal{V}_i|, |\mathcal{V}_j|)\right) - \log(|\mathcal{V}_i \cap \mathcal{V}_j|)}{\log(|G|) - \log(\min(|\mathcal{V}_i|, |\mathcal{V}_j|))}. \tag{2.5.1}$$

where $|\mathcal{V}_x|$ denotes the number of elements in $\mathcal{V}_x$. This is referred to as the Wikipedia Link-based Measure (WLM) of semantic relatedness.

### 2.5.4 Comparison of Wikipedia-based measures

Table 2.1 compares the three main Wikipedia-based methods discussed in this section and their Pearson correlations with human judgements over three standard datasets. This clearly identifies ESA as the best-performing measure in terms of accuracy, and

**Table 2.1:** Correlations of the three main Wikipedia-based relatedness measures with human judgements (from Witten [2]).

| Dataset | WikiRelate! | ESA | WLM |
|---|---|---|---|
| Miller and Charles | 0.45 | 0.73 | 0.70 |
| Rubenstein and Goodenough | 0.52 | 0.82 | 0.64 |
| WordSimilarity-353 | 0.49 | 0.75 | 0.69 |
| Weighted average | 0.49 | 0.76 | 0.68 |

WikiRelate! as the worst performing by far. However, there are various pros and cons to keep in mind regarding ESA and WLM:

ESA is more robust than WLM since it only requires text to be in the target article's body text in order to be mapped to the concepts it represents. However, this additional robustness requires all the article text of Wikipedia to be indexed (almost 21 GB uncompressed as of 2009). Each term in every article must be weighted, sorted, and pruned and an inverted index must be constructed from this, which results in a very large database of roughly three million indexed documents.

There are two advantages to WLM in comparison to ESA. Firstly, WLM only requires the link structure, anchor text, and article headings from the Wikipedia database. This results in less than one tenth of the full Wikipedia database, as required by ESA. Secondly, WLM has been converted into an open-source project [52] and a freely accessible Web Service [53] with an active group of contributors who can provide technical assistance when needed, while no known reference implementation or active project exists for ESA.

## 2.6 Chapter summary

This chapter introduced the major areas of background information used thoughout the rest of this thesis. We introduced the Normalised Compression Distance (NCD) distance-metric, the Vector Space Model (VSM), Okapi BM25 retrieval model, and query-likelihood statistical language models, since these methods will be evaluated against human ratings in Chapter 7.

We also introduced the three main approaches that make use of the Wikipedia online encyclopaedia to extract semantic relationships between concepts and documents, namely WikiRelate!, WLM and ESA.

# Chapter 3

# Inter-concept similarity: Spreading activation over Wikipedia's hyperlink structure

This chapter addresses our first hypothesis, namely that interpreting Wikipedia's hyperlink structure as an associative network can be used for accurately calculating similarity between concepts. We discuss how the Wikipedia hyperlink structure is modelled as an associative network. Pure spreading activation is introduced as a technique for processing associative networks. Several strategies are proposed for overcoming some of the problems associated with using SA over the Wikipedia hyperlink graph. The Spreading Activation (SA) model is adapted to function over the hyperlink structure of Wikipedia and will be evaluated over the WordSimilarity-353 word-pair similarity dataset [33] in Chapter 7.

## 3.1 Introduction

As discussed in Section 2.5, Explicit Semantic Analysis (ESA) can be viewed as mapping document text into a Wikipedia 'concept space' via a Vector Space Model (VSM) based information retrieval process. This results in vectors containing identifiers to Wikipedia articles ('concepts'), and weights denoting how strongly the text is seen to relate to the content of those articles. Similarity is computed as a cosine correlation between the concept vectors of two articles.

The Wikipedia Link-based Measure (WLM) was also introduced in Section 2.5. In this approach an article (viewed as a 'concept') is described by the vector of article identifiers that link directly to it. For two concepts (articles), similarity is computed as the information overlap between their descriptions, using a formula derived from Lin's [21] information theory based Similarity Theorem introduced in Section 2.3.1.

ESA might miss certain associations since it uses a purely keyword-based mapping from article text to Wikipedia concept vectors [1]. However, it is currently the most accurate Wikipedia-based method. WLM might also miss certain semantic or conceptual associations between concepts since it only considers one level of intermediate nodes between concept articles [2]. As we will argue later on, links of two or even three intermediate links can provide additional indications of semantic relatedness.

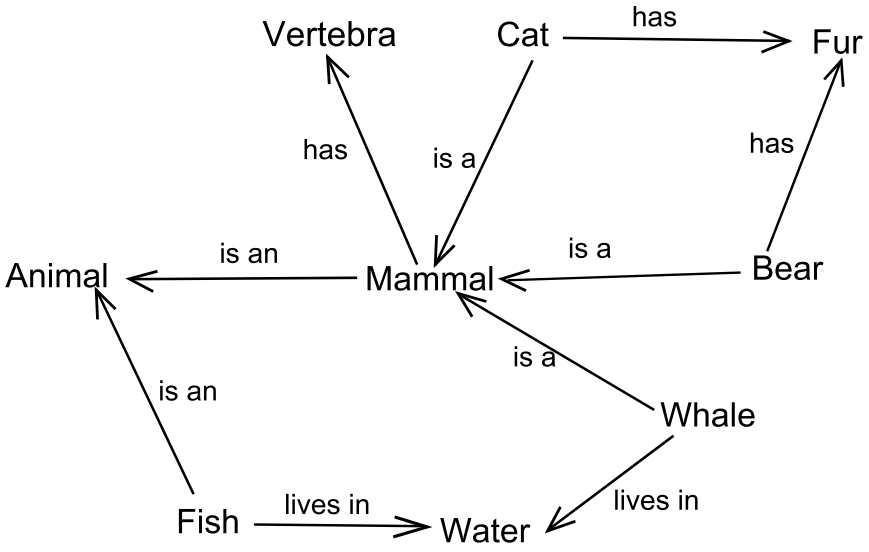



**Figure 3.1:** Example of a semantic network.

Our approach to measure the relatedness between two concepts in Wikipedia relies fundamentally on viewing the Wikipedia hyperlink structure as an associative network. This view is strengthened by the success of the WLM method which considers only one level of intermediate nodes. An associative network (AN) is a more general form of a semantic network. Semantic networks were introduced in 1968 by Quillian [31] and have since played a significant role in knowledge representation [54].

In their original definition, semantic networks express knowledge in terms of concepts, their properties, and the hierarchical sub-superclass relationships between concepts. Each node in the network represents a concept, and the hierarchical relationships between concepts in the network are represented by connecting appropriate nodes in the network with for instance *is-a* or *instance-of* links [55]. Nodes at lower levels generally represent individuals (such as `Whale` in Figure 3.1) while nodes at higher levels represent classes or categories of individuals (such as the category `Mammal`).

An AN is a generic semantic network with information items or concepts represented by nodes, and with only one type of link between nodes representing the level of association between nodes, usually as some real-valued weight.

We compute semantic relatedness over the Wikipedia hyperlink structure using a technique called Spreading Activation (SA), which is introduced in Section 3.2. We will use this approach to evaluate our initial hypothesis stated at the beginning of this chapter.

In short, SA can be described as supplying an initial real-valued 'energy' to one or more concept nodes in an AN, and to iteratively spread this energy via their connections through the network. Energy is distributed based on how strongly nodes are associated with one another. When certain termination conditions are met, relatedness is measured as an interpretation of the activations of the nodes that are activated, for which certain strategies are required (introduced later).

There are three problems which need to be addressed:

1. The Wikipedia hyperlink structure needs to be extracted and modelled as an associative network. Strategies are required to compute the weights which link two concept nodes in the AN, since these do not exist in the Wikipedia hyperlink struc-

ture.

2. The spreading activation technique needs to be adapted and implemented to function over this AN.

3. Strategies are required for translating the node activations which result from the SA process, into scores of relatedness between the initial nodes.

In the rest of this chapter, we systematically develop the fundamentals of our approach by addressing and offering solutions to the problems identified above.

## 3.2 Pure spreading activation

The techniques we develop to compute semantic similarity between concepts and documents in the rest of this thesis rely on knowledge encoded in an associtive network data structure, and a processing technique called Spreading Activation (SA).

Spreading Activation (SA) is a technique for searching and traversing network data structures, and is based on the supposed mechanisms of human memory operations [32]. In its pure form it consists of a conceptually simple, iterative processing technique on a network data structure such as a semantic or associative network.

Generally speaking, each iteration consists of one or more 'pulses', followed by a termination check. We will look at this process in a bit more detail below, but quite simply, the idea is to supply one set of nodes with an initial amount of real-valued 'energy', and to iteratively 'spread' this energy to all nodes that are connected to them. The spread of energy from one node to another is governed by the strength of association between these two nodes and some other factors. When a set of termination checks are reached, the set of activated nodes in the network is interpreted based on the specific application.

To be more precise, each pulse consists of three stages: preadjustment, spreading, and postadjustment [18]. During pre- and postadjustment some form of activation decay is optionally applied to the active nodes. This serves to avoid retention of activation from previous pulses and, from a connectionist point of view, models 'loss of interest' when nodes are not continually activated.

The spreading phase consists of a number of energy transfers from one node to all its neighbour nodes. For instance, consider the spreading phase associated with node $v_j$ in Figure 3.2. Let $a_{j,\text{in}}$ denote the total real-valued activation input for node $v_j$, and $\mathcal{N}(v_j)$ denote the set of $v_j$'s neighbour nodes and $a_{i,\text{out}}$ the activation output of a node $v_i$ connected to node $v_j$. Furthermore, let $w_{ij}$ denote the strength or weight of connection between nodes $v_i$ and $v_j$. We can then compute the activation input value of node $v_j$ using the following formula:

$$a_{j,\text{in}} = \sum_{v_i \in \mathcal{N}(v_j)} a_{i,\text{out}} w_{ij}. \tag{3.2.1}$$

After computing a node's input value, we can compute its output value as a function of its input value, e.g.

$$a_{j,\text{out}} = f(a_{j,\text{in}}), \tag{3.2.2}$$

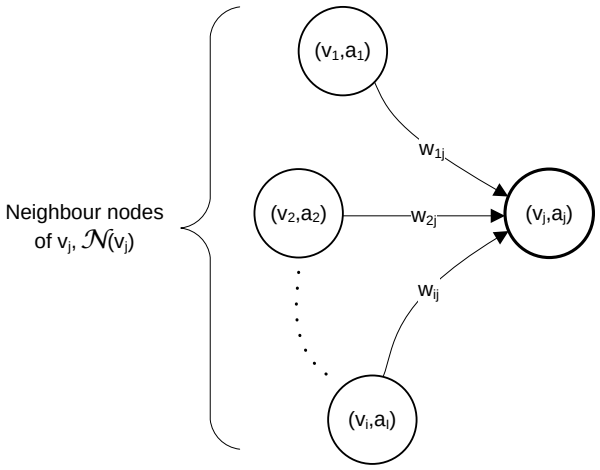



**Figure 3.2:** Example graph for spreading activation from nodes $\mathcal{N}(v_j) = \{v_1, \ldots, v_i\}$ to node $v_j$. $a_x$ denotes the real-valued activation value for node $v_x$. $w_{ij}$ is the weight of the link connecting nodes $v_i$ and $v_j$.

where $f(\cdot)$ can be any function, although most frequently used functions include a linear function, step function or sigmoidal function [18]. Of these, a thresholding (step) function is most commonly used to determine if node $v_j$ is considered 'active' or not, and is simply defined as

$$a_{j,\text{out}} = \begin{cases} 0 & \text{if } a_{j,\text{in}} < T_j \\ a_{j,\text{in}} & \text{otherwise} \end{cases} \tag{3.2.3}$$

where $T_j$ is the threshold value associated with node $v_j$.

Activation spreads across the network pulse after pulse, reaching nodes far away from the initially activated nodes. After every predetermined number of pulses, a set of termination conditions are checked and if they are met, spreading stops. Otherwise it continues for another series of pulses. SA is therefore iterative, consisting of a sequence of pulses and termination checks.

The result of the SA process is a set of activated nodes and their respective activations at termination time. The interpretation of the level of activation is dependent on the application and something we will discuss in more depth when we discuss spreading strategies in Section 3.5.

## 3.3 Modelling Wikipedia's hyperlink structure as a directed graph

Everyone is free to edit articles in Wikipedia. Each article covers one specific topic in detail. Certain topics can be ambiguous, such as 'Table' which could refer to either a type of furniture or for instance a formatting structure in HTML. In such cases, a *disambiguation page* is automatically created where a user can select the specific meaning of the word they are referring to. Users are free to insert links in any article to other related Wikipedia entries. This naturally leads to a structure of links connecting articles to other articles related in either a more specific sense (hyponyms) or in a more general sense (hypernyms).

There are, however, certain links with questionable relatedness to articles they link to. Hyperlinks link a page $A$ to a page $B$, and are thus *directed*. We can model Wikipedia's hyperlink structure using standard graph theory as a *directed graph* $G$, consisting of a set of vertices (pages) $V$ and a set of edges (hyperlinks) $E$, each connecting one vertex $v_i$ in $V$ to another vertex $v_j$. The set of edges $E$ thus consists of a set of two-element subsets of $V$. For consistency, we shall refer to edges (hyperlinks) as links, and vertices (pages) as nodes.

In this model, each Wikipedia article is seen to represent a single *concept* and the hyperlink structure relates these concepts to one another. In order to compute similarity (strictly speaking we are measuring 'strength of association') between two concepts, we rely on the fundamental principle of an associative network, namely that it connects nodes that are associated with one another via weighted links denoting how strongly these nodes relate to one another.

However, Wikipedia was not created as an associative network, it was primarily created as an online encyclopaedia. Therefore, no weights denoting strength of association exist in the hyperlink structure, and we will have to deduce these automatically. We will discuss this matter in more depth later on when we discuss weighting schemes.

In an associative network, the strength of association between two nodes $v_i$ and $v_j$ is captured by the paths that exist in the network between these two nodes. A 'path' between $v_i$ and $v_j$ is simply a set of distinct links that connect $v_i$ and $v_j$. A direct link between $v_i$ and $v_j$ (i.e. a path consisting only of the link $v_iv_j$) is seen as the strongest association between them. A path with one intermediate node indicates a weaker association, etc.

It is therefore important to consider the directionality introduced by the hyperlink structure, since it can limit the number of connections one might discover between two nodes (as will be shown below). This might in turn affect the accuracy with which relatedness can be computed, since the measures we will introduce later on rely on accurately discovering the paths between concepts. In a directed graph, a node $v_i$ and $v_j$ can link to each other in four ways:

1. Directly, i.e. $v_i \leftrightarrow v_j$,
2. Both can link to the same node, $v_k$, i.e. $v_i \rightarrow v_k \leftarrow v_j$,
3. Both can be linked to from the same node, i.e. $v_i \leftarrow v_k \rightarrow v_j$,
4. Links in opposite directions, i.e. $v_i \leftarrow v_k \leftarrow v_j$ and $v_i \rightarrow v_k \rightarrow v_j$.

A directed search can only detect paths of types 1 and 4. In order to find paths of type 2 and 3, an *undirected search* is required.

Milne and Witten found their best results using links *into* pages [2], an approach we also followed. Therefore, instead of directly using the graph of hyperlinks out of pages which connect one page with another, we make use of the graph of hyperlinks into pages by reversing all links in the former graph.

## 3.4 Adapting spreading activation for Wikipedia

The pure model of spreading activation as discussed in Section 3.2, has several significant problems [18], the most important in our case being that activation can spread over the

entire network unless carefully controlled in the pre- and post-adjustment phases.

It is therefore critical to impose certain **constraints** on the spreading process to keep it under control, and in order to keep the activations 'meaningful'. In other words, to ensure that activation only spreads between nodes that are semantically connected in a meaningful way.

In an associative network generated specifically to model a specific domain, this is not such a big problem since the network is constructed with this in mind. However, as Crestani points out [18], it is very time-consuming to construct a network of associations among information items when the size of the collection is very large. That is one of the reasons why SA has not enjoyed such widespread use.

We do not have that problem, since we extract the network from the collaboratively-created Wikipedia where users created hyperlinks (connections in our associative network model) for us. However, this leads to a lower-quality network from which the meaningful information must be extracted. We address these problems and our proposed solutions in the following paragraphs and sections.

Crestani mentions three specific constraints which can be applied to the spreading process [18]:

1. **Distance** constraint – spreading should cease when it reaches nodes far away from the initially activated nodes.

2. **Fan-out** constraint – spreading should cease at nodes with very high degree or connectivity (i.e. connected to many other nodes).

3. **Path** constraints – activation should spread using preferential paths, reflecting application dependant inference rules.

4. **Activation** constraint – activations should be regulated via altering the threshold function based on the total activation level of the network, making it possible to implement several complicated inference rules (in a semantic network where different links indicate different semantic relations).

We discard constraint number 4, since it would significantly increase the computational time required for the algorithm, and we do not have a sufficient grammar of inference rules implemented in our simple associative network to make use of this. However, constraints 1–3 are important to avoid flooding the network with negligible activations, and we model these constraints using three parameters, namely network decay $K_{\text{decay}}$, link weight $w$, and node activation threshold $T$.

Recall from Equation 3.2.1 and Equation 3.2.3, that for a node $v_j$, with $\mathcal{N}(v_j)$ denoting the set of all $v_j$'s neighbours, the basic spreading operation can be given as

$$a_{j,\text{in}} = \sum_{v_i \in \mathcal{N}(v_j)} f(a_{i,\text{in}}) \cdot w_{ij}, \tag{3.4.1}$$

with $f(\cdot)$ some thresholding function. In the following sections we address how these constraints were implemented.

### Path constraint

At every step of the spreading process, we multiply a node's activation value with a network decay parameter $0 \leq K_{\text{decay}} \leq 1$ – i.e. we replace $w_{ij}$ in Equation 3.4.1 with $w_{ij} \cdot K_{\text{decay}}$. This decays activation exponentially in the path length. For path length of one, activation is multiplied by $K_{\text{decay}}$, for a path length of two, activation is multiplied by $K_{\text{decay}} \cdot K_{\text{decay}} = K_{\text{decay}}^2$, etc. This penalises activation transfer over longer paths (constraint one above).

The intuition, as we hinted at above, is that a shorter path shows a stronger association between two concepts in an associative network. For instance, if there exist two paths between `Cat` and `Dog`, namely `Cat` → `Nine lives` → `9Lives` → `Cat Food` → `Pet food` → `Pet` → `Dog` and the other being `Cat` → `Dog`, the latter is more indicative of a strong semantic relationship in our model, than the former.

### Fan-out constraint

In the model presented in Equation 3.4.1, a real-valued weight is required to describe the strength of the link connecting nodes $v_i$ and $v_j$. We need to estimate this value ourself, since it is not contained in the hyperlink structure of Wikipedia. We refer to this as the *weighting strategy* and propose three potential approaches to this problem.

In the first approach, simply called the **pure Energy Distribution weighting scheme (ED) weighting scheme**, a node's weight is made inversely proportional to its degree (number of nodes it is connected to). This reduces the effect of very connected nodes on the spreading process (constraint number 2 above). For instance, the article `United States` is linked to from $322,000$ articles. Therefore, a path connecting two concept nodes through `United States` would not mean nearly as much semantically, as a path connecting them through, say, `Hair Pin`, which is only linked to by 20 articles. If $\mathcal{N}(v_i)$ is the set of nodes linking to node $v_i$, we model this by making $v_i$'s weight $w_{ij} = \frac{1}{|\mathcal{N}(v_i)|}$. Since $\mathcal{N}(v_i) \geq 1$ it holds that $0 < w_{ij} \leq 1$.

The second weighting strategy which we introduce is called the **Inverse Link Frequency weighting scheme (ILF)**. ILF is inspired by tf-idf (see Section 2.4.1). It is based on the idea that the more a feature is contained across all objects, the less discriminatory power it has to identify a specific object. Taking the degree of node $v_i$ as the number of articles that either link to $v_i$ or that it links to, we can say that the *link frequency* of node $v_i$ is its degree divided by the number of articles it could be connected to in the entire Wikipedia graph $|G|$. The log-smoothed, *Inverse* Link Frequency, then, is defined as:

$$\text{ILF}(v_i) \triangleq \log\left(\frac{|G|}{|\mathcal{N}(v_i)|}\right) \tag{3.4.2}$$

This has the added effect of 'boosting'[1] really unconnected nodes. For instance, if a path is made through a very **specific** node with a low degree, $\frac{|G|}{|\mathcal{N}(v_i)|}$ is very large and $\text{ILF}(v_i) > 1$, thus boosting that specific edge's weight.

To test the result of this boosting effect we also introduce a third weighting scheme, **normalised ILF (NILF)**, defined as:

[1] Increasing the contribution to relatedness made by that path – not related to the machine learning interpretation of *boosting*.

$$\text{NILF}(v_i) \triangleq \frac{\text{ILF}(v_i)}{\log(|G|)}. \tag{3.4.3}$$

since $\text{ILF}(v_i)$ reaches a maximum of $\log(|G|)$ for $|\mathcal{N}(v_i)| = 1$, we divide by $\log(|G|)$ to normalise $\text{ILF}(v_i)$ to the range $[0, 1]$.

#### Threshold constraint

Finally, we enforce these constraints through the use of a threshold parameter $0 \leq T \leq K_{\text{init}}$. We used $K_{\text{init}} = 1.0$. Activation ceases to spread from one node to the next node, when the activation value of the first node drops below $T$. This stops very weakly activated nodes from propagating their negligible activations.

## 3.5 Interpreting activations: Spreading strategy

After activation has finished spreading through the network (all termination conditions are met), we are left with a set of nodes in various levels of activation. Given two initial nodes $v_i$ and $v_j$, it is this set of activated nodes that we wish to interpret as an indication of strength of association (or relatedness) between $v_i$ and $v_j$.

This crucial step is lightly skipped over in most works dealing with SA we found in the literature, with most works advising that the interpretation is *'application specific'*. We approached this problem using two different approaches, based on two distinct hypotheses:

1. The strength of association between two nodes $v_i$ and $v_j$, can be measured as the ratio of the initial activation energy $K_{\text{init}}$ at the input node that reaches the target node after SA has terminated; and

2. The strength of association between two nodes $v_i$ and $v_j$ can be measured as the amount of *overlap* that exists between their individual *sets of activated nodes*, after SA has terminated.

#### Target Activation Approach spreading strategy (TAA)

We call the first approach the **Target Activation Approach** and it can be explained as follows:

Assume we want to compute the strength of association between $v_i$ and $v_j$. We supply $v_i$ with some initial activation energy $K_{\text{init}}$ and the activation of all other nodes including $v_j$ is set to 0. After the SA process has terminated, $v_j$ is activated with $a_{j,\text{incident}}$. We therefore compute the strength of association as the ratio $\frac{a_{j,\text{incident}}}{K_{\text{init}}}$.

#### Agglomerative Approach spreading strategy (AA)

The second approach is called the **Agglomerative Approach spreading strategy** since we agglomerate all activations into one score resembling relatedness. After spreading has terminated, relatedness is computed as the amount of overlap between the activation vectors of the inidividual nodes, using either the cosine similarity (called AA-cos), or an adapted version of the information theory based WLM [2] measure (called AA-wlm).

Assume the same set of initial nodes $v_i$ and $v_j$. Let $\mathbf{A}_k$ be the $N$-dimensional vector of real-valued activation values obtained by spreading over the $N$ nodes in the Wikipedia associative graph from node $v_k$. Therefore, $\mathbf{A}_k$ is in $\mathcal{R}^N$. We use $a_{kx}$ to denote the element at position $x$ in $\mathbf{A}_k$.

Furthermore, let $\mathcal{V}_k = \{v_{k1}, \ldots, v_{kM}\}$ be the set of $M$ nodes that are activated by spreading from $v_k$, i.e. the set of identifiers of the nodes with nonzero activations in $\mathbf{A}_k$ after spreading has terminated (and therefore $M \leq N$).

We then define the **cosine Agglomerative Approach** (henceforth called **AA-cos**) as

$$\begin{aligned} & \text{sim}_{\text{AA,cos}}(\mathbf{A}_i, \mathbf{A}_j) \\ = \; & \frac{\mathbf{A}_i \cdot \mathbf{A}_j}{||\mathbf{A}_i||\ ||\mathbf{A}_j||} \\ = \; & \frac{\sum_{k=1}^{N} a_{ik} a_{jk}}{\sqrt{\sum_{k=1}^{N} a_{ik}^2 \sum_{k=1}^{N} a_{jk}^2}}. \end{aligned} \tag{3.5.1}$$

For our adaptation of the Wikipedia Link-based Measure (WLM) approach to spreading activation, we define the **WLM Agglomerative Approach** (henceforth called **AA-wlm**[2]) as

$$\begin{aligned} & \text{sim}_{\text{AA,wlm}}(\mathcal{V}_i, \mathcal{V}_j) \\ = \; & \frac{\log(\max(|\mathcal{V}_i|,|\mathcal{V}_j|)) - \log(|\mathcal{V}_i \cap \mathcal{V}_j|)}{\log(|G|) - \log(\min(|\mathcal{V}_i|,|\mathcal{V}_j|))} \end{aligned} \tag{3.5.2}$$

with $|G|$ representing the number of nodes in the entire Wikipedia associative network, and $|\mathcal{V}_x|$ the number of node identifiers in $\mathcal{V}_x$. The difference between our AA-wlm and WLM [2] (see Equation 2.5.1) is in how the $\mathcal{V}_*$ sets are populated. WLM takes $\mathcal{V}_i$ to be the set of all nodes linking to $v_i$. AA-wlm takes $\mathcal{V}_i$ to be the set of all nodes activated after Spreading Activation from $v_i$ has terminated, using some configuration of parameters (which affect whether nodes will be activated or not).

Note that the AA-wlm method does not take final real-valued activations into account, while the AA-cos method does.

## 3.6 Spreading activation algorithm

An algorithm is required to implement the two SA approaches (TAA and AA) discussed in Section 3.5. It needs to satisfy the constraints that were introduced in Section 3.4, while operating over the Wikipedia associative network in the structure discussed in Section 3.3.

As was noted before, shorter paths contribute more towards a strong association between two concepts in an associative network. Furthermore, it has been shown that most 'interesting' nodes can be reached from every other node in an average of roughly 3.5 – 4.5 links or hops [56]. This excludes pages such as 'List of Asteroids' which form a chain of more than 70 articles, each only linking to the next article.

[2] *AA-wlm* is our adaptation of WLM [2] for SA, not to be confused with their method, which we simply call *WLM*.

For traversing the graph, one can use either a breadth-first or a depth-first approach. For normal graph search problems, where the goal is finding some solution node, these two approaches can be described as follows: Breadth-first search traverses all nodes connected to a given node, iteratively, level by level until a solution is found. Depth-first search goes down each path as far as possible, and then iteratively backs up and repeats the process, until a solution is found. Depth-first search tends to require less memory, since it only needs to record nodes on the 'current' path and not all nodes up to the current depth, as for breadth-first search (where memory requirements often grows exponentially in the depth) [57].

Furthermore, we limit the maximum depth to some $L_{p,\max}$[3] and we stop spreading when activation drops below a certain threshold $T$, which prevents the algorithm getting 'stuck' traversing down some highly-connected, albeit meaningless (in our model), path.

To compute $sim_{\mathrm{TAA}}(v_i, v_j)$, i.e. using the Target Activation Approach spreading strategy, an algorithm is required to traverse $G$ in a depth-limited fashion, starting with a node $v_i$, and spreading its activation to all its neighbour nodes and all their neighbours, etc., up to a certain maximum level $L_{p,\max}$. It has to implement the general spreading mechanism as described in Section 3.4 and by Equation 3.4.1 – i.e. it must consider the degree of nodes and network decay, and enforce a thresholding function.

The TAA similarity function $\mathrm{sim}_{\mathrm{TAA}}(v_i, v_j)$ is computed as the ratio $\frac{a_{j,\mathrm{incident}}}{K_{\mathrm{init}}}$. If a non-symmetric weighting function is used (i.e. if $w(v_i, v_j) \neq w(v_j, v_i)$) we spread from both $v_i$ and from $v_j$, and similarity is computed as the average target activations, $\frac{1}{2} \cdot (\frac{a_{j,\mathrm{incident}}}{K_{\mathrm{init}}} + \frac{a_{i,\mathrm{incident}}}{K_{\mathrm{init}}})$.

To compute $sim_{\mathrm{AA}}(v_i, v_j)$, i.e. using the Agglomerative Approach spreading strategy, an algorithm is also required to traverse $G$ in a depth-limited fashion, starting from a node $v_i$ and spreading activation as described above to all its neighbour nodes and all their neighbours, to produce the activation vector $\mathbf{A}_i$. However, the spreading process then has to be repeated from node $v_j$ to produce the activation vector $\mathbf{A}_j$. Finally, similarity is computed as the overlap between $\mathbf{A}_i$ and $\mathbf{A}_j$ using either the AA-cos method or the AA-wlm method.

From the above description it is evident that both these approaches rely on some method to spread activation outwards starting from some source node. TAA stops at this point and computes similarity as the ratio of energy received between source and target node. AA, however, repeats the process from the target node and uses the resulting activation vector overlap as an indication of similarity.

We therefore define a function `spread_unidir()` as shown in Algorithm 1, which takes as parameters an adjacency list graph structure $G$, a starting node in this graph $v_i$, a maximum level to which it is allowed to spread $L_{p,\max}$, a weighting scheme $\mathbf{W}$ such that $0 < w_{ij} < 1$ and $w_{ij} \in \mathbf{W}$ is the real-valued weight associated with the link between $v_i$ and $v_j$, a network decay parameter $K_{\mathrm{decay}}$, and a threshold value $T$. The activation vector $\mathbf{A}$ is an $N$-dimensional vector, updated in-place. $\mathbf{P}$ is a dynamic list of nodes in the path to $v_i$, to avoid cycles.

A function `spread_taa()` is also defined as shown in Algorithm 2. `spread_taa()` uses `spread_unidir()` to compute the similarity between $v_i$ and $v_j$ using the TAA as described above.

Lastly, we define `spread_aa()` as shown in Algorithm 3 which also uses `spread_unidir()`

[3] A depth-first search which is limited in its depth is referred to as a depth-limited search.

**Algorithm 1:** Pseudo code to spread activation depth-first from node $v_i$ up to level $L_{p,\max}$, using global decay $K_{\text{decay}}$, and threshold $T$, given an adjacency list graph structure $G$ and a weighting scheme $\mathbf{W}$ such that $0 < w_{ij} \in \mathbf{W} < 1$ for $w_{ij}$ the weight of the link between $v_i$ and $v_j$. $\mathbf{A}$ is a vector in $\mathcal{R}^N$ which is updated in-place. $a_x \in \mathbf{A}$ is the real-valued activation value of node $v_x$. $\mathbf{P}$ is a dynamic list which records the nodes up to $v_i$ to avoid cycles.

**Require:** $G, L_{p,\max}, K_{\text{decay}}, T$
  **function** SPREAD_UNIDIR($v_i, \mathbf{A}, \mathbf{P}$)
    **if** $a_i < T$ **then** ▷ Threshold constraint
      return
    **end if**
    Add $v_i$ to $\mathbf{P}$ ▷ To avoid cycles
    **for** $v_j \in \mathcal{N}(v_i)$ **do** ▷ Process $v_i$'s neighbours
      **if** $v_j \notin \mathbf{P}$ and $|\mathbf{P}| \leq L_{p,\max}$ **then**
        $a_j^* = a_j + a_i w_{ij} K_{\text{decay}}$
        Replace $a_j \in \mathbf{A}$ with $a_j^*$
        SPREAD_UNIDIR($v_j, \mathbf{A}, \mathbf{P}$)
      **end if**
    **end for**
    return
  **end function**

**Algorithm 2:** Pseudo code to measure strength of association between $v_i$ and $v_j$ using TAA. $\mathbf{A}$ is a zero-initialised $N$-dimensional vector ($N = |G|$).

**Require:** $G, L_{p,\max}, K_{\text{decay}}, T$
  **function** SPREAD_TAA($v_i, v_j$)
    Set $a_i \in \mathbf{A}$ to $K_{\text{init}}$ ▷ $K_{\text{init}} = 1.0$
    SPREAD_UNIDIR($v_i, \mathbf{A}, \emptyset$) ▷ $\mathbf{A}$ updated in-place
    return $(a_j / K_{\text{init}})$ ▷ $a_j$ is the energy that reached $v_j$
  **end function**

to compute the similarity between $v_i$ and $v_j$ using either the AA-cos or AA-wlm as described in Section 3.5.

## 3.7 Implementation considerations

A test-driven design (TDD) methodology was followed for the development of the core spreading modules. This method of software development calls for *first* writing the unit tests for a new piece of code, and then writing the code to pass the test [58]. This generally leads to an organic development of the design, and ensures that unit tests cover all critical aspects of the code, which simplifies debugging.

The graph structure is represented in an *adjacency list* structure. For each node $v_i$, we store its list of neighbours in a dictionary entry using the identifier for $v_i$ as key. This approach is preferred over an adjacency matrix structure, since the Wikipedia graph is very sparse. For each node, there are an average of 34 neighbours. Wikipedia has roughly $3 \cdot 10^6$ pages (i.e. nodes in the graph, or $|G|$). In an adjacency matrix representation, an $N \times N$ matrix represents a graph of degree $N$, with the element at row $i$ and column

**Algorithm 3:** Pseudo code to measure strength of association between $v_i$ and $v_j$ using AA. $\mathbf{A}_i$ and $\mathbf{A}_j$ are two zero-initialised $N$-dimensional vectors ($N = |G|$)

**Require:** $G, L_{p,\max}, K_{\text{decay}}, T$
  **function** SPREAD_AA($v_i, v_j$)
    Set $a_i \in \mathbf{A}$ to $K_{\text{init}}$ ▷ $K_{\text{init}} = 1.0$
    SPREAD_UNIDIR($v_i, \mathbf{A_i}, \emptyset$) ▷ $\mathbf{A}_i$ updated in-place
    Set $a_j \in \mathbf{A_j}$ to $K_{\text{init}}$
    SPREAD_UNIDIR($v_j, \mathbf{A_j}, \emptyset$)
    Compute similarity using either AA-cos or AA-wlm on $\mathbf{A}_i$ and $\mathbf{A}_j$
  **end function**

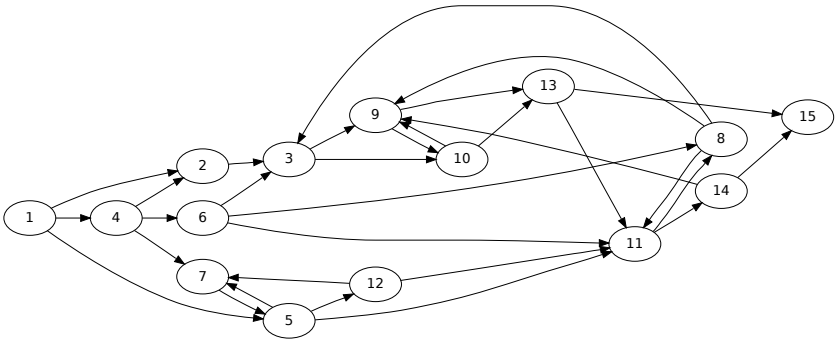



**Figure 3.3:** The simple reduced test graph that was used for unit testing all graph-related functions. All algorithms could be verified to work correctly by hand on this very simple graph, before using it on the full Wikipedia hyperlink graph, which is four orders of magnitude larger and would be nearly impossible to verify directly.

$j$ indicating whether an edge exists between node $v_i$ and node $v_j$. This would lead to a very sparse graph. An adjacency list representation, on the other hand, leads to a very compact representation.

An adjacency list structure also reduces the time required to find a node's neighbours, since looking up a node $v_i$'s neighbours is linear in the number of its neighbours, whereas using an $N \times N$ adjacency matrix structure (with $N$ the size of the graph, i.e. $|G|$), it is linear in the order of the entire graph $|G|$.

Each article in Wikipedia can be identified by its unique article identifier. The graph is stored on disk in the format '$v_i : v_1, v_2, ..., v_k$' with $v_1$ through $v_k$ the article identifiers of node $v_i$'s neighbours $\mathcal{N}(v_i)$. This structure is loaded into memory for performance-increase as a dictionary structure in Python. Since Python is very wasteful in its memory usage for storing integers[4], we made use of the `array` Python-module to store these article identifiers as unsigned integers. This resulted in a data structure of roughly 500 MB.

Critical components in the code, such as the depth-first search algorithm used by `spread_unidir()`, were tested using a small test graph created specifically for testing (see Figure 3.3). We used this approach since the entire Wikipedia graph structure is very large and takes some time to load into memory, which makes it unsuited for frequent

[4]Python can use up to 32 **bytes** in certain cases to store one 32 **bit** integer, according to some sources [59].

cycles of *test, refactor, test*, as called for by TDD. Furthermore, we cannot easily use some subgraph of the full graph structure from the adjacency list representation, since its nodes might contain links to unresolved nodes not contained in the subgraph – and verifying results would be very difficult. Using a reduced test graph enabled us to easily verify the paths between nodes by hand, before adding these expected results as unit tests.

## 3.8 Chapter summary

In this chapter the general Spreading Activation (SA) framework was introduced, discussed and adapted for computing conceptual similarity over the Wikipedia hyperlink structure. Wikipedia's hyperlink structure was introduced and the problems related to modelling it as an associative network were addressed, namely:

1. Constructing an associative network from the noisy Wikipedia hyperlink structure. This required strategies for inferring the link weights connecting concept nodes, since these weights do not exist in the hyperlink structure.

2. Adapting the spreading activation technique to function over the Wikipedia associative network as constructed in (1).

3. Translating the node activations which result from the SA process into scores of relatedness between the initial concept nodes.

Spreading activation was extended, and three different spreading strategies were developed. Target Activation Approach spreading strategy (TAA) measures relatedness between two concepts by supplying one concept with an initial amount of energy and measuring the ratio of energy that reaches the target node after several spreading iterations.

The Agglomerative Approach spreading strategy (AA) spreads energy outwards from both nodes and measures similarity as the amount of overlap between the sets of activated nodes that are produced in this way.

This model for spreading activation relies on several parameters which affect its performance. We will evaluate this model and optimise these parameters by conducting inter-concept similarity experiments in Chapter 7.

In the next chapter, we discuss computing similarity between two documents.

# Chapter 4

# Inter-document similarity: A unified approach

In this chapter we focus on computing document similarity. We address the second hypothesis, namely that a combination of a keyword-based method and a knowledge-based method captures document similarity ratings better than either approach in isolation.

We start by motivating our approach. Document preprocessing techniques (or *term space modification*) are discussed as an attempt to introduce additional context into the term space of simple similarity measures, such as the cosine Vector Space Model (VSM) and Normalised Compression Distance (NCD) metric. This idea will be tested by conducting experiments in Section 7.2.

Two new metrics for computing inter-document similarity scores, using Wikipedia only, are presented. These techniques will be evaluated and the results will be discussed in Section 7.3.

Finally, a combined similarity measure is proposed using one of our Wikipedia-based methods to augment the cosine VSM, in order to arrive at a conclusion about our initial hypothesis. This method will be evaluated in Section 7.4.

## 4.1 Introduction

Traditionally, many textual similarity algorithms operate from the so-called *bag-of-words (BOW)* model where a document is modelled as an unordered set of words or concepts (possibly duplicate, therefore 'bag'). In this model the two phrases *'Man accused of murder released on bail'* and *'Murder man accused of bail on released'* would be ranked as equivalent.

Clearly this is not how humans determine similarity. Even for short sentences most 'meaning' to humans is lost. For instance, the BOW representation of the last sentence is: *'clearly determine how humans is not similarity this'*. However, this bag-of-words model has proved to be very useful and to provide very robust scores [43].

These measures extract 'content-words' and discard 'stop words' (words like 'a', 'an', 'the', etc., which appear often but do not serve very well to distinguish text). Typically algorithms in this class order texts as vectors of concepts, ranked by weights representing their prominence in the text (see Section 2.4.1). A vector similarity is then performed between two texts to produce a similarity score.

Approaches to inter-document similarity which do not discard semantics, but try to use the semantic structure of text to infer meaning and determine relatedness, include Markov Random Fields (MRF) (e.g. the full-dependence MRF model [60]), Textual Entailment (TE) (using tree edit-distances to transform one sentence's parse tree into another [61]) and paraphrase detection (e.g. *'X is the writer of Y'* ⇔ *'X wrote Y'* ⇔ *'Y is written by X'*, see [62] and [63]). In a setting such as Question-Answering, where the system is presented with a natural language question and must produce a natural language output as answer, and systems where in-depth understanding of the text needs to be performed, these involved approaches can be justified. However, for automatic clustering of documents and for recommending related content, the computational complexity of these approaches cannot be justified.

Instead, we take a different approach based on our hypothesis that a combined keyword-based and knowledge-based approach leads to more accurate document similarity ratings. We analyse the potential of combining two very different approaches for computing textual similarity. Simple vector-based methods measure similarity by the amount of keyword overlap between two documents. Approaches utilising man-made knowledge repositories such as Wikipedia on the other hand, try to extract deeper relationships at a conceptual level. This chapter focuses on analysing the effects that combining these two different approaches can have on accuracy.

We start by evaluating the effects of different preprocessing techniques on the accuracy of the vector space model and the Normalised Compression Distance. We then introduce and evaluate two distinct approaches for measuring inter-document similarity. Finally, we combine the two approaches into one similarity metric.

## 4.2 Preprocessing document text for improved document similarity performance

In Information Retrieval (IR), **query expansion** is the process of augmenting a (short) query with additional terms for increased retrieval performance [43]. The main idea is that a document can only be matched to a query if it contains some form of feature overlap as measured by some similarity function (the retrieval algorithm). Therefore short queries are extended by adding additional keywords that are strongly associated with the query terms.

In the standard Vector Space Model (VSM), two words are only considered a match if they are indeed the same. For instance, the terms 'gather' and 'gathered' would be considered unrelated. One of the most prevalent preprocessing steps used in the VSM is the process of removing 'stop words', and reducing words to their base forms (lemmas) prior to indexing and computing relatedness [43]. Stop words include high frequency words such as 'a', 'an', 'the', etc. which generally contribute very little to the content of a piece of text.

Stop word pruning and word stemming are the two most commonly used preprocessing steps. However, several other lexico-syntactic[1] preprocessing steps will be identified and introduced in the next section.

[1]Operating directly at the lexical (word) level as well as on the syntactic level, such as identifying the part-of-speech of a word.

Since we were unable to find any works in the literature where the actual effects of these preprocessing steps on the similarity metric's performance have been evaluated, it was decided to explore the effects of these processes on the performance of two easily computable measures of similarity, namely cosine similarity and the Normalised Compression Distance. The main idea is that preprocessing texts using simple syntactic manipulations might introduce some stronger sense of context into the term space[2] itself, which could lead to improved results using simpler similarity metrics. In the following sections, the preprocessing steps that were evaluated are discussed. These methods will be evaluated in Section 7.2.

## Natural Language Toolkit (NLTK)

NLTK [64, 65] is an open source, Python-based natural language processing toolkit. It supports rapid prototyping (due to its implementation in Python) and provides access to several natural language processing tasks, including word tokenisation, part-of-speech (POS)-tagging, word stemming, etc.

## Shallow lexico-syntactic preprocessing steps

Five main lexico-syntactic preprocessing manipulations were identified for evaluation, namely part-of-speech (POS) tagging, stop word removal, word stemming, converting text into bigrams and finally, converting text into trigrams. These preprocessing steps were combined to form 'pipelines' of the various combinations in which they can be applied. The pipeline module is discussed in the next section.

**Part-of-speech (POS) tagging** involves parsing the words of a sentence into their constituent classes, such as nouns, verbs, adjectives, etc. Due to polysemy in natural language (see Section 2.5), this is more complex than simply keeping a list of words and their POS tags, since two identical words can be classified into different parts of speech depending on their context.

Current approaches to POS-tagging use a combination of machine learning and rule-based learning to accurately parse text. The NLTK POS-tagger we used accurately identifies 85% of the manually tagged sentences of the Brown Corpus [65].

**Stop word removal**[3] involves removing all the high-frequency, low-content words from text. These include words like 'a', 'an', 'the', 'him', 'myself', etc.

**Word stemming** involves reducing a word to its base form. For instance, 'listening', 'listens' and 'listened' would all be reduced to 'listen', and 'distributed', 'distribution' and 'distributes' to 'distribut'. We used the standard de facto Porter stemmer implementation included in NLTK for this task.

**Bigrams** were chosen because we wanted to investigate the idea of introducing 'context' directly into the term space by grouping adjacent terms. To better illustrate this, consider the simple sentences `'She likes dancing but hates travelling'` versus `'She likes travelling but hates dancing'`. Simple VSM methods would rate these sentences equally since they contain the same words. However, by converting this sentence into

[2] In the VSM 'term space' refers to the vector space in which each unique term in $\mathcal{D}$ has its own dimension. In general, 'term space' is used here to refer to the way in which a specific method represents the text of a document.

[3] In general, text from which stop words have been removed is simply referred to as 'stopped' text.

**Table 4.1:** An example to illustrate the idea that bigrams can incorporate a sense of word order into simple similarity metrics. $M$ denotes the number of matched tokens. In the top example (unigrams), keep in mind that tokens are matched regardless of word order. Words are simply presented in their original order to show the ordering of the initial sentences, however matches are determined based on the presence or absence of the token in the entire text (sentence).

| | | | | | | | $M$ |
|---|---|---|---|---|---|---|---|
| Sentence one | She | likes | dancing | but | hates | travelling | |
| Sentence two | She | likes | travelling | but | hates | dancing | |
| Matching unigrams | 1 | 1 | 1 | 1 | 1 | 1 | 6/6 |
| Sentence one | She_likes | likes_dancing | dancing_but | but_hates | hates_traveling | | |
| Sentence two | She_likes | likes_travelling | travelling_but | but_hates | hates_dancing | | |
| Matching bigrams | 1 | 0 | 0 | 1 | 0 | | 2/5 |

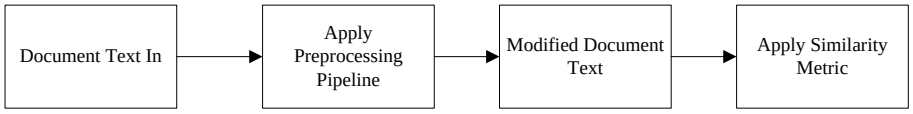



**Figure 4.1:** Modifying the term space by substituting original terms for processed (modified) terms (TSS).

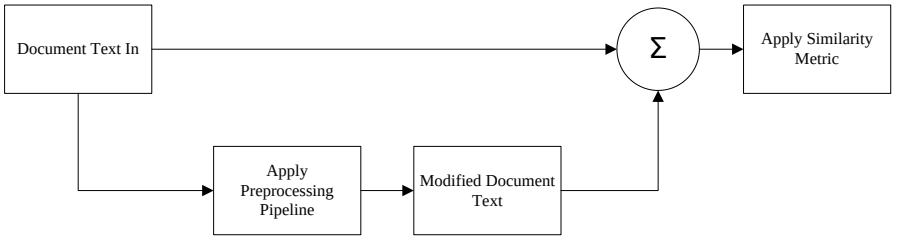



**Figure 4.2:** Modifying the term space by augmenting the original terms with the processed (modified) terms (TSA).

its constituent bigrams, namely `'She_likes likes_dancing dancing_but but_hates hates_travelling''` and `'She_likes likes_travelling travelling_but but_hates hates_dancing'` we see that matching tokens are reduced to only `'She_likes'` and `'but_hates'` (see Table 4.1). The same applies for trigrams. We wanted to evaluate if this could lead to any improvements in performance of the similarity ratings of two 'simple' approaches (i.e. approaches requiring no extensive semantic post-processing), namely the cosine VSM with term-frequency inverse document-frequency (tf-idf) and the NCD.

### 4.2.1 Approaches to term space modification

Two distinct ways can be identified in which preprocessing can be applied to the text of a document. Firstly, the initial text can be **substituted** by the modified text (see Figure 4.1 for a visual representation) or secondly, the original text can be **augmented** by the modified text (see Figure 4.2).

For example, using Term Space Substitution (TSS), if we apply the NLTK Porter-stemmer to the sentence `'18 rescued from illegal initiation schools'`, it pro-

duces `'18 rescu from illeg initi school'`. Applying stemming using the Term Space Augmentation (TSA) approach, produces `'18 rescued from illegal initiation schools 18 rescu from illeg initi school'`. The idea is that introducing more specific information into the term space might improve the performance of simpler similarity metrics.

From here on, we will refer to the first approach as **Term Space Substitution (TSS)** and the second as **Term Space Augmentation (TSA)**. The effects of the different preprocessing steps that were identified will be evaluated in Section 7.2 for both the TSS and TSA model discussed above.

## 4.3 Wikipedia-based document similarity

One of the problems with simple VSM methods such as the cosine similarity metric (as introduced in Section 2.4.1) and NCD methods, is that they discard potential matches between words which might be very related semantically. That is, if two terms do not match exactly, they are considered distinct. Stemming is an attempt to reduce words to their base form in order to match morphologically related words (like 'distributes','distributed' and 'distribution'). However, in general, VSM methods suffer from a sparse term space problem where potentially very related concepts (e.g. synonyms such as 'kid' and 'child') are considered distinct and unrelated.

In order to overcome this problem, we make use of lexical resources (as discussed in Section 2.5) to identify these semantically related concepts. Wikipedia is a rich source of knowledge, and following our investigations into inter-concept similarities in Chapter 3, we investigate the viability of representing documents as concept vectors defined by Wikipedia articles, and then to use the spreading techniques developed in Chapter 3 to compute an overall inter-document similarity measure.

This process involves first identifying the salient representative articles that adequately represent each document (the *document concept vectors*). This process is known as *wikification* and is discussed next. Secondly, it involves computing the similarities between the identified articles and combining these similarities into a score indicating the relatedness between the two documents. For this we propose three new approaches, namely two methods using only Wikipedia (MaxSim and WikiSpread), and one method which combines the cosine VSM with a Wikipedia-based method.

These methods are tested and the results are discussed in Chapter 7.

### 4.3.1 Identifying salient concepts

Identifying key representative Wikipedia topics in a document is a process known as *wikification*. The first approach to this problem was introduced in the Wikify! system [66]. This system has a link detection and a disambiguation phase. *Link detection* is based on the probability of a phrase being used as a link (the **link probability**), defined as the number of articles that use the phrase as an anchor, divided by the number of articles that contain the phrase at all. To *disambiguate* phrases (i.e. if a document contains the phrase 'plane' in the fixed-wing aircraft sense, to discard links to Wikipedia articles about the mathematical sense) the Wikify! system used a very involved preprocessing stage whereby the surrounding context of phrases were used to disambiguate senses.

> Nigerian **President Olusegun Obasanjo** said he will weep if a single mother **sentenced to death** by **stoning** for having a child out of wedlock is killed, but added he has faith the **court system** will overturn her sentence. Obasanjo's comments late Saturday appeared to confirm he would not intervene directly in the case, despite an international outcry.

**Figure 4.3:** A sample document from the Lee dataset with the extracted concepts highlighted.

**Table 4.2:** An example of the topics extracted from the document in the Lee dataset shown in Figure 4.3 with the associated confidence scores.

| Article ID | Article Title | Confidence |
|---|---|---|
| 382736 | Olusegun Obasanjo | 0.94 |
| 19186813 | Stoning | 0.87 |
| 31737 | Supreme Court of the United States | 0.65 |
| 59564 | Judiciary | 0.58 |
| 21383 | Nigeria | 0.57 |
| 870351 | Sentence (law) | 0.55 |
| 5902 | Capital punishment | 0.55 |

Milne [67] presents a machine-learning approach which is trained using a decision tree classifier on features such as link probability as discussed above, generality (the minimum depth of the concept in the Wikipedia category tree), etc. They achieve superior performance compared to other methods, with reported recall and precision values of almost 75% [67] (see Section 7.7 for a discussion of recall and precision).

We used an implementation of their approach to construct the document-concept vectors from document text. The decision tree algorithm identifies topics with a certain level of classifier confidence, and initial experimentation revealed that classifier confidence levels of 50% produces around 5-8 topics per document which fairly accurately represents the contents of the document, based on manual inspection. For example, a sample document is shown in Figure 4.3 with the extracted concept vector of Wikipedia articles shown in Table 4.2.

The result of the topic extraction process for document $d_k$ is the document concept vector $\mathcal{C}_k$. $\mathcal{C}_k$ consists of a set of node-value-pairs of the form $\mathcal{C}_k = \{(v_{k1}, p_{k1}), (v_{k2}, p_{k2}), \ldots\}$ where $v_{kx}$ is some Wikipedia article identifier and $p_{kx}$ is the respective classifier confidence level associated with that particular topic. The document concept vector $\mathcal{C}_k$ can be seen as representing document $d_k$ somewhere in the $N$-dimensional Wikipedia concept-space (for $N = |G|$).

### 4.3.2 Computing inter-document similarity using only Wikipedia

For two document concept vectors $\mathcal{C}_i$ and $\mathcal{C}_j$, an algorithm is required that accepts document concept vectors as defined above, and produces a similarity rating. For this we present two novel algorithms for evaluation, namely MaxSim and WikiSpread.

#### 4.3.2.1 The MaxSim inter-document similarity metric

The first inter-document similarity metric we define (see Algorithm 4) is based on the idea of measuring document similarity by pairing up each concept in one document with its most similar concept in the other document, and averaging those similarities to produce an inter-document similarity.

Given two documents and their concept vectors, $\mathcal{C}_1$ and $\mathcal{C}_2$, let $v_i$ be some topic (article identifier in Wikipedia graph) in document one's concept vector $\mathcal{C}_1$, and $v_j$ some topic in document two's concept vector $\mathcal{C}_2$. Furthermore, let $\text{sim}(v_i, v_j)$ be the similarity between $v_i$ and $v_j$ using one of the approaches outlined in the previous chapter.

We define $s_{ik} = \text{sim}(v_i, v_k)$ to be the maximum similarity-value returned for $v_i \in \mathcal{C}_1$ compared to some node $v_k \in \mathcal{C}_2$, such that $v_k = \underset{v_j}{\text{argmax}}\ \text{sim}(v_i, v_j)$. In other words, topic $v_k \in \mathcal{C}_2$ is the topic in document two that is most similar to $v_i$ in document one, and $s_{ik}$ is the associated real-valued similarity value.

We use the probability $p_i$ returned by the topic classifier to weight the contribution a concept can make to the final score. We also further weight the contribution of a concept by its Inverse Link Frequency weighting scheme (ILF) score as defined in Equation 3.4.2 so that the contribution of a concept that is very frequently linked to is decreased. These scores are combined using the following equation:

$$\text{sim}(\mathcal{C}_1, \mathcal{C}_2)_{\mathcal{C}_1} = \frac{\sum_{v_i \in \mathcal{C}_1} s_{ik} p_i \text{ILF}(v_k)}{\sum_{v_i \in \mathcal{C}_1} \text{ILF}(v_k)}, \tag{4.3.1}$$

Note that Equation 4.3.1 is a directional measure of similarity. To obtain the bidirectional relatedness between two texts, $d_1$ and $d_2$, we compute the directional similarity from both sides and compute the average:

$$\text{sim}(\mathcal{C}_1, \mathcal{C}_2)_{\text{undirected}} = \frac{\text{sim}(\mathcal{C}_1, \mathcal{C}_2)_{\mathcal{C}_1} + \text{sim}(\mathcal{C}_2, \mathcal{C}_1)_{\mathcal{C}_2}}{2}. \tag{4.3.2}$$

Equation 4.3.2 and Equation 4.3.1 represent the definition of a new document similarity metric we define, using Wikipedia articles as concepts, called the **MaxSim method**.

#### 4.3.2.2 The WikiSpread inter-document similarity metric

The second inter-document similarity metric we define (see Algorithm 5) builds on the inter-concept spreading activation work introduced in the previous chapter. We consider a document concept vector as a cluster of concepts, and build a single *document activation vector* by spreading outwards from each concept in the vector using spreading activation as defined by the *spread_ unidir()* algorithm introduced in the previous chapter (see Algorithm 1).

This process results in a document activation vector – i.e. a vector of article identifiers and their respective activations – for each document. Finally, we compute the similarity between two such activation vectors in the same way in which we did for individual concepts, by using one of the two agglomerative approaches introduced in the previous chapter (AA-wlm or AA-cos).

**Algorithm 4:** Pseudocode for the MaxSim algorithm for computing inter-document similarity.

```
Require: ILF lookup function
  function MaxSim(C_1, C_2)
      num=0
      den=0
      for (v_i, p_i) ∈ C_1 do                ▷ Topics and classifier confidence in document one
          s_ik = 0                            ▷ Topic most related to v_i in C_2
          for v_j ∈ C_2 do                    ▷ Find most related topic
              s_ij = sim(v_i, v_j)
              if s_ij > s_ik then
                  v_k = v_j
                  s_ik = s_ij
              end if
          end for
          num += s_ik p_i ILF(v_k)
          den += ILF(v_k)
      end for
      return num / den
  end function
```

**Algorithm 5:** Pseudo code for the WikiSpread algorithm for computing inter-document similarity. $K_{\text{init}} = 1.0$. $\mathbf{A}_1$ and $\mathbf{A}_2$ are two $N$-dimensional zero vectors ($N = |G|$).

```
  function WikiSpread(C_1, C_2)
      for (v_i, p_i) ∈ C_1 do                ▷ Document 1
          Set a_1i ∈ A_1 to K_init · p_i      ▷ Update a_1i ∝ p_i
          spread_unidir(v_i, A_1, ∅)
      end for
      for (v_j, p_j) ∈ C_2 do                ▷ Document 2
          Set a_2j ∈ A_2 to K_init · p_j
          spread_unidir(v_j, A_2, ∅)
      end for
      Compute similarity using AA-cos or AA-wlm using A_1 and A_2
  end function
```

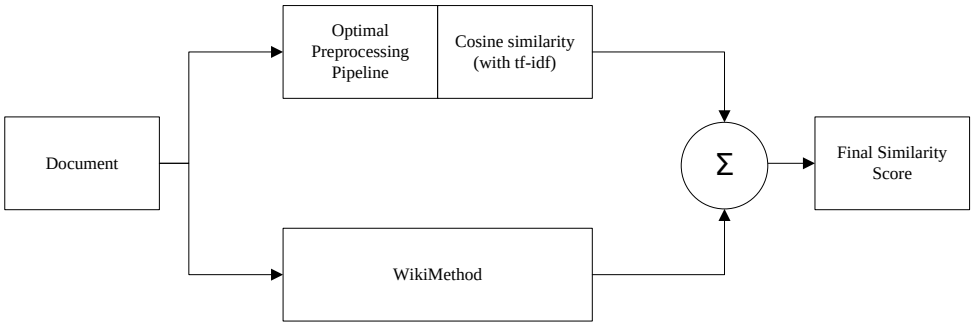



**Figure 4.4:** Diagram to show the final combined semantic similarity algorithm as a linear combination of an augmented cosine similarity metric and our Wikipedia-based measure.

### 4.3.3 Combined cosine VSM and Wikipedia-based inter-document similarity metric

The final document similarity measure we propose augments the cosine VSM method with one of our Wikipedia-based methods introduced above.

The cosine approach measures direct overlap between keywords. The Wikipedia-based approach aims to capture conceptual relationships between concepts, as defined in Wikipedia. We therefore implemented a final combined similarity metric using a linear combination of the best cosine model and the best-performing Wikipedia-based method, namely

$$\text{sim}_{\text{combined}}(d_1, d_2) = \lambda \text{sim}_{\cos}(d_1, d_2) + (1 - \lambda)\text{sim}_{\text{wiki}}(d_1, d_2), \tag{4.3.3}$$

with $\lambda$ a mixture weight between 0.0 and 1.0 denoting the contribution from the two different methods (see Figure 4.4).

## 4.4 Chapter summary

In this chapter, motivation was presented for using a combination of simple lexical-level feature overlap similarity metric with a conceptual Wikipedia-based metric as a combined approach for measuring document similarity.

Document preprocessing (term space modification) was introduced and classified as either pertaining to Term Space Substitution (TSS) or Term Space Augmentation (TSA). TSS is the standard approach of replacing a document's text with the preprocessed text prior to computing similarity. TSA appends the processed text to the original document, since we hypothesise that this process introduces more specific context into the document text.

Two new Wikipedia-based document similarity metrics, called the MaxSim method and the WikiSpread method, were also developed. Both rely on a process of *wikification* to extract the *document concept vectors* which best describe the document text. Document concept vectors consist of Wikipedia article identifiers and their associated weights describing how strongly the article is seen to relate to the text.

Finally, we described how we will combine the Wikipedia-based method with the simpler cosine VSM methods in order to test our hypothesis that a combined metric leads to higher accuracy.

These approaches will be evaluated by conducting several experiments in Chapter 7.

# Chapter 5

# Content-based recommendations of news articles as a use case for document similarity metrics

As an application and use case for document similarity metrics, a remote news article recommender system is designed and implemented for an online news provider in South Africa. Specifically, Health24.com[1] desired a recommender system which could help journalists submitting an article into the content management system (CMS) for publication, to identify other possibly related articles. Journalists include these related articles as recommended articles in a *Read more* section.

We wanted to investigate the extent to which purely content-based recommendations, using three widely-used information retrieval models, could be utilised to provide recommendations in this context. In this section we discuss our design choices and some of the conceptual design motivations and decisions that were faced. We also touch on some of the more practical implementation details. The performance and satisfaction results obtained from using this system in a production setting, as rated by the journalists using the system, are presented in Section 7.7.

## 5.1 Conceptual design overview

### 5.1.1 General design considerations

Health24.com generates approximately 15-17 new articles per day. They have a proprietary database (DB) back-end implemented in Microsoft SQL Server and ASP .NET.

We decided to implement the recommender as a stand-alone Web Service accessible using a protocol such as XML Remote Procedure Call (XML-RPC) or Simple Object Access Protocol (SOAP) (see Figure 5.1). The main motivations for this decision were threefold:

1. Firstly, implementing the system on Health24.com's servers could lead to possible data corruption.

[1] http://www.health24.com/

2. Secondly, making the service separate from Health24.com would allow other users access to the same service.

3. Thirdly, most design and implementation would be done by the author, and being geographically separated from the Health24.com headquarters led us to implement this service on our own Media Lab[2] server.

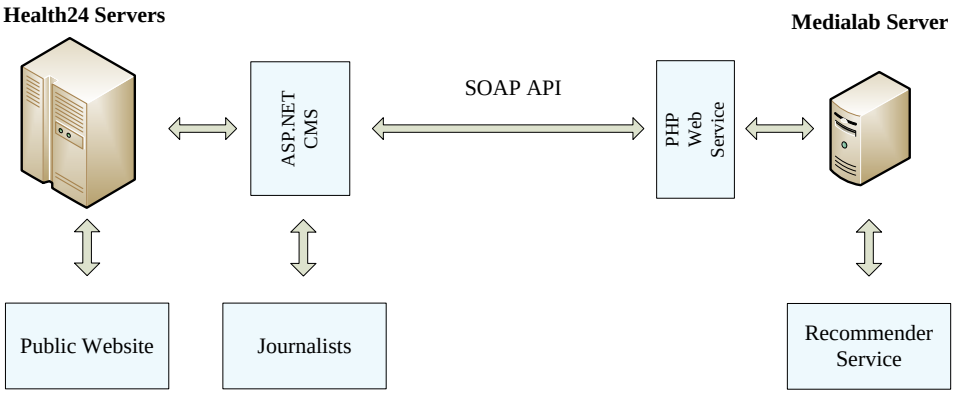



**Figure 5.1:** Main conceptual overview of the recommender service.

### 5.1.2 The recommender as an information retrieval system

Recommender systems are considered a specific form of information filtering that attempts to present a user with additional material (news articles in our case) by taking into account certain features and comparing these to the active user's history and profile [68]. These features can range from a user's past interests (based on the content of the material, or the so called **content-based (CB)** approach) to the material other users who display similar taste to the active user (the user receiving recommendations) have liked, i.e. based on the active user's social environment (the so-called **collaborative-filtering (CF)** approach). User profiles are constructed either via **implicit** means such as collecting click-through rates or times spent per page, etc.; or via **explicit** means such as questionnaires or letting users explicitly rate items with e.g. "like" or "dislike" options, etc.

Both types of recommenders aim to improve recommendations over time based on feedback from users. CF systems use user feedback as a larger collection of ratings to match up user profiles and use other users' ratings to make recommendations for the active user. CB systems build a profile of the active user and use this profile to compare against available material, based on their content, to make predictions from.

In designing the news article recommender, however, we were simply interested in testing the viability of using purely content-based similarity metrics to provide recommendations for related articles to journalists. We designed the recommender as an information retrieval system where the current active article viewed by the user is used as a *query* against the database of indexed articles to produce a ranked list of conceptually related articles as output (see Figure 5.2).

In this model, the efficiency of recommendations is based solely on the *similarity measure* that is employed, and this provided a useful platform for comparing the performance

[2]http://www.ml.sun.ac.za/

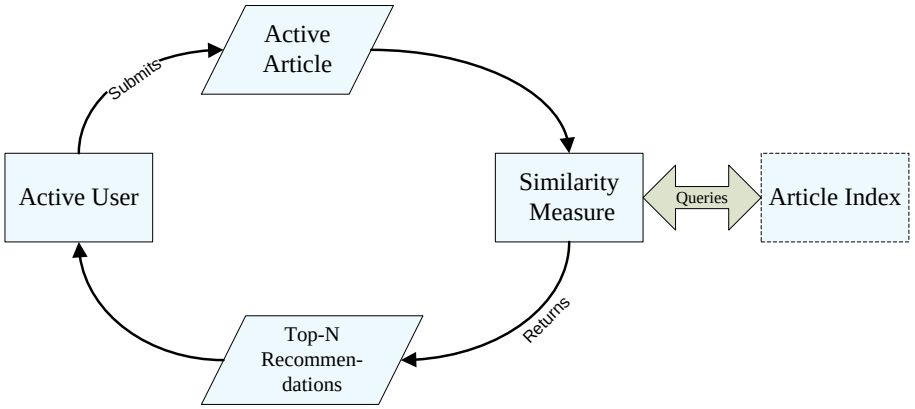



**Figure 5.2:** Conceptual overview of the recommender service.

of various standard document similarity metrics used in Information Retrieval (IR) in a document recommendation context. More specifically, we evaluated the efficiency (measured as mean average uninterpolated precision (MAP) scores, as rated by journalists using the service) produced by cosine similarity with tf-idf term weighting (see Section 2.4.1), the Okapi BM25 probabilistic similarity measure (see Section 2.4.2) and unigram query-likelihood statistical language models (see Section 2.4.3). The results of this evaluation are presented in Section 7.7.

## 5.2 Web service

A PHP Web Service was implemented on our Media Lab[3] server as front-end to the recommender. This was not considered to cause any noticeable delays in the recommender process, since average network access times between Health24.com and the Media Lab server was less than 64ms with ample bandwidth for real-time SOAP calls. The SOAP interface was implemented using the PHP nuSOAP libraries [69]. SOAP was chosen as Web Service interface over alternative options because it required the least amount of change in Health24's current CMS setup, since they already had other services that were using SOAP.

In order to make the Web Service accessible to as many different platforms as possible, a Web Services Description Language (WSDL) [70] interface was defined which is a machine-processable definition of the services offered by the Web Service.

Service requesters *bind* to the service using the WSDL interface definition. This is used to create a client-side *proxy* through which they can access the features of the Web Service.

Clients communicate with the Web Service following the definition of the service as defined in its WSDL file, over standard HTTP (with the possibility of SSL for secure connections) using SOAP-messages serialised as eXtensible Markup Language (XML) strings.

The Web Service acts as a server-side proxy between the recommender and the client. Clients bind to the service and can then request a specific feature (as defined in the API, see Section 5.3) by passing the request as a SOAP message. The Web Service performs

[3]http://www.ml.sun.ac.za/

the requested feature server-side and then returns the result to the client, wrapped as a SOAP object.

## 5.3 Development of the API

### 5.3.1 Use case analysis

In the following discussion we will refer to the client's storage mechanism as her database (DB) or content management system (CMS) and to our storage mechanism as our *index*. A general UML use case analysis of the system produces the following scenarios (see Figure 5.3):

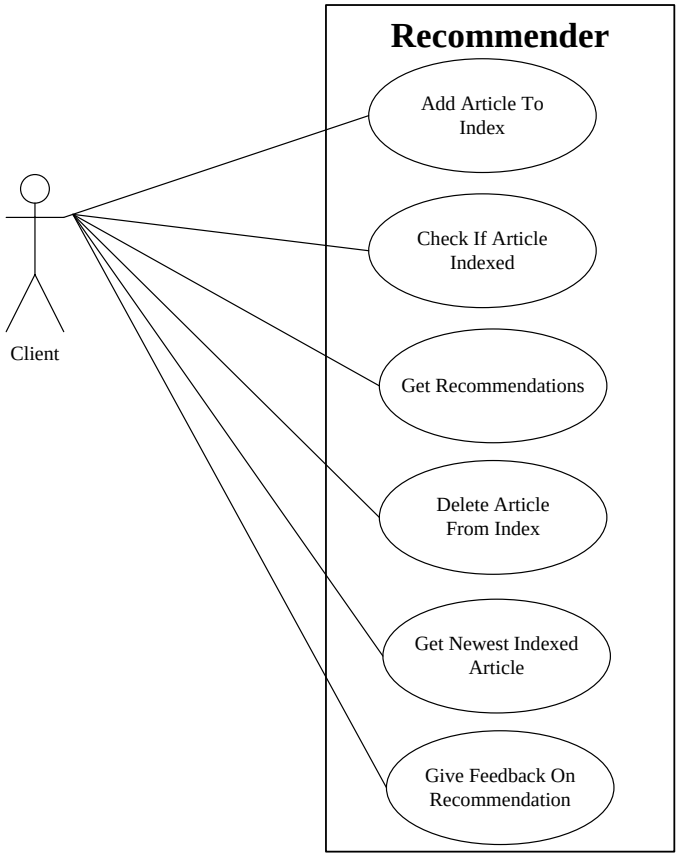



**Figure 5.3:** Use case diagram for the news article recommender system.

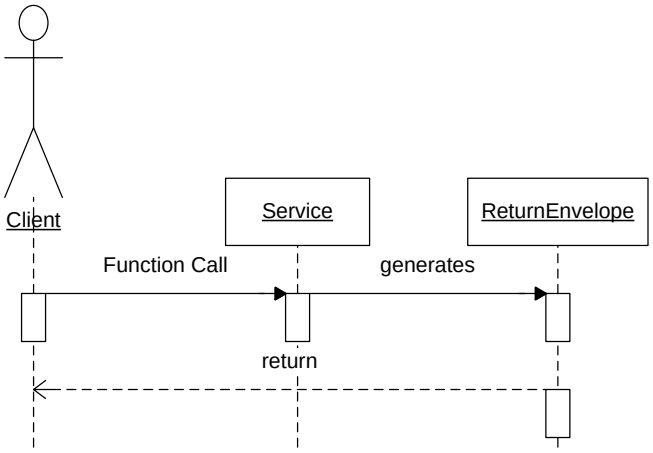



**Figure 5.4:** A sequence diagram illustrating the general flow of control during a typical function call.

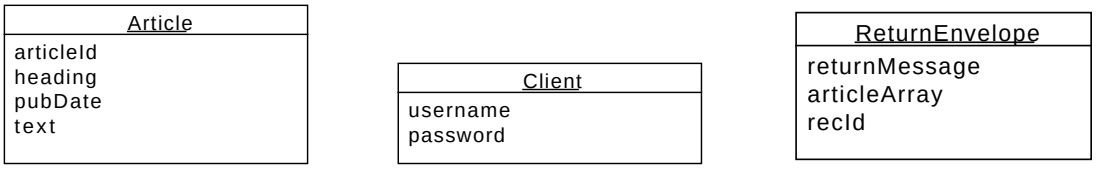
A client wants to:

- add a new article to our index;
- determine if we have a certain article in our index;
- get the top $n$ recommendations for a given article in our index;
- get the top $n$ recommendations for a given text string;
- delete an article from our index;
- obtain the newest indexed article in our index;
- give us feedback regarding a recommendation.

Figure 5.4 illustrates the basic flow of control during a typical function call. The client initiates a function call to the system. The system then processes the function call and wraps the response in a ReturnEnvelope (see Figure 5.5(c)) object and returns it to the client.

#### 5.3.1.1 Adding a new article

As the client's collection of articles grows (i.e. as journalists are writing and adding new articles to their CMS) she might want to add those to our (mirrored) collection of her articles to allow our recommender system to index the new article and return it in the case of a relevant recommendation in future.




(a) The Article object. (b) The Client object. (c) The ReturnEnvelope object.

**Figure 5.5:** UML diagrams of the Article, Client and ReturnEnvelope objects.

Clients identify themselves with a *user name* and *password* (see Figure 5.5(b)), and articles consist of a unique article id (*articleId*), a *heading*, a date on which it was originally created, *pubDate*, and the body text of the article, *text* (see Figure 5.5(a)).

#### 5.3.1.2 Determining if an article exists

Each article is required to possess a unique identifying id, *articleId* (see Figure 5.5(a)). If a user wants to see if we have a certain article in our index, she can make an *IDExists()* function call, passing the specific article's *articleId* as parameter.

The system then checks its DB and returns the result in *returnMessage* of ReturnEnvelope (see Figure 5.5(c)).

#### 5.3.1.3 Returning recommendations for articles or text

To find related articles, the user makes a *GetRecommendations()* function call. Recommendations can be generated for articles in the index by passing the article's *articleId* as parameter. Recommendations can also be made for a string of text by passing the string as parameter. This function is useful if the client is typing a new article and, before submitting the article to the CMS or to our index, wants to see if there are any previous articles related to her work-in-progress.

If *returnText* is set to true, *GetRecommendations()* returns the first paragraph of text of the recommended articles. This can be used to provide context for the recommendations (as news headings are not always very indicative of the article's content).

Each recommendation our system makes is logged and assigned a unique recommendation id, *recId*, which is passed back to the client. This can be used for feedback later to refer to a specific set of recommendations (see 5.3.1.6).

#### 5.3.1.4 Deleting an article

In the event where a client needs to remove an article from the index, this function can be called with the article's *articleId*.

#### 5.3.1.5 Finding the most recently indexed article

It is very important to keep the article index of the recommender synchronised with the articles in the client's CMS. The recommender cannot make recommendations for articles that it does not have in its index.

When the client wants to synchronise the index with her CMS, this function provides the newest article that we have indexed. The client can then update our index with all articles in her CMS that were published after our newest article.

#### 5.3.1.6 Returning feedback regarding a recommendation

In order to allow us to measure the accuracy of our system and to allow for the possibility that user feedback might be used to improve the recommendations, the client can return a rating of the recommendations which we can then use to train the system to improve its perceived accuracy.

Each recommendation issued by the system is assigned a unique recommender identification code. A user can 'rate' the recommendations and then return the list of ratings along with the recommender id code which the system then uses to calculate its accuracy or to train itself to give improved recommendations.

## 5.4 Recommender back-end

In order to provide real-time recommendations in a production environment over a database containing more than 40,000 articles and growing each day, a very efficient approach was needed.

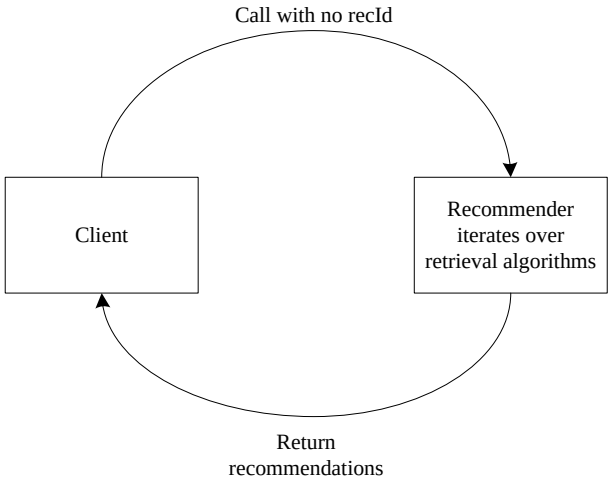
A good compromise was found between extensibility and efficiency in the open-source *Lemur Toolkit* [71]. The Lemur Toolkit was designed to facilitate research in especially language modelling and information retrieval, but has grown into a robust and production-level package which supports indexing of large-scale textual databases. Most of the toolkit is implemented in C and C++ and it runs very efficiently due to its use of inverted indexes to speed up lookup.

The Lemur Toolkit is aimed towards researchers and supports various document comparison methods. In order to aid our research into similarity measures we tested the three most prominent algorithms, namely the tf-idf cosine similarity (see Section 2.4.1), the probabilistic BM25 Okapi method (see Section 2.4.2) and statistical query-likelihood language models (see Section 2.4.3).

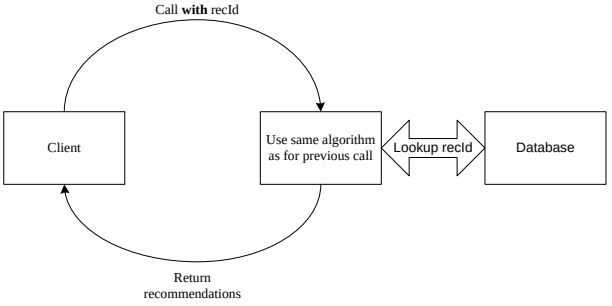
The Okapi BM25 retrieval function has two main tuning parameters $K_1$ and $b$ (see Section 2.4.2). We used $K_1 = 1.2$ and $b = 0.75$. This configuration has been shown to perform best empirically [29]. For the query-likelihood language models approach we used Dirichlet priors as smoothing method with Kullback-Leibler divergence for measuring similarity which has also been shown to be the best over several TREC[4] test runs [29].

Using Lemur as the document indexing back-end, we implemented an *algorithm switching* method (see Figure 5.4) which cycles between the above three algorithms on a call-by-call basis. Session persistence was introduced whereby, if a user is busy editing a document and requests recommendations several times before publication of that article we would use the same algorithm for all those partial requests (kept track of via the recommender session id, *recId*, introduced in Section 5.3.1.3). This was done to eliminate the scenario where a journalist requests recommendations several times in a row and receives a 'cycling set' of recommendations (from the three different algorithms). Using algorithm persistence the journalist would see the same set of recommendations. This

[4] http://trec.nist.gov/




(a) Calling the recommender with no `recId`.




(b) Calling the recommender with a `recId`.

**Figure 5.6:** Illustrating the process of algorithm switching, i.e. using the same retrieval methods for partial calls and cycling between retrieval methods otherwise in order to fairly evaluate all three retrieval methods.

is done to avoid users 'losing faith' in the system (thereby creating a rater's bias which would skew our ratings if they perceive the system to be unreliable, as shown in [72]). This could happen from getting what might appear to them as different recommendations each time they make a request (with no change in article text).

Using the approach described above allowed us to test the perceived accuracy of the recommender and of the three retrieval models in a live, production, content-based news-article recommendation scenario (see Section 7.7).

## 5.5 Database design

In order to allow accurate evaluation of the three similarity algorithms, it was imperative that very precise records were kept of all the transactions. Figure 5.7 gives an overview of the schema describing the recommender system's database. A database back-end utilising the MySQL[5] relational database management system (RDBMS) was implemented. MySQL is a pervasive, production quality, open-source RDBMS. The main motivation for using MySQL were threefold:

[5]http://www.mysql.com/

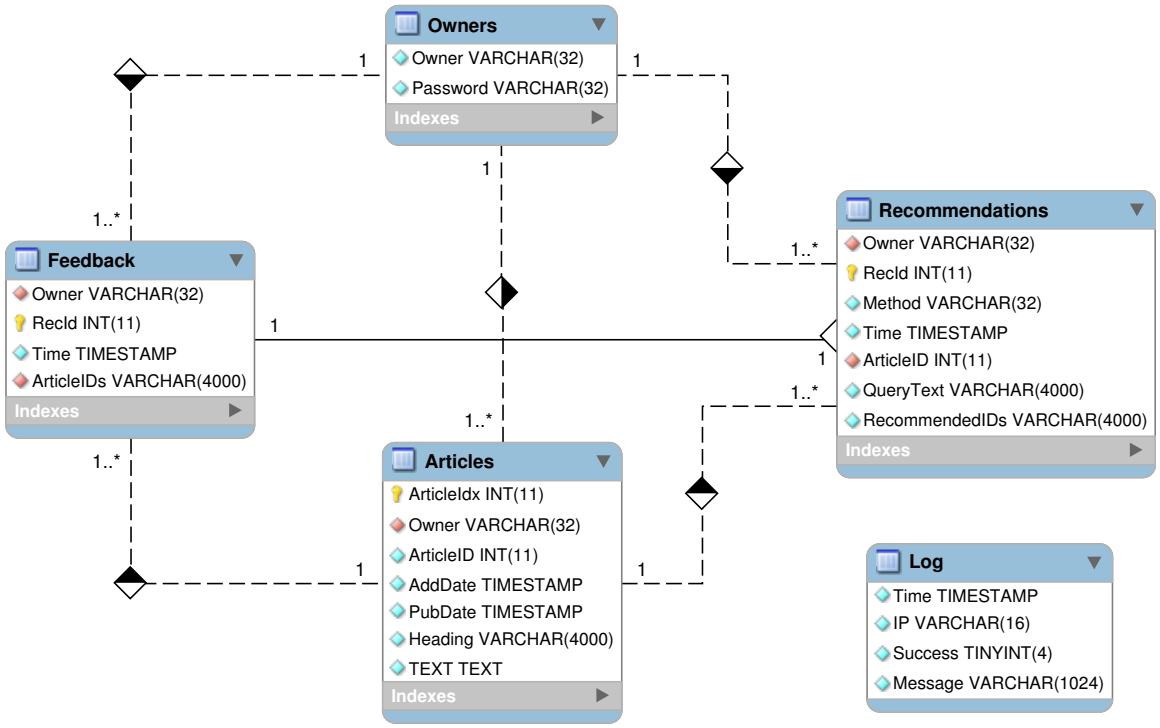



**Figure 5.7:** MySQL schema of the news article recommender database back-end.

1. We used PHP for implementing the Web Service which interacts directly with the database and PHP works out-of-the-box with MySQL.

2. MySQL comes pre-installed in most Linux distributions (which the Media Lab servers run) and has very good support channels.

3. It is free.

Figure 5.7 gives an overview of the database schema. In short, different users of the recommender service would be included in the **Owners** table, identified by a *Username* and *Password* field. Passwords are stored and compared using their hashed MD5 values. This is used for user identification on a call-by-call basis. Users pass their user name and password in the Web Service call. User names are used to create users' individual document indexes so that different users' documents do not interfere with one another.

All articles indexed by the recommender is stored in the **Articles** table. Text preprocessing (data sanitisation) is performed on every parameter before insertion into the database to prevent possible SQL-injections. All characters are replaced by their HTML escape codes upon insertion, and converted back to their 'normal form' upon retrieval.

Each recommendation the service produced is stored in the **Recommendations** table, logging the user who made the recommendation in the *Owner* field, the unique recommender session id (*RecId*), the algorithm used to make the recommendation (*Method*), the time the recommendation was made (*Time*), the specific ArticleId or the query text for which a recommendation was requested (*ArticleId* is set to -1 if a text query string is passed for partially complete articles) and all the recommended document ids are logged in the *RecommendedIDs* field (as comma-separated values).

All feedback obtained from users are stored in the **Feedback** table, logging the user who provided the feedback in the *Owner* field. The *RecId* is logged for cross-reference with *RecId* in the **Recommendations** table (this allows us to know exactly who received which articles in what order). The articles for which feedback was returned was logged as a comma-separated string in the *ArticleIDs* field. This way, for evaluation, we knew exactly which articles were recommended and in what order, by what algorithm, as well as which articles we received ratings for. As discussed in Section 7.7, this data was used to compute the mean average uninterpolated precision (MAP) scores to compare the different algorithms' performance.

## 5.6 Chapter summary

This chapter described a specific use case for document similarity algorithms, namely the recommendation of related news articles to journalists of Health24.com. This helps them to find related articles to the one they are writing or editing in a quicker and more convenient manner. The service was implemented as a stand-alone Web service in PHP and was integrated into their CMS to provide seamless access from the point of view of the journalists.

This chapter documents and motivates the design choices that were made during the implementation of this project. The service was used to collect relevance ratings from the journalists using the system, for evaluating the performance (in producing content-based-only recommendations to related news articles) of three information retrieval models (cosine Vector Space Model (VSM) with tf-idf, Okapi BM25 and query-likelihood statistical language models).

The performance of these algorithms will be evaluated and the results will be presented in Section 7.7.

# Chapter 6

# Data collection

Comparing different document similarity metrics over a dataset requires a properly constructed benchmark against which to compare (the 'gold standard'). In this chapter we present the methodology that was followed in designing the Web-based experiments that were used to collect similarity ratings for our proprietary Health24 dataset. We also discuss in detail how these ratings were collected and what measures were implemented to try and retain the 'good' ratings and discard the 'bad' ratings.

The second document similarity dataset which will be used in the document similarity experiments in Chapter 7 is the Lee & Pincombe [16] dataset, of which a brief overview is also presented at the end of this chapter.

## 6.1 Health24 dataset: Collecting and constructing the human gold standards

The corpus snapshot we obtained from health24.com contained 39,936 articles. This can be seen as a small-sized corpus in Information Retrieval (IR) terms, compared to other corpora, such as the Reuters RCV1 corpus [73], with more than 800 thousand documents, and the GOV2 corpus [74], with more than 25 million documents. However, for developing and testing various algorithms, including our own new algorithms, a small subset of this corpus was required.

The evaluation methodology we followed was inspired by an observation that most evaluation measures used in the literature tend to evaluate different aspects of the IR process, but not directly the comparison of two documents to calculate their similarity. For instance, typically, the evaluation procedure would involve submitting queries to an IR system and then rating its output as either relevant or non-relevant (a binary rating). Another popular approach is to use a similarity measure to automatically cluster a corpus of documents into groups which are then correlated with manual labels assigned to documents by users. The efficiency of an algorithm is then based on either the relevance rating in some reduced dimension space (to indicate related or non-related); or the number of correctly clustered documents.

Although these measures produce a good idea of the efficiency of these systems, as noted by Pincombe [3], one can never be entirely sure if errors that slip into the evaluation process are due to errors in the clustering algorithm, the human assigned labels (i.e. wrongly assigned), or if errors are due to the dimensionality reduction in the rating

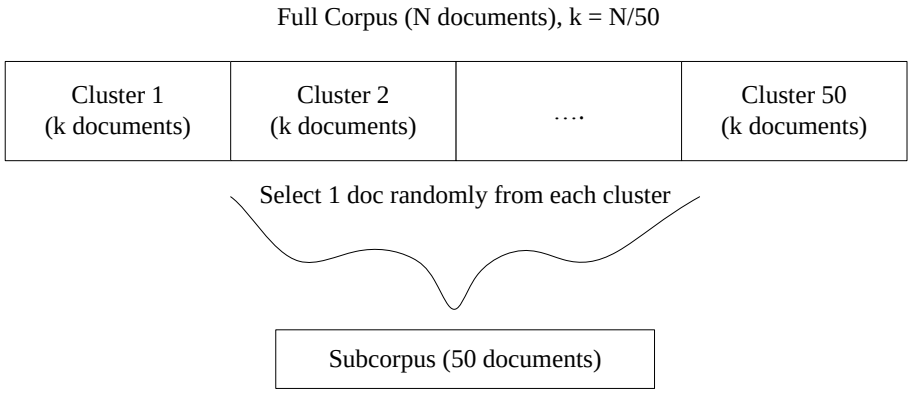



**Figure 6.1:** Visualising sub-corpus selection as a Cluster Random Sampling procedure.

process.

Since many areas, such as document visualisation, search engines, library retrieval systems, and so forth, rely on some measure to calculate the similarity between two documents, it is valuable to directly evaluate different measures for achieving that.

In this thesis, one of our goals is to evaluate different *document similarity algorithms*, which, given two documents $d_i$ and $d_j$, produce a similarity score $r$ which is representative of the relatedness between the two articles. In order to evaluate the document similarity algorithms, we followed an approach similar to the one used by Pincombe [3]. A small representative sub-corpus of 50 documents was selected from the full corpus, and then paired to produce a $50 \times 50$ *document*$\times$*document* matrix. These document pairs were presented to human relevance judges who rated them on a scale from 1 - 5, 1 being 'Not at all related' and 5 being 'Almost exactly similar'. These human ratings were then correlated with the measures produced by the various algorithms to measure their effectiveness.

The next sections describe the methodology that was followed for selecting a representative sub-corpus from the full 40,000 document corpus. The process of setting up Web-based tests to collect 'gold standard' ratings for the representative corpus (against which to compare the different algorithms) is discussed in detail. We also discuss the Lee & Pincombe corpus, a popular 50-document corpus for evaluating document similarity metrics in the literature [3].

### 6.1.1 Sub-corpus selection

Croft [43] provides some guidelines to help with sub-corpus selection. One of the main points mentioned is to use a sub-corpus that is representative of the full corpus.

In selecting the sub-corpus, it was important to have a good representation of all the documents in the full corpus. We used a variation of a method called Cluster Random Sampling (see Figure 6.2) to achieve this. Documents were ranked by their publication dates. Then, to achieve a sample of 50 documents, the full list of documents were divided into 50 clusters. One document was then extracted from each of the clusters according to a uniform random distribution.

Some of the articles contained in the corpus were not very good candidate articles (e.g. containing only one sentence, or just a hyperlink to another article, see Listing 6.1). To account for the possibility where an unwanted article was selected, and to make the

**Listing 6.1:** Examples of bad article artefacts in Health24 corpus.

```
ArticleId 5510: More news on News24.
More news on News24.
Read more news on <a href="http://www.news24.com/">News24</a>

ArticleId 6502: More on Abuse.
<A class=main_link href="Default.asp?action=article&ContentID=11888">
Shaken baby syndrome</a>

ArticleId 40037: sdfsd.
kjslkafj kjhjh
```

whole process easier, a Web-based sub-corpus selection tool was created using PHP and JavaScript which aided in the selection process (see Figure 6.2).

Using this Web application, one can enter the desired number of articles and it automatically selects that many articles from the database using the principle of Cluster Random Sampling explained above. Clicking on an article would retrieve and display its text for evaluation. If an article is considered undesirable, the 'Replace Article' option would replace the unwanted article with another article uniformly randomly selected from the **same** cluster as the unwanted article. When the user is satisfied with the selection, an option to 'Save this List' would persist the collection of article ids to disk for use as the test sub-corpus.

### 6.1.2 Web-based experiments

Semantic similarity is a very subjective process. In order to evaluate the efficiency of the various available algorithms, we need a so-called 'gold standard' (known truth) to compare them to. As noted in Section 6.1, most comparisons in the literature have been done using binary – and much fewer using graded – ratings of relevance in an IR setting. Much less have there been systematic experiments done where a similarity measure was compared with a human gold standard.

The only other study of this nature, to the best of our knowledge, is a study done by Pincombe and Lee in 2004 [3], which measured the correlation between Latent Semantic Analysis similarity measures and those obtained from human participants. This data was obtained from Lee through personal correspondence, and was used as the gold standard in one of our experiments (see Section 6.2).

In order to measure the performance of the various algorithms in the particular domain of Health24.com's corpus, a Web-based experiment was set up. All documents in the 50-document sub-corpus were paired, excluding self-comparisons, to generate a 50×50 document×document matrix (1,225 pairs when self-pairs are excluded).

Most other similar experiments we came across in the literature used some form of Web-based interface for rating documents; some partially supervised, other completely unsupervised. However, this sort of unsupervised test environment introduces the question of data quality, or data integrity. To address this issue, we decided on including several loaded 'control pairs' in the rating procedure.

*Control pairs* were hand-picked for each document in the sub-corpus. In the sub-corpus


**Figure 6.2:** A screen shot of the sub-corpus selection Web application.

selection Web-application (see Figure 6.2) a feature was added to 'Get Top N Hits'. The recommender back-end is used to find a selection of N most related documents to the focus article, which were not already contained in the 50-document sub-corpus. From this list we hand-picked an article, called the 'top control', which was considered very similar to the focus article, out of the full corpus of ± 40,000 documents. For each focus article, a 'bottom control' article was also picked which was considered highly *unrelated* to the focus article.

**Table 6.1:** Example focus article and its 'top control' and 'bottom control'.

| Focus article. | Top control for focus article. | Bottom control for focus article. |
|---|---|---|
| If you want your stomach to have that rippled, muscular look, you'd better first hope that you inherited the right genetic material, because achieving this look is difficult. And, for some, it's impossible - no matter how hard they try they'll never develop so-called 'washboard abs' or a 'six pack'- simply because they don't have the genes for it. Along with good genes, you also need a lot of spare time to spend working out in the gym and analysing the food you eat. To develop washboard abdominal muscles, women need to lower their total body fat to 15 percent to 18 percent, and men can only carry 7 percent to 10 percent body fat. Healthy women usually have about 21... | Want abdominals to do your washing on? Sure you do - women love to touch it (yeah!), and it does wonders for your posture. But, getting that elusive six-pack is a full-time job for many. Washboard abs are the ideal when it comes to being in shape. However, it's not a realistic ideal for most men, just as the media-generated female image is not perfection to most women. But for those men want to give it a try anyway, here are some tips: First, quit your job. You will need the extra time to work on your abs. One male model who has a near-perfect body says he starts his day with an hour on the treadmill, burning... | Spending your vacation in an exotic location is exciting, and nowadays its a luxury that has to be carefully saved for and planned beforehand. Whether youre touring the ancient sites, going on a let-your-hair-down package round of beaches and nightclubs, or whether youre into natural wonders and white-water rafting adventures, dont forget to do a little health research on the region. Get your answers from the Travel expert. Read more: (Joanne Hart, Health24, June 2009) |

This process produced a collection of 150 documents: the sub-corpus of 50 documents, plus 50 'top controls' and 50 'bottom controls'.

This collection of documents were then paired up as follows: The 50 documents were paired up into 1,125 pairs. For document $d_i$ and $d_j$, pair $d_i d_j$ was considered equivalent to pair $d_j d_i$. These pairs were presented in random order to raters. Scattered in-between these pairs, a set of 300 control pairs were included, consisting of 50 'top controls' and 50 'bottom controls' – thus 100 pairs in total – cloned 3 times to create 300 control pairs. The reasons for cloning the control pairs were twofold: Budget and time allowed us to receive about 4,500 ratings of article pairs in total. For 1,225 article pairs, this means roughly three ratings per article-pair, in order to calculate averages so that one bad rating does not unduly skew the document pair-rating. Three ratings per pair were considered a suitable trade-off between available time and rater accuracy. Typical similar experiments use one rating per pair (binary, i.e. relevant or non-relevant); Pincombe [3], however, used between 8 and 12 ratings per pair, which is the maximum number we have come across in the literature.

Cloning the control pairs three times ensures that three times more ratings are received for the control pairs than for the other pairs. This is because one 'run' consists of $1,225 + 300 = 1525$ documents, and each control pair is contained three times in one such run. Budget and time allowed for three complete runs ($4,500$ ratings), which resulted in three times more ratings per control pair on average. Raters would typically rate a control pair every $\frac{1,225+300}{300} \approx 5$ pairs. We require roughly six control ratings from a user in order to

correlate their ratings with other users and to decide if we would make use of their ratings. A lower cut-off of 30 ratings per user was therefore established, for ratings received from a user to be included in the study. If a user rated less than 30 article pairs we excluded their ratings, since we could not suitably correlate their ratings over control pairs with other users to detect 'click fraud', i.e. when users made nonsense ratings.

To avoid the debate between 'similarity' and 'relatedness', raters were asked to think of these experiments as follows: "A good way to think about this is, if you read one of these articles on an online news site, would you find the second article related or relevant to the first article or not?" Students were presented with instructions to read both texts carefully before rating them, keeping the following scale in mind:

1. *Not at all related*: no shared words, no shared concepts. No clear link between the two articles.
2. *Somewhat related*: some concepts shared, i.e. 'big', 'blue' or 'house', but in very different contexts.
3. *Moderately Related*: some words shared, some concepts shared, but ultimately different scenarios
4. *Very related*: Same idea, different words; the same idea expressed differently or the same story, where one includes a little more detail
5. *Almost exactly similar*: same words, same concepts OR exactly the same meaning but different (synonymous) words

Raters were also given the following advice: "Most articles relate to some aspect of **health**, don't let that unduly influence your judgement. Rather focus on the specific content. I.e. an article about 'colon cancer' is more related to an article about 'skin cancer' than to an article about 'pneumonia'."

Figure 6.3 shows a screen shot of the Web-based ratings page.

Students logged in with their university credentials and could then start rating articles. Times were logged between ratings, and each run of 30 ratings were recorded under a different session id which were regulated with PHP sessions. This allowed us to correlate each session's ratings with other raters' ratings for those documents, and to throw out bad sessions, but keep good sessions. For instance, if a rater in one session rates their control pairs with very bad correlation to other raters, we could throw out just that session – since it might have been a fluke – instead of having to throw out the user's entire set of ratings.

We considered using scripts which disable the 'Submit Rating' button for some number of seconds upon loading the page, to try and ensure that raters read an article and cannot just click through ratings. However, it was decided that this might in fact be counter-productive, since it would indicate to raters one of the measures that was used to throw out bad ratings (very short times between ratings), which would make it easier for raters to fake ratings. For instance, if one allows raters to just click through articles, one can at least pick up when they do so. However, if one disables the ratings button for say 30 seconds one has no measure of whether raters actually read the article, or whether they just wait for the time-out. It is better to correlate their times taken and their ratings with other users as an indication of 'fitness' of rating. The control measures we implemented

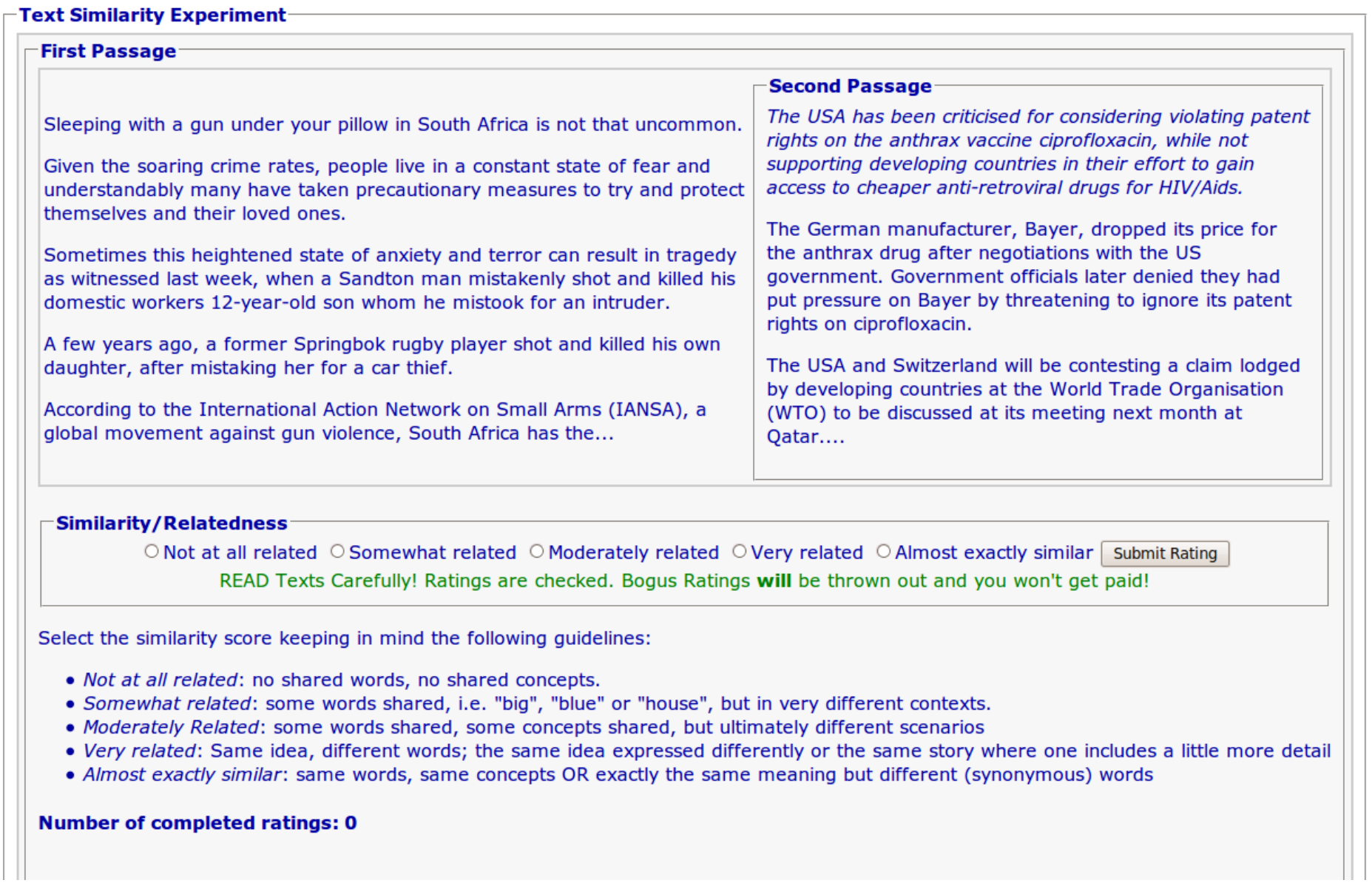



**Figure 6.3:** A screen shot of the Web-based experiments page.

**Table 6.2:** Comparing rating distributions between our experiments and Lee [3].

| | 1 | 2 | 3 | 4 | 5 |
|---|---|---|---|---|---|
| Lee | 0.64 | 0.18 | 0.10 | 0.06 | 0.02 |
| Us | 0.74 | 0.13 | 0.06 | 0.04 | 0.03 |

**Table 6.3:** Rating distributions obtained through Web-based experiments for the control pairs in the Health24 document corpus.

| | Nr of Ratings | Min Rating | Max Rating | Mean |
|---|---|---|---|---|
| Top Controls | 455 | 1.8 | 5.0 | 3.5 |
| Bottom Controls | 457 | 1.0 | 1.9 | 1.18 |

**Table 6.4:** Number of 'good' ratings received per document-pair after data sanitisation.

| Nr of Ratings | 1 | 2 | 3 | 4 | 5 |
|---|---|---|---|---|---|
| Nr of Document Pairs | 122 | 359 | 652 | 33 | 1 |

were deemed adequate and on par with other tests described in the literature, and also aggressive enough to weed out bad ratings.

### 6.1.3 Processing the ratings

In this section we present the actual statistics making up the Health24 gold standard of ratings. Altogether, 4,494 ratings were received from 31 student raters participating in the Web-based experiment. Table 6.2 shows the distributions of ratings received for our 50-document corpus, compared to the distributions reported by Lee et al. [16] for their 50-document corpus. In general, we see a consistent heavy skew towards lower (unrelated) ratings across both corpora.

As discussed, loaded control pairs were presented with a higher frequency to obtain a higher distribution of ratings from which to estimate suitable cut-off levels for the inclusion or exclusion of ratings. Table 6.3 shows the number of ratings obtained for top and bottom controls. Altogether, 455 ratings were received for top control pairs, with averaged ratings ranging from 1.8 to 5.0, and with a mean of 3.5. Bottom control ratings (457 in total) ranged from a lowest value of 1.0 to 1.9 with a mean of 1.18.

In total, 159 sessions were recorded, consisting of a maximum of 30 ratings each. All sessions with less than 30 ratings – for instance if a user started but stopped midway through the experiment – were thrown out point-blank. This resulted in the removal of 13 sessions, based on insufficient ratings.

Users received on average six control tests per session, and they passed 80% of these. We set a lower cut-off rate at 75% pass rate and thus eliminated a further 23 sessions which did not pass the minimum number of control tests per session.

After performing these data sanitising operations, we were left with 2,929 actual ratings, with the distributions shown in Table 6.4 and Table 6.5.

As can be seen in Table 6.5, the predominant amount of ratings received were strongly

**Table 6.5:** Final percentages of ratings received for individual values (1, 2, etc.) after data sanitisation.

| Actual Rating | 1 | 2 | 3 | 4 | 5 |
|---|---|---|---|---|---|
| % Ratings | 84 | 12 | 2 | 1 | <1 |

unrelated. This agrees with Lee's findings [16].

## 6.2 Lee & Pincombe dataset

A collection of fifty documents was used from the Australian Broadcasting Corporation's news mail service [16]. Documents in this test collection are between 51 and 126 words long and cover a multitude of general news topics. In their study, ratings were collected from 83 University of Adelaide students who were paid a nominal fee to participate in the experiment. Documents were paired in all possible ways, similar to how our documents are paired, and each of these 1,225 document pairs received a total of 8 to 12 human ratings, which were averaged for each pair.

Document pairs were presented in random order and document placement (left or right) for each pair was also randomly determined. Averaging these human judgements for each document produces 67 distinct values.

We used this dataset together with our proprietary Health24 dataset, since it covers a wider range of topics and has more average ratings per document-pair than ours. This provides a more fine-grained gold standard to correlate to.

## 6.3 Chapter summary

This chapter discussed the approach which was followed to construct a gold standard of ratings to use as benchmark for all the experimental tests that will be conducted in Chapter 7 in order to measure the performance of the different document similarity models presented thus far.

We discussed and motivated the process which was used to create the 50-document sub-corpus from the 40,000-document Health24 corpus.

The Web-based experiments which were used to obtain the human gold standard ratings for the sub-corpus were discussed. We laid out the design choices which were made and the control checks that were put in place at different levels. This was done in an attempt to ensure maximum data-integrity (against, for instance, raters just 'clicking through' ratings).

The results from processing the ratings were shown, and shows a heavy skew towards low (unrelated) ratings. This agrees with similar studies using comparable strategies [3].

# Chapter 7

# Experiments

In this chapter several experiments are conducted to evaluate the performance of the proposed technique for Spreading Activation (SA) which was presented in Chapter 3. We obtain the best-performing configuration for these parameters by correlating performance over a dataset of inter-concept similarity ratings between word pairs.

The effects that document preprocessing techniques can have on document similarity performance, as introduced in Chapter 4, is evaluated and discussed. We also evaluate the novel document similarity metrics proposed in Chapter 4 by conducting several experiments over our proprietary Health24 document similarity dataset and the standard Lee & Pincombe [16] document similarity dataset. These datasets were discussed in Chapter 6.

The performance of Normalised Compression Distance (NCD), the cosine Vector Space Model (VSM) with term-frequency inverse document-frequency (tf-idf), Okapi BM25, statistical language models, and finally Latent Semantic Analysis (LSA) is also evaluated over these same datasets in order to fairly compare all document similarity metrics evaluated for this study.

Finally, the performance results obtained from the Health24 journalists using the news-article recommender system discussed in Chapter 5 are presented and discussed.

Preliminary conclusions are presented for each experiment. More significant results will be discussed in more detail in Chapter 8.

## 7.1 Inter-concept similarity experiments using spreading activation

### 7.1.1 Purpose

The model for spreading activation over the hyperlink structure of Wikipedia, as introduced in Chapter 3, relies on several important parameters, namely the weighting scheme $w$ (TAA, AA-cos and AA-wlm), network decay $K_{\text{decay}}$, activation threshold $T$ and the maximum path length to spread over $L_{p,\max}$. These parameters influence the levels of activation that reach different parts of the network. Since we interpret the resulting activated nodes and their activation levels as indications of semantic relatedness, we need to optimise these parameters to best reflect or mimic that.

**Table 7.1:** Sample pairs and their human ratings (out of 10.0) from our random subset of the the WordSimilarity-353 corpus used to evaluate inter-concept similarity.

| Concept One | Concept Two | Human Rating |
|---|---|---|
| sugar | approach | 0.88 |
| opera | industry | 2.63 |
| governor | interview | 3.25 |
| day | summer | 3.94 |
| computer | news | 4.47 |
| death | inmate | 5.03 |
| grocery | money | 5.94 |
| lover | quarrel | 6.19 |
| competition | price | 6.44 |
| minister | party | 6.63 |
| hundred | percent | 7.38 |
| Mexico | Brazil | 7.44 |
| dividend | payment | 7.63 |
| life | death | 7.88 |

### 7.1.2 Experimental method

In order to evaluate the performance of these proposed approaches, experiments were conducted over the WordSimilarity-353 dataset (WS-353) [33]. A small sample of this dataset is given in Table 7.1. The full subset of 50 randomly-selected word-pairs used for our experiments is given in Appendix A. WS-353 contains two sets (altogether 353 pairs) of English word-pairs, along with human-assigned similarity judgements. It is the *de facto* dataset used in the literature to evaluate word-based similarity measures, and was chosen in order to be able to compare our results to the state of the art.

The literature related to the two other major Wikipedia-based approaches, Wikipedia Link-based Measure (WLM) and Explicit Semantic Analysis (ESA), is vague regarding how the performance of these methods were evaluated. Both approaches were evaluated by choosing fifty word-pairs from the WS-353 dataset, and correlating the human-assigned similarity ratings with the particular algorithm's score of relatedness. However, this approach can easily lead to overly optimistic ratings in the event where sample set that was chosen happens to be favourable to the algorithm, even though it might not fare as well using a different sample.

In order to reduce the possibility of overestimating our algorithm's performance in the way described above, we made use of a procedure called **repeated holdout** [75]. To evaluate the performance of the algorithm for a specific set of parameters, this approach can be described as follows: Given a sample test set of $N$ elements (in our case, pairs of words with human-assigned ratings of relatedness), divide this set randomly into $l$

**Table 7.2:** Performance comparison for the three spreading strategies (TAA=Target Activation Approach, AA=Agglomerative Approach, $L_{p,\max}$ = maximum path length used, ED=energy distribution only, ILF=Inverse Link Frequency, NILF=normalised ILF.)

| Spreading Strategy | $\rho_{\max}$ | Parameters |
|---|---|---|
| TAA | 0.56 | ED, $L_{p,\max} = 3, K_{\text{decay}} = 0.6, T = 0.001$ |
| AA-wlm | 0.60 | NILF, $L_{p,\max} = 3, K_{\text{decay}} = 0.1, T = 10^{-6}$ |
| AA-cos | 0.70 | ILF, $L_{p,\max} = 3, K_{\text{decay}} = 0.5, T = 0.1$ |

parts[1] of roughly equal size. In an iterative fashion, hold out one part of the data and evaluate the performance of the algorithm on the remaining $l - 1$ parts until all $l$ parts have been held out once. Average the algorithm's performance over all $l$ runs into one score resembling the algorithm's performance for that specific set of parameters.

Since there are five parameters (spreading strategy, path length, weighting scheme, decay and threshold), we implemented a **grid-search** procedure by holding three of the five parameters constant, and evaluating several combinations of the remaining two parameters by stepping over the possible parameter values using first a coarse and then a finer-grained step size. For instance, since decay and threshold both fall between 0 and 1, we can evaluate the effects of both parameters by constructing a grid of one hundred ($K_{\text{decay}}$,$T$) pairs for $K_{\text{decay}}, T \in \{0.1, 0.2, 0.3, ..., 1.0\}$.

This approach was feasible, since the parameter space was fairly small. It was also desirable, since it reduced the possibility of optimising to local maxima, whilst simultaneously reducing the possibility of overestimating the technique's performance.

We implemented the parameter optimisation procedure in Python using a modular object-oriented design as shown in Figure 7.1. The `GraphModule` loads the Wikipedia hyperlink graph structure. The `ParameterOptimisation` module loads the gold truths and the parameter ranges over which to optimise. These include the three spreading strategies, path lengths of one to three, TAA, AA-cosine and AA-wlm weighting schemes, and values for $K_{\text{decay}}$ and $T$.

The `GridSearch` module accepts a pointer to the graph structure, the word (node) pairs in the sample test set, the gold truths and the parameter ranges. It generates $l$ new sample sets from the original sample set as per the repeated holdout procedure described above. For all combinations of spreading strategy, path length and weighting scheme, the *holdOutAndCorrelate()* function spreads over the $l$ sample sets for all $(K_{\text{decay}}, T)$ pairs in the parameter ranges, and averages the $l$ results into one correlation value for that specific $(K_{\text{decay}}, T)$ pair.

These results are logged to disk via the `Logging` module, and the best runs for each strategy are plotted as a 3-dimensional graph in correlation space over the $(K_{\text{decay}}, T)$ parameter space using the `matplotlib` library.

### 7.1.3 Results

Five parameters are evaluated, namely the spreading strategy (TAA, AA-cos, and AA-wlm), the weighting scheme (pure energy distribution, ILF, and NILF), maximum path length ($L_{p,\max}$), and finally network decay ($K_{\text{decay}}$) and threshold ($T$). Experiments are

[1] $l$ was chosen as 5 in our experiments.

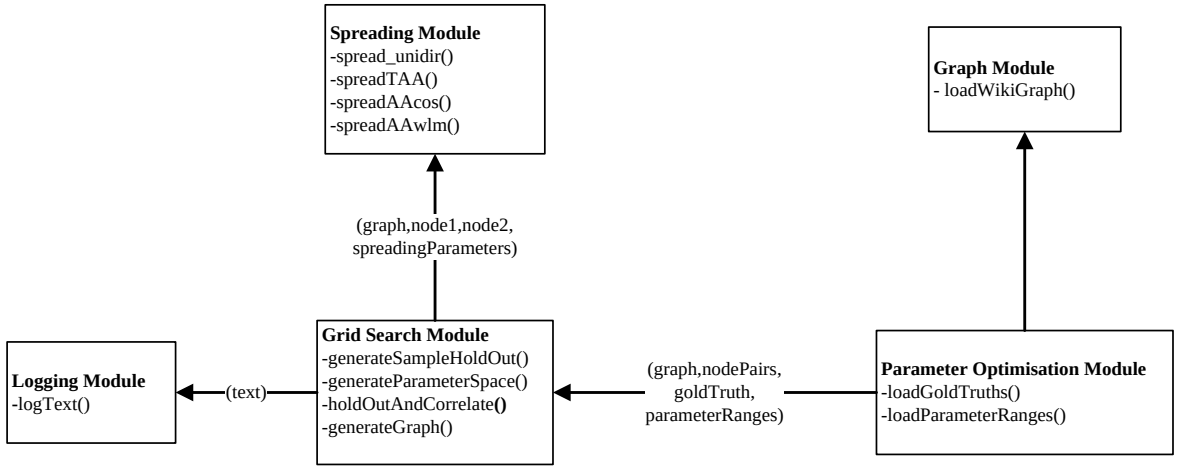



**Figure 7.1:** Simplified diagram showing the experimental setup that was used to find the optimal spreading parameters, by using the WordSimilarity-353 corpus of word pairs and similarity ratings as gold truth.

**Table 7.3:** Spreading results by maximum path length $L_{p,\max}$.

| $L_{p,\max}$ | $\rho_{\max}$ | **Parameters** |
|---|---|---|
| 1 | 0.47 | TAA, ED/ILF/NILF |
| 2 | 0.66 | AA-cos, ILF, $K_{\text{decay}} = 0.4, T = 0.1$ |
| 3 | 0.70 | AA-cos, ILF, $K_{\text{decay}} = 0.5, T = 0.1$ |

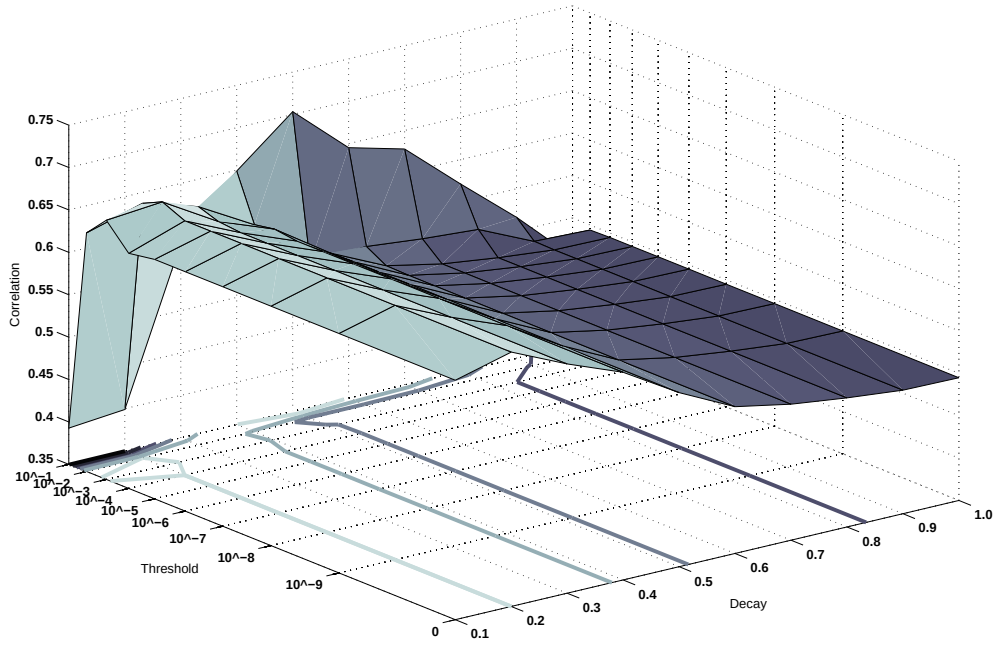
**Table 7.4:** Spreading results by weighting scheme. ED = energy distribution only, ILF = inverse link frequency, NILF = normalised ILF.

| **Weighting Scheme** | $\rho_{\max}$ | **Parameters** |
|---|---|---|
| NILF | 0.63 | AA-cos, $L_{p,\max} = 3, K_{\text{decay}} = 0.9, T = 0.01$ |
| ED | 0.64 | AA-cos, $L_{p,\max} = 3, K_{\text{decay}} = 0.9, T = 0.01$ |
| ILF | 0.70 | AA-cos, $L_{p,\max} = 3, K_{\text{decay}} = 0.5, T = 0.1$ |

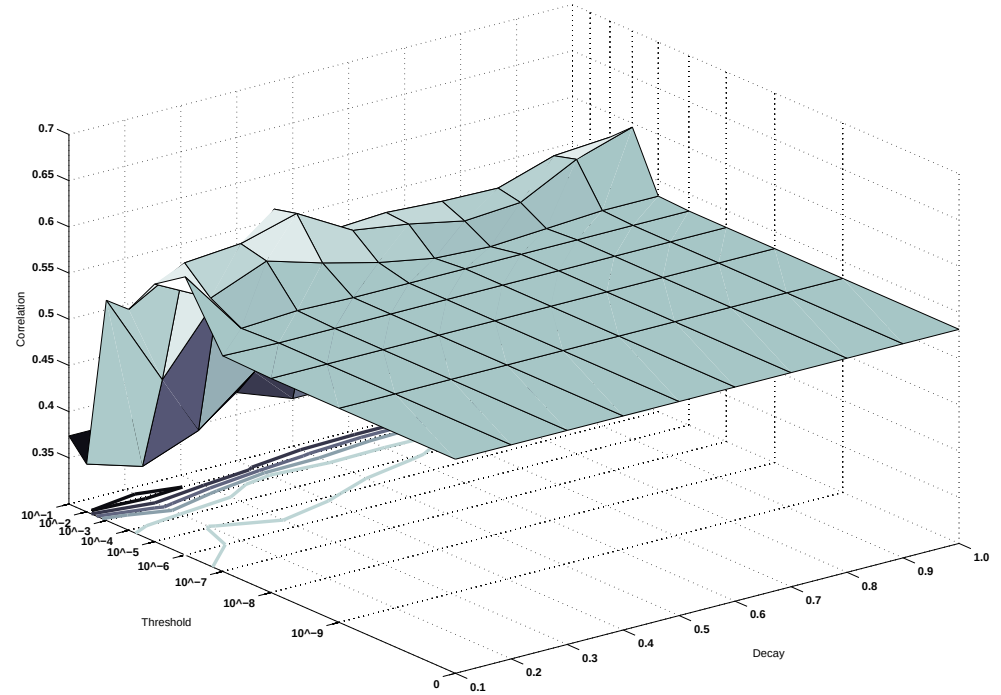
divided into three parts to measure the influence of the first three parameters respectively, by conducting a grid search over the last two parameters (decay and threshold).

For each grid search, a coarse-grained search was first conducted over $K_{\text{decay}}$ and $T$ with a step size of 0.1 over $K_{\text{decay}}$ and a logarithmic scale over $T$, thus $T = \{0, 0.1, 0.01, 0.001, ...\}$. The best values of $K_{\text{decay}}$ and $T$ were then chosen to conduct a finer-grained grid search.

The results for the inter-concept similarity experiments are visualised in Figure 7.2 and Figure 7.3. We will now discuss the influence of spreading strategy, maximum path length, and weighting scheme on the results obtained.




(a) AA-cos grid search visualisation.




(b) AA-wlm grid search visualisation.

**Figure 7.2:** Visualisation of grid search results for the agglomerative approach. Lighter is higher (better), darker is lower (worse).

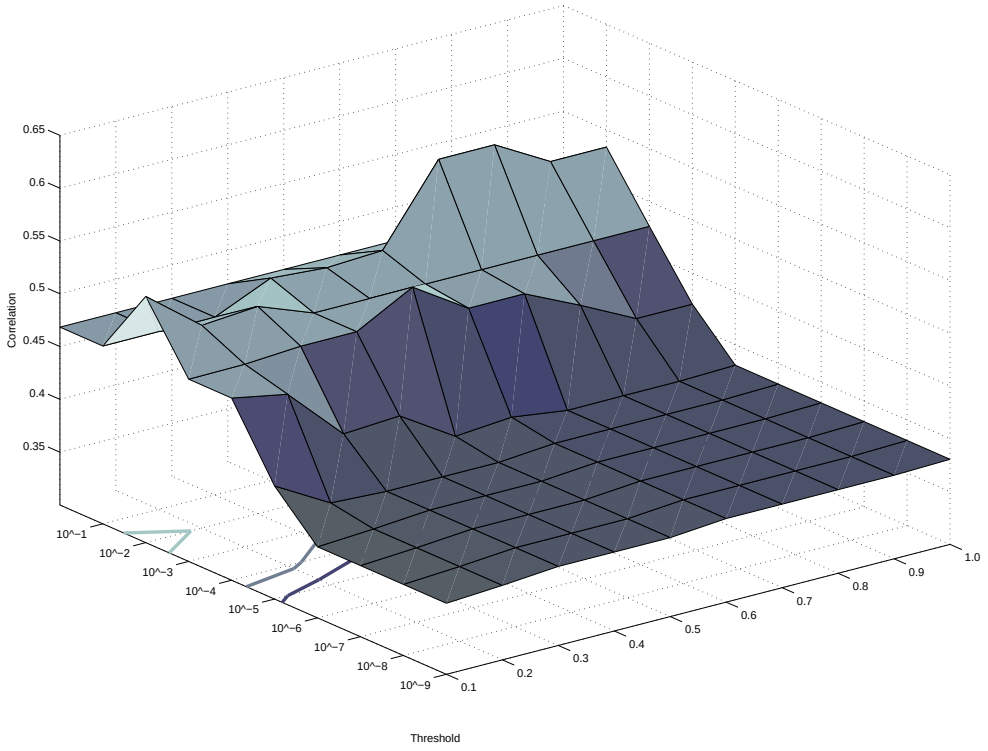



**Figure 7.3:** Visualisation of grid search results for the Target Activation Approach spreading strategy (TAA) results. Lighter is higher (better), darker is lower (worse).

#### 7.1.3.1 Spreading strategy

The spreading strategy determines how activations resulting from the spreading process are converted into scores of relatedness or similarity between two nodes. Table 7.2 summarises the best results obtained for each of the three strategies, with the specific set of parameters that were used in each run.

Firstly, results tend to be better using the agglomerative approach than using the target activation approach ($\rho_{\max} = 0.56$ for TAA versus $\rho_{\max} = 0.70$ for AA-cos).

Secondly, the cosine spreading strategy outperforms the WLM spreading strategy over this sample set (correlation $\rho = 0.60$ for AA-wlm versus $\rho = 0.70$ for AA-cos.

#### 7.1.3.2 Maximum path length

Path length is related to how far one node can spread its activation in the network. In an associative network, since neighbour nodes are all associated with a given node, and their neighbour nodes are all by extension associated with them, it is a valid assumption that nodes can be related to one another over longer path lengths. We assumed this to be true for the Wikipedia graph in order to make use of a spreading activation algorithm (see our initial hypotheses in the first chapter).

In this experiment we test this hypothesis by limiting the path length to one, two, and finally three hops between nodes. If this hypothesis is invalid, then longer path lengths should lead to worse results, since the spreading process would activate nodes unrelated to the source nodes.

Table 7.3 summarises the results for this experiment. It is clear that increasing path length from one to two hops increases performance from $\rho_{\max} = 0.47$ to $\rho_{\max} = 0.66$. Moreover, increasing path length from two to three hops furthermore increases performance from $\rho_{\max} = 0.66$ to $\rho_{\max} = 0.70$.

#### 7.1.3.3 Weighting scheme

In an associative network, each edge has a real-valued weight denote the *strength of association* between the two nodes it connects. The derived Wikipedia hyperlink graph does not possess these weights, therefore we proposed three different *weighting strategies* in Section 3.4 to estimate these weights. These are pure energy distribution (ED), Inverse Link Frequency weighting scheme (ILF), and normalised ILF (NILF). NILF was introduced to measure if ILF's ability to boost node activations could lead to improved performance.

Table 7.4 summarises the results of this experiment. Firstly, all three weighting schemes achieve their individual best results using the AA-cos approach. Secondly, ILF outperforms both ED and NILF. Thirdly, both ED and NILF perform best for higher values of network decay (both 0.9) and smaller values of threshold (both 0.01) compared to ILF (0.5 and 0.1 respectively for decay and threshold).

We attribute this observation to the boosting effect of ILF weightings, which can be illustrated as follows: Let $\text{ILF}(v_i)$ be the ILF weighting of some node $v_i$. In order to have $\text{ILF}(v_i) > 1$, we need $\text{ILF}(v_i) = \log\left(\frac{|G|}{|\mathcal{N}(v_i)|}\right) > 1$, and thus $\frac{|G|}{|\mathcal{N}(v_i)|} > 10$. For Wikipedia, $|G| \approx 3 \cdot 10^6$, which means $|\mathcal{N}(v_i)| < 3 \cdot 10^5$. Thus, every node connected to less than 300,000 nodes will have an ILF-weight of larger than 1, thus boosting that connection to some degree.

Recall from the discussion on *Fan-out constraint* in Section 3.4, that nodes which link to fewer nodes are usually more **specific** concepts in Wikipedia (such as `Hair Pin`). We argued that a path linking two concepts via these more specific concepts are more indicative of a stronger semantic relationship than through some very general concept (such as `USA`). In the ILF weighting scheme, these less connected nodes are automatically weighted as more important by boosting the respective edge weight above unity.

We relate this to ILF's higher performance compared to ED and NILF, whilst using a lower decay and higher threshold parameters as follows: Assume that our hypothesis is correct, and that a higher ILF rating does in fact point to a semantically more meaningful connection. Then, by using a *lower* decay parameter and a *higher* threshold parameter, ILF effectively limits the amount of non-important nodes that are activated, since their activations are more quickly decayed whilst at the same time requiring a higher threshold to continue spreading.

On the other hand, important nodes (semantically more related nodes) are boosted and more important nodes thus spread further than non-important nodes. This results in activation vectors that capture the semantic context of the source node more accurately, which leads to higher performance.

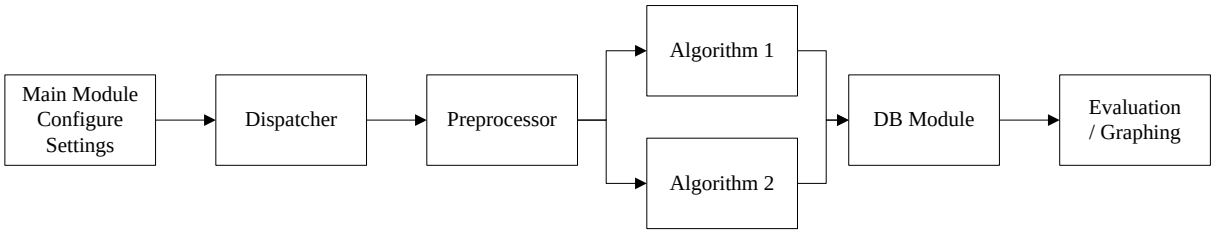



**Figure 7.4:** Preprocessing pipeline to investigate the effects of lexico-syntactic term space modification on correlation performance.

## 7.2 The effect of document preprocessing on similarity performance

### 7.2.1 Purpose

It is common practice to preprocess document text before indexing or computing similarity. This process typically involves removing high-frequency stop words and stemming words to a normalised base form.

We are not aware of any study where the effects of these common preprocessing steps are quantified in terms of percentage improvement over the baseline (not performing any preprocessing). Therefore, we want to evaluate what effect these and different combinations of preprocessing techniques have on document similarity accuracy.

Three additional lexico-syntactic preprocessing techniques were identified in Section 4.2, and will also be tested in this experiment. These include: appending part-of-speech tags to words, converting words in text to bigrams, and converting words to trigrams.

### 7.2.2 Experimental method

Experiments are carried out using the NCD and the cosine VSM using tf-idf over the proprietary Health24 dataset and the standard Lee [16] document similarity dataset (discussed in Chapter 6).

An object-oriented preprocessing pipeline system was implemented in Python with which the effects of various preprocessing processes could be analysed on the performance of the Normalised Compression Distance (NCD) and cosine similarity with tf-idf.

Individual units as shown in Figure 7.4 were implemented as separate classes. The *Main* module is initialised with several pipelines of preprocessing tasks as Python lists (e.g. ‘‘`TOK STP COMBINE`’’ would be a pipeline to tokenise text, remove stop words, and combine the tokens back as a single string, please see Table 7.6). A *Dispatcher* unit is initialised with a specific algorithm to use and a pipeline to implement. A *Preprocessor* is then initialised with either the Term Space Substitution (TSS) or Term Space Augmentation (TSA) model and it is passed the pipeline string. The pipeline is implemented as a Python dictionary-based mapping to function calls based on the pipeline tokens passed, e.g. ‘‘`TOK`’’ was mapped to `Tokenise()` via `function_map = {‘TOK’ : Tokenise}` and then called with `function_map[pipeline_token]()` for every token (preprocessing process) in the pipeline string.

All algorithms implement a `preprocess()` function (for computing tf-idf values for cosine, not applicable to NCD) and a `compare()` function which compute the similarity

**Table 7.5:** Best results using different document preprocessing pipelines on the Health24 and Lee datasets, compared to a baseline with no preprocessing (all results are Pearson linear product-moment correlations, and best results in each row are highlighted). Please refer to Table 7.6 for an explanation of what each code means.

| | **TSS** | **TSA** | **TSA** (stp stm base) | **Baseline (raw)** | **% improve** |
|---|---|---|---|---|---|
| Health24 Cosine | 0.49 (STP,STM) | 0.49 (STP,STM) | **0.50 (STP)** | 0.39 | +28% |
| Health24 NCD | 0.25 (STP,TRI) | 0.25 (STP,STM,BI) | **0.27 (STP,STM,BI)** | 0.15 | +80% |
| Lee Cosine | 0.56 (STP,STM) | 0.55 (POS,STP,STM) | **0.57 (POS,STP)** | 0.51 | +9.8% |
| Lee NCD | 0.29 (TRI) | **0.37 (STP,STM,BI)** | 0.31 (BI) | 0.24 | +54% |

**Table 7.6:** Different preprocessing models and techniques and the codes they are represented by.

| | |
|---|---|
| TSA (stp stm base) | TSA model with base text stopped and stemmed |
| TSA Only | TSA model where the base text is unprocessed |
| STP | stopwords removed |
| STM | words stemmed |
| BI | terms converted to bigrams, e.g. ‘‘`she said so`’’ → ‘‘`she_said said_so`’’ |
| TRI | terms converted to trigrams, e.g. ‘‘`the party was great`’’ → ‘‘`the_party_was party_was_great`’’ |
| POS | part-of-speech tags appended to terms, e.g. ‘‘`the car`’’ → ‘‘`the_ART car_NN`’’ |

score which is passed to the database unit for persistence into a MySQL database.

The *Evaluation* module correlates all the similarity scores with human gold standard scores which were obtained by conducting controlled Web-experiments. Chapter 6 describes the two document similarity datasets which were used for the experiments and how human ratings were collected.

### 7.2.3 Results

The results of the document preprocessing experiments are shown in Appendix B. Each graphic shows the Pearson correlation of the shown metric (cosine VSM or NCD[2]) with the specified lexical and syntactic preprocessing steps applied to the input text.

The first observation from Table 7.5 is that cosine similarity outperforms NCD in all categories.

Secondly, NCD approaches benefit most from lexico-syntactic preprocessing with improvements in correlations of 54% and 80% over the two datasets respectively over the baseline model. Nonetheless, the best result obtained for NCD of $\rho = 0.37$ is still much lower than $\rho = 0.57$ obtained using the cosine similarity.

Thirdly, cosine similarity achieves its highest scores with stop words removed and part-of-speech (POS) tags appended to words in the TSA model with a stopped and stemmed

[2]NCD was implemented as shown in Equation 2.3.4, using a bzip2 compression algorithm.

base (i.e. stop word removal and stemming is performed on document text to which the preprocessed terms are appended). This result, however, is very close to the value obtained for stopped and stemmed text using TSS – the 'traditional' approach to using cosine VSM with tf-idf. Nonetheless, this allows us to quantify the performance gain introduced by the process of stopping and stemming using the cosine similarity metric. In our experiments it resulted in performance gains of 9.8% and 28% respectively.

Another important observation is that NCD actually performs *worse* by applying stop word removal to its text over both datasets (see Figure B.1(b) and Figure B.4(b)). Furthermore, NCD performs worse over the Lee dataset for *both* stopping and stemming (see Figure B.4(b)). This is an important result since removing stop words and stemming base words is such a standard procedure in IR and in computing similarity in general.

This suggests that significant improvements can be achieved with computationally simpler similarity metrics by applying lexical-syntactic preprocessing, as described in this chapter, to the data.

## 7.3 Computing document similarity using only Wikipedia

### 7.3.1 Purpose

In this experiment we want to evaluate the performance of the MaxSim and WikiSpread document similarity metrics introduced in Section 4.3.2.

### 7.3.2 Experimental method

The two algorithms were implemented and included in the Python document similarity test suite as discussed in Section 7.2.2. The approach was modularised, starting with identifying salient topics for each document to produce its document concept vector. For this we used an implementation of Milne [67]. These topics were stored to disk and could thus easily be verified.

The next module, computing inter-document similarities, loads the topics and proceeds with computing inter-document similarities as per the approach being tested. These results are cached to disk and can also be inspected.

These methods make use of the spreading activation framework, as developed in Chapter 3. Both these methods rely on several spreading-specific parameters, and we used the parameters that obtained the best results in the inter-concept experiments, as shown for the AA-cos strategy in Table 7.2.

### 7.3.3 Results

The results using the MaxSim method and WikiSpread method are summarised in Table 7.7. Good results were obtained by both methods on the Lee dataset, with MaxSim achieving the best score of $\rho = 0.68$. However, both methods achieved significantly worse results on the Health24 dataset, with WikiSpread achieving slightly better results of $\rho = 0.36$.

**Table 7.7:** Results obtained using the MaxSim method and WikiSpread method for computing inter-document similarity scores over the Health24 and the Lee datasets using Wikipedia only.

| Dataset | MaxSim | WikiSpread |
|---|---|---|
| Health24 | 0.34 | **0.36** |
| Lee & Pincombe | **0.68** | 0.62 |

> Taking tamoxifen to treat breast cancer won't increase your risk of stroke. However, a report on that finding, which appears in the October 20 issue of the Journal of the National Cancer Institute, did conclude that chemotherapy can raise the odds of a brain attack. No link between Tamoxifen and stroke The bottom line from this study is when we looked at women who had a first stroke after their breast cancer diagnosis, we did not see a relationship between tamoxifen and stroke, said study co-author Ann Geiger, group leader in cancer research at Kaiser Permanente Southern California in Pasadena.
> Tamoxifen is a drug sometimes called an anti-oestrogen because it interferes with oestrogen activity that can help breast cancer cells grow.

**Figure 7.5:** An example document from the Health24 corpus showing the niche-specific terms and topics compared to the broader range of topics in the Lee dataset (see Figure 4.3).

This drop in performance between the two datasets can be attributed to two reasons, both relating to the fact that most articles in the Health24 corpus (see for instance Figure 7.5) are somewhat niche-specific and biased towards health-related topics and specialised concepts, which in turn could lead to:

1. *rater bias* (seeing several health-related articles might make raters less sensitive to rate articles as similar even though conceptually they are) as opposed to the Lee dataset which covers a multitude of general news wire topics;

2. less well-defined document concept vectors, since very niche-specific concepts, such as *tamoxifen*, even though they are contained in Wikipedia, might not be as well defined by their link structures as other, more general topics.

## 7.4 Computing document similarity using a combined VSM and Wikipedia-based method

### 7.4.1 Purpose

This experiment tests our second hypothesis (see Section 1.5) where we stated that a keyword-based method (such as the cosine VSM), in combination with a knowledge-based approach (such as the MaxSim Wikipedia-based method) can capture document similarity better than either in isolation.

Simple keyword-based methods cannot accurately capture conceptual relationships between different keywords. Knowledge-based approaches can capture more complex conceptual relationships, however nothing can be said about concepts not defined in the knowledge base. A combination of the two methods is therefore hypothesised to produce more accurate document similarity ratings.

**Table 7.8:** Summary of final document similarity results for the cosine VSM, MaxSim and WikiSpread on their own, and also for the cosine VSM in combination with the better-performing MaxSim method. Results are shown over the Health24 and Lee & Pincombe datasets (described in Chapter 6).

| | **Health24** | **Lee & Pincombe** |
|---|---|---|
| Best VSM | 0.50 | 0.56 |
| MaxSim Method | 0.34 | 0.68 |
| WikiSpread Method | 0.36 | 0.62 |
| **Combined (Cosine + MaxSim)** | **0.53** | **0.72** |

### 7.4.2 Experimental method

The output of the better-performing MaxSim method (see Section 4.3.2.1) was combined with the output of the standard cosine VSM (see Section 2.4.1) in the ratio $\lambda$ and $(1-\lambda)$ respectively, as given by Equation 4.3.3.

By performing a parameter sweep over the range $0 < \lambda < 1$ in increments of 0.1, we can weigh the contributions made by the individual methods, and observe the effect this has on final performance.

The results for the Health24 dataset are shown in Figure 7.6(a) and the results for the Lee dataset are shown in Figure 7.6(b). The final results are summarised in Table 7.8.

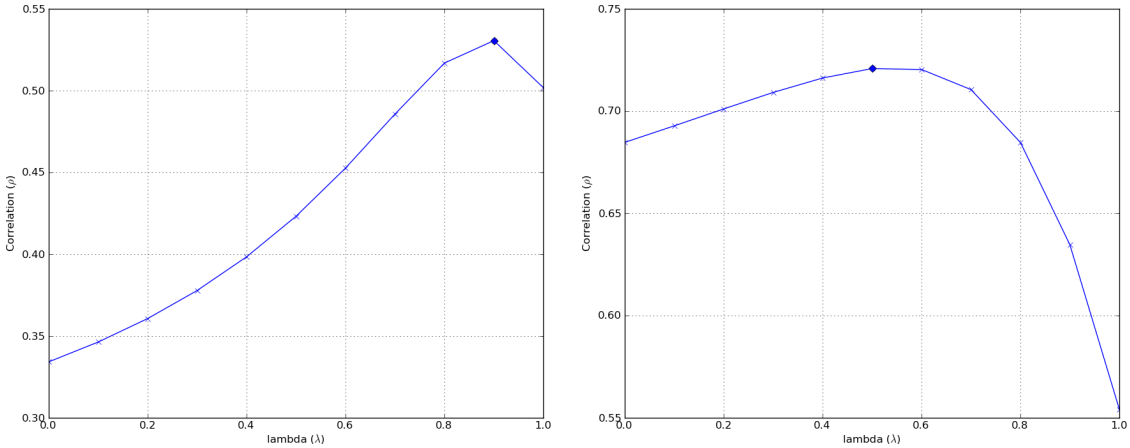
### 7.4.3 Results

Figure 7.6(a) and Figure 7.6(b) show the final correlation values as the contributions from the two different methods are varied. Over the Health24 dataset the best-performing $\lambda = 0.9$ indicates that in that dataset, the augmented cosine metric provides the most valuable contribution to the final score, however the Wikipedia method can increase performance to a final best score for that dataset of $\rho = 0.53$. This makes sense, since the Health24 dataset contains many rare and unique terms. The spreading activation methods rely on a concept being well-defined (linked to other related concepts) by its *link structure*, and many niche terms might not link to many other terms and are thus not that well-defined.

Over the Lee dataset we can see that both methods contribute equally to the best score obtained of $\rho = 0.72$. This is also insightful, since the Lee dataset contains more general topics, compared to the Health24 dataset. This result indicates that over the Lee dataset, the keyword-based technique and the knowledge-based technique contribute equally to the final best performance.

This is a competitive score. Gabrilovich and Markovitch [1] report a similar score of 0.72 over the same Lee dataset using Explicit Semantic Analysis (ESA) (see Section 2.5). ESA involves building an inverted index over the entire Wikipedia, requiring the full Wikipedia database (21GB as of 2009) and significant further preprocessing. In comparison, our approach utilises only the link structure and anchor text (roughly 1GB) of Wikipedia, comparable to the disk space requirements of the Wikipedia Link-based Measure [2], and the use of simple and computationally cheap vector methods.

It would probably be worthwhile to compare ESA and our method based on computational time required. However, implementing the ESA system for this purpose was out




(a) Health24 dataset. (b) Lee & Pincombe dataset.

**Figure 7.6:** Parameter sweep over $\lambda$ showing contributions from cosine VSM ($\lambda$) and the Wikipedia-based MaxSim method ($1 - \lambda$) to the final document similarity performance over the Health24 dataset ($\rho_{\max}$ at $\lambda = 0.9$) and Lee & Pincombe corpus ($\rho_{\max}$ at $\lambda = 0.5$).

of the scope of this project, and since no public reference implementation of this system exists to the best of our knowledge, such a comparison was not possible.

Finally, these results support our initial hypothesis and indicate that combining a keyword-based, statistical method with a knowledge-based approach can increase performance in measuring document similarity.

## 7.5 Computing document similarity using Okapi BM25 and query-likelihood statistical language models

### 7.5.1 Purpose

We want to evaluate how accurately Okapi BM25 (see Section 2.4.2) and query-likelihood statistical language models (see Section 2.4.3) can compute document similarity. These two algorithms are two state-of-the-art models used in Information Retrieval (IR). In Section 7.7, these algorithms are evaluated for providing purely content-based document recommendations. However, in this experiment, we want to evaluate how accurately these models can estimate document similarity.

### 7.5.2 Experimental method

In order to evaluate all similarity metrics fairly, we used the same datasets, namely our proprietary Health24 dataset and the standard Lee & Pincombe dataset. Document similarity ratings from the different algorithms were then correlated with the human gold standards.

Okapi BM25 (see Section 2.4.2) and query-likelihood unigram statistical language models (see Section 2.4.3), were used to generate the document similarity matrices for the two datasets, which were then correlated with the gold ratings using Pearson's linear product-

**Table 7.9:** Correlations over the Lee and Health24 dataset for the cosine similarity metric with tf-idf term weighting, Okapi BM25 probabilistic model and query-likelihood statistical language models using Kullback-Leibler divergence and Dirichlet prior smoothing.

| | **Cosine tf-idf** | **Okapi BM25** | **LMs** |
|---|---|---|---|
| Lee Dataset | 0.49 | 0.16 | 0.11 |
| Health24 Dataset | 0.39 | 0.12 | 0.12 |

moment correlation coefficient. This is the same process that was used to evaluate all other document similarity metrics for this study.

The Lemur [71] implementations were used for Okapi BM25 and unigram language models. As a sanity check, we also used Lemur's implementation of the cosine similarity metric with tf-idf to compare these results with results obtained using our own implementation of the cosine VSM metric as discussed in Chapter 4.

The Okapi BM25 retrieval function has two main tuning parameters $K_1$ and $b$ (see Section 2.4.2). We used a configuration of $K_1 = 1.2$ and $b = 0.75$, which has empirically been shown to produce good results most consistently [29]. For the query-likelihood language models approach we used Dirichlet priors as smoothing method with Kullback-Leibler divergence for measuring similarity which has also been shown to be the best general configuration over several TREC[3] test runs [29].

Table 7.9 summarises the results that were obtained for this experiment.

### 7.5.3 Results

Several important observations can be made from the results in Table 7.9. Firstly, the cosine score obtained using Lemur's implementation of the metric as a sanity check ($\rho = 0.49$ for the Lee dataset and $\rho = 0.39$ for the Health24 dataset), agrees very well with the results we obtained using our own implementation of the cosine metric of $\rho = 0.51$ for the Lee dataset and $\rho = 0.39$ for the Health24 dataset (see *Baseline* score in Table 7.5).

Secondly, results from both Okapi BM25 and language models are significantly worse, compared to the results obtained using the cosine metric.

Thirdly, Okapi BM25 outperforms statistical query-likelihood unigram language models slightly on the Lee dataset ($\rho = 0.16$ versus $\rho = 0.12$), and performs equally badly on the Health24 dataset.

## 7.6 Computing document similarity using Latent Semantic Analysis

### 7.6.1 Purpose

Latent Semantic Analysis (LSA) has been shown to produce accurate document similarity results over the Lee document similarity dataset [3]. We wanted to evaluate the performance of LSA over the Health24 dataset in order to enable a fair comparison to all other document similarity metrics evaluated for this study.

[3] http://trec.nist.gov/

**Table 7.10:** Best correlation results for LSA over the Health24 dataset with different number of factors and with or without backgrounding documents.

| | **Health24 Dataset** |
|---|---|
| LSA, 50 factors, no backgrounding | 0.31 |
| LSA, 150 factors, no backgrounding | 0.48 |
| LSA, 150 factors, backgrounding used | **0.52** |

### 7.6.2 Experimental method

Pincombe and Lee conducted a thorough investigation over the Lee dataset, and reported a top correlation of $\rho = 0.5988$ [3]. In order to evaluate LSA over our proprietary Health24 dataset, we implemented an LSA-based similarity measure using the Python `Divisi` package [76].

To also enable a fair comparison with the best results obtained using the pure cosine VSM method (shown in Table 7.5), stopwords were removed and word stemming was applied. This is the same preprocessing setup which achieved the best results using the pure cosine method.

LSA performance increases as it indexes more documents, since it uses term co-occurrence counts (how often words appear in the same context) as an indication of relatedness between terms [3]. Therefore, we conducted a second test which included 300 additional documents from the Health24 corpus as backgrounding data.

LSA also relies on the use of an appropriate number of *factors* (dimensions) to reconstruct the term-document matrix following singular value decomposition [77]. Generally, using more factors leads to increased performance [3].

The results for this experiment are summarised in Table 7.10.

### 7.6.3 Results

Firstly, we see that increasing the number of factors leads to a higher correlation performance, which is consistent with the literature [3].

Secondly, we see that using additional backgrounding documents further increases correlation to a best score of Pearson $\rho = 0.52$. This is a slight improvement ($\approx 4\%$) over the best score of $\rho = 0.50$ obtained over this dataset for the pure cosine similarity evaluated in Section 7.2.

## 7.7 Using information retrieval models for news-article recommendation

### 7.7.1 Purpose

As a use case for document similarity metrics, a selection of IR document similarity algorithms were implemented into a purely content-based news-article recommender system for the online media company Health24 (see Chapter 5). This served both as a proof-of-concept for using document similarity measures in providing purely content-based recom-

mendations, and as a means for collecting human ratings from professional journalists about how useful such a system could be.

In this experiment we use the ratings by the journalists to measure how successfully these algorithms can be used to recommend related news articles to users.

### 7.7.2 Experimental method

As described in Section 5.3.1.6, the Health24 recommender is equipped with a per-recommendation feedback feature where journalists can rank the recommendations produced by the system. These ratings are collected by the service for evaluation purposes.

Ratings fall in the range 1, 2, or 3. 1 means 'Not relevant', 2 'Neutral or Undecided' and 3 means 'Relevant'. There is a rich tradition of IR evaluation measures in the literature, and we discuss these and our results in the next two sections.

#### A note about IR evaluation metrics

The *de facto* modus operandi for testing the effectiveness of a retrieval algorithm in information retrieval is by calculating the precision and recall values [43].

Given a query, these measures assume that there is a set of documents that is retrieved and a set of documents that is not retrieved (the remaining indexed documents). We define $A$ as the relevant set of documents for the query, $\bar{A}$ as the non-relevant set, $B$ as the total set of retrieved documents and $\bar{B}$ as the set of documents that are not retrieved. Furthermore, as usual, the $\cap$ operator denotes the intersection of two sets. Therefore, $A \cap B$ denotes the set of documents both relevant *and* retrieved.

With these definitions in place, we can define

$$\text{Precision} = \frac{|A \cap B|}{|A|}, \text{ and} \tag{7.7.1}$$

$$\text{Recall} = \frac{|A \cap B|}{|B|}. \tag{7.7.2}$$

In words: Precision is the ratio of relevant documents in the returned set to the total number of documents in the returned set. In other words, the fraction of the returned documents that are relevant. Recall is the ratio of relevant documents returned to the *total* number of relevant documents in the corpus.

Note that these measures implicitly assume that the retrieval task involves returning as many of the total number of relevant documents in the corpus, and as little of the non-relevant documents as possible. For our purposes, each recommendation-set contained a fixed number of documents which were usually much less than the total number of relevant documents to the given focus article. Recall is therefore not suitable for our purposes.

Instead, we calculated precision values. More specifically, we calculated the mean average uninterpolated precision (MAP) scores. MAP is the most widely used effectiveness measure in the IR literature [43], and can be seen as the mean of the averages of the precision values, measured at the rank positions of all *relevant* documents.

To illustrate this concept, consider a retrieved set of recommendations, documents $d_1, d_2, ..., d_j$. For this recommendation set, a set of *relevance ratings* are produced, $r_1, r_2, ..., r_j$.

**Table 7.11:** Example of calculating precision and average precision values.

| Retrieved Document | Relevance Score | Precision | Average Precision |
|---|---|---|---|
| $d_1$ | $r_1 = 1$ | $\frac{1}{1} = 1.0$ | 1.0 |
| $d_2$ | $r_2 = 0$ | $\frac{1}{2} = 0.5$ | 1.0 |
| $d_3$ | $r_3 = 1$ | $\frac{2}{3} = 0.667$ | $\frac{(1+0.667)}{2} = 0.835$ |
| $d_4$ | $r_4 = 1$ | $\frac{3}{4} = 0.75$ | $\frac{(1+0.667+0.75)}{3} = 0.81$ |
| $d_5$ | $r_5 = 0$ | $\frac{3}{5} = 0.6$ | 0.81 |

**Table 7.12:** MAP performance scores (not to be confused with correlations scores) obtained from Health24 journalists for the three main IR retrieval models.

| Retrieval Model | MAP |
|---|---|
| Cosine VSM using tf-idf | 0.70 |
| Okapi (BM25) | **0.74** |
| Language Models (KL divergence) | 0.73 |

In any given set of recommendations, we followed the standard approach of ignoring any documents for which we did not receive ratings.

To calculate the MAP score, the average precisions for all the queries is first calculated. Table 7.11 illustrates the process of calculating average precision. As can be seen, it simply involves computing the average of the precisions at all the rank positions where *relevant* documents are returned, i.e. non-relevant queries do not contribute to it.

MAP is then simply the arithmetic average of all calculated average precision values for the system under evaluation.

#### Processing the ratings

As noted, relevance ratings were collected as three discrete values, 1 denoting non-relevant, 2 denoting neutral or undecided, and 3 denoting a relevant recommendation. Every recommendation the system issued over the evaluation period of two months was stored in the database under a unique `recId` (recommendation session identifier).

Relevance feedback was returned with an associated `recId`, which was used to determine the initial order in which recommendations were presented, since document *ranks* are used to determine precisions at a specified rank. This is used in turn to calculate MAP scores and to discard unrated recommendations.

MAP scores for the individual retrieval models are presented in Table 7.12. Since precision is calculated on binary scores of relevance, a neutral score of 2 was randomly assigned to either 1 (relevant), or 0 (non-relevant), using a uniform distribution.

### 7.7.3 Results

MAP scores are not to be confused with correlation scores. MAP scores can be seen as an indication of the average number of recommendations that raters (the journalists) found

**Table 7.13:** Best correlation results for all similarity metrics that were evaluated over the Health24 dataset and the Lee dataset (best results in bold).

| Similarity metric | Health24 Dataset | Lee Dataset |
|---|---|---|
| NCD (bzip2) | 0.27 | 0.37 |
| Cosine tf-idf | 0.50 | 0.57 |
| MaxSim method Only | 0.34 | 0.68 |
| WikiSpread method Only | 0.36 | 0.62 |
| Combined (Cos+MaxSim) | **0.53** | **0.72** |
| Okapi BM25 | 0.12 | 0.16 |
| LMs | 0.12 | 0.11 |
| LSA | 0.52 | 0.59 [16] |

useful using that particular retrieval model.

Table 7.12 shows, firstly, that average ratings were quite high, indicating that journalists found a large ratio of recommendations useful (or 'relevant'). Secondly, it shows the Okapi probabilistic retrieval method to achieve the best MAP score of 0.74, closely followed by language models using KL-divergence, and cosine VSM using tf-idf performing worst in this evaluation, with a MAP score of 0.70. This is consistent with results obtained by other comparable studies [78].

## 7.8 Chapter summary

This chapter discussed several experiments which were conducted for this study. The spreading activation model for computing conceptual similarity, as it was introduced in Chapter 3, was evaluated by conducting inter-concept similarity experiments over the WordSimilarity-353 word-pair similarity dataset [33].

The effects of applying different preprocessing techniques prior to computing document similarity were evaluated and discussed.

Experiments are also conducted to compare our proposed methods for computing document similarity with several other state-of-the-art methods. Table 7.13 presents a summary of the best results, as Pearson linear product-moment correlations, for all the similarity metrics that were evaluated in this study.

Lastly, the results obtained from the news-article recommender service that was designed and implemented for Health24.com and discussed in Chapter 5 were shown.

More significant results are reiterated and discussed in more detail in Chapter 8.

# Chapter 8

# Conclusion

This thesis investigated the problem of computing semantic similarity or relatedness between two natural-language documents. In the following section we briefly recap the main findings of this study. We offer final concluding thoughts based on these results and how this sheds light on the initial problem statement. Lastly, suggestions are made for further research in this area.

## 8.1 Summary of findings

Available approaches for computing document similarity can broadly be divided into three categories (our classification):

I Purely algorithmic approaches which rely on no external or background data and uses simply the text in the individual documents to compute document similarity;

II approaches which rely on large collections of unstructured text; and

III approaches which make use of background knowledge encoded in knowledge bases to compute similarity.

In this thesis we proposed a novel technique pertaining to the third category identified above, for computing similarity or relatedness between concepts and documents. This technique relies on modelling the hyperlink structure of Wikipedia as an associative network of concepts, and computes similarity between two concepts by iteratively spreading activation energy over the links connecting concepts to the rest of the network. An associative network uses edge weights between nodes to represent how strongly one node is associated with the other, and since this does not exist for Wikipedia, we introduced and tested three new *weighting schemes* to estimate these weights. We also introduced and tested three *spreading strategies* for computing relatedness between two concepts based on the result of the spreading process.

We furthermore evaluated how preprocessing text can improve results using two metrics pertaining to the first category identified above, namely the Vector Space Model (VSM) using cosine similarity with term-frequency inverse document-frequency, and the Normalised Compression Distance (NCD).

Lastly, we looked at content-based recommendation of news articles as a use case for document similarity metrics. The performance of three standard Information Retrieval (IR) models were evaluated in this context.

The following is a brief synopsis of the main findings of this thesis, with concluding thoughts for each:

**Spreading activation**

> *The Agglomerative Approach spreading strategy (AA) outperforms the Target Activation Approach spreading strategy (TAA), and of the two agglomerative spreading strategies – cosine and Wikipedia Link-based Measure (WLM) – the cosine method outperforms the WLM based method.*

These results suggest that in the Wikipedia associative network, relatedness between concepts can more accurately be computed by comparing the two concepts' activation vectors[1] than by simply considering the direct paths which exist between the two concepts (as used by the TAA).

Furthermore, the cosine agglomerative spreading strategy computes relatedness based on the individual levels of activation of nodes in the activation vectors. The WLM-based agglomerative strategy, on the other hand, discards activation values and computes similarity based on the ratio of nodes that are jointly activated from both nodes, to the total number of nodes activated, regardless of their respective levels of activation.

Better performance on the cosine spreading strategy suggests that the real-valued level of activation that reaches nodes connected to a concept is more indicative of semantic relatedness in the Wikipedia associative network than simply knowing which concepts it is connected to.

> *The Inverse Link Frequency weighting scheme (ILF) weighting strategy outperforms both pure Energy Distribution weighting scheme (ED) and Normalised Inverse Link Frequency (NILF).*

In the Spreading Activation (SA) model we presented, these activation values are regulated by the weighting scheme in use. The SA algorithm relies on real-valued edge weights to determine how strongly one concept node is associated with another. Since these values do not exist for the Wikipedia hyperlink graph, we proposed the use of three new strategies for estimating these weights, namely pure ED, ILF and NILF (see Section 3.4).

The AA-cos spreading strategy takes into account the levels of activation of nodes when computing relatedness, and the values of activation in turn are regulated by the edge weights. We can therefore use the AA-cos results for the different weighting schemes to give an indication of which strategy provides the best estimations for these weights. We evaluated this and it was found that ILF outperforms both ED and NILF.

ILF is based on the term-frequency inverse document-frequency concept (as discussed in Section 2.4.1) whereby the discriminatory power (weight) of a term decreases logarithmically as it is contained in more documents in the corpus – in our case, a node's weight in the network decreases logarithmically as it connects to more nodes in the network.

[1]The set of all concept nodes that can be activated from a specific node, by spreading activation up to a certain maximum path length.

The ILF weighting strategy is based on the assumption that more specific concept nodes have a lower level of connectedness in an associative network, and that paths that connect two nodes through these more specific nodes are more *indicative of a semantic relationship* than paths through more general nodes with higher degrees of connectedness. In all our experiments the ILF weighting scheme gave the best results, which leads us to conclude that this is a valid assumption and a valid approach (or at least starting point) for computing edge weights.

> *Using a longer path length in the spreading activation experiments produces more accurate results than using shorter path lengths.*

The use of spreading activation was predicated on the hypothesis that the Wikipedia hyperlink graph structure resembles an associative network, where nodes can be related to one another via connections of one or more links. We tested this hypothesis and showed that using connections of longer than one hop between two concepts lead to more accurate results.

These results empirically validate our initial hypothesis, and lead us to conclude that the model for spreading activation, as we have developed and presented in this thesis, can be used to process the hyperlink structure of Wikipedia as an associative network to accurately compute relatedness or similarity between the concepts represented by the individual articles they represent.

### Effects of preprocessing on document similarity performance

> *Augmenting a document's term space with additional preprocessed terms (the Term Space Augmentation (TSA) model), gives better results than simply replacing a document's terms with preprocessed terms (the Term Space Substitution (TSS) model) in all our experiments. Furthermore, the cosine VSM with tf-idf outperforms NCD in all experiments.*

It is common practise in IR to perform preprocessing on the text of a document before computing relatedness to other documents or queries. Preprocessing typically involves removing high-frequency low-content words such as 'a', 'an', 'the', etc., and reducing words to their base forms (via stemming).

We introduced and evaluated a second approach, the Term Space Augmentation approach, which preprocesses text using processes of case-normalisation (converting all text to lower-case), stop word removal, word stemming, etc., and then *appends* these processed terms to the original document. Since category I and II algorithms such as the VSM and NCD methods as introduced above, do not use any background information, the hypothesis is that this process of appending linguistically processed terms adds more linguistic information to the document, which can result in increased performance.

In all experiments it was found that the TSA process increased performance. In the VSM, the performance increase was slight compared to the standard method of preprocessing text. However, in the NCD the improvement was substantial, namely 54% and 80% respectively over the two datasets.

Nonetheless, the cosine VSM significantly outperforms the NCD in all experiments. Furthermore, it also significantly outperforms the Okapi BM25 method and query-likelihood statistical language models in computing similarity between natural-language documents.

Based on this data, we therefore conclude that of the four category I and II algorithms that were evaluated – NCD, cosine VSM using tf-idf, Okapi BM25 and statistical LMs – the VSM methods provide the most accurate inter-document similarity ratings as compared to human ratings.

### Wikipedia-based inter-document similarity

> *Wikipedia-based methods using SA significantly outperform the VSM methods on the Lee dataset. However, over the Health24 dataset the VSM methods outperform the Wikipedia-based methods. Furthermore, combining the best-performing Wikipedia-based method with the VSM model leads to best results over both datasets.*

The documents in the Lee document-similarity dataset [16] cover a wide range of general topics which are well-defined in Wikipedia. 'Well-defined' in this sense refers to the links connecting a concept to other concepts in the network, since the SA algorithm computes relatedness between concepts based on how those concepts are connected to other concepts. The more mature a topic is in Wikipedia, the more links are created to and from related concepts – we call this a 'well-defined' topic. The proprietary Health24 dataset, on the other hand, is largely related to medical topics, and contains several niche terminology which, although contained in Wikipedia, are much less well-defined compared to the Lee dataset.

We noted in the first chapter that a knowledge-based approach, such as the Wikipedia-based methods, can capture more complex relationships between concepts mentioned in a document, even when such concepts share no explicit words or other lexical features, since such relationships are contained in the underlying knowledge base – in this case, Wikipedia. However, such a knowledge-based algorithm can say nothing about concepts not present in its knowledge base. On the other hand, VSM methods and other category I and II methods (following our classification above) can match any keywords which are common between two documents, regardless of the keyword. We therefore hypothesised that a combination of two such methods will produce more accurate results than either in isolation.

We tested this hypothesis and showed that a combined method does improve performance over both the Lee dataset and the Health24 dataset. Furthermore, we tested several ratios in which to combine the relatedness scores from the two methods, to see which ratios led to the best overall performance. In the best configuration over the more well-defined Lee dataset, the knowledge-based method and the keyword-matching method contributes 50% each to the final score of relatedness. However, over the less well-defined Health24 dataset, the keyword-matching method contributes 90% and the SA method only 10%.

These results indicate that over the two datasets we evaluated, a keyword-matching method such as the VSM can capture 50% and 90% respectively of the conceptual similarity between two documents, as measured by correlation with human ratings. The addition of a knowledge-based method provides increased accuracy. However, the weight of the contribution of such a knowledge based method must be carefully adjusted to reflect the range of topics in the corpus that are contained in the underlying knowledge base.

**Content-based article recommendation**

> *Okapi BM25 outperforms both query-likelihood statistical language models and the VSM in providing content-based news article recommendations.*

All three retrieval models achieved high mean average uninterpolated precision (MAP) scores, however the probabilistic Okapi BM25 method achieved marginally better scores. This result is interesting, since in our purely document-similarity experiments, the VSM outperforms Okapi BM25 and LMs, which perform poorly. However in providing related news articles, these methods score better than the VSM.

It is our opinion that although news-article recommendation is a specific use case of document similarity metrics, the context is slightly different. In a pure document-similarity setting, we measure the ability of the system to rank documents based on how identical their content are. However, in a document recommender setting, users might not want identical content, rather, they might want content about the same topic, but which contains new information.

In a news-article recommender setting, these results suggest that a probabilistic method such as Okapi BM25 (which combines term frequency information into the probabilistic model of relevance, see Section 2.4.2) provides better results than either LMs or the VSM.

## 8.2 Suggestions for future research

Spreading activation was found to be a viable method for computing inter-concept similarity using Wikipedia, however there are many areas on which future work can improve. It might be worthwhile to investigate the possible use of heuristics to reduce the number of paths that need to be considered. Moreover, it might be worthwhile to investigate using a machine learning approach to identify possibly rewarding paths as a heuristic for an A* algorithm to explore these paths first.

Term space modification was found to increase performance in both the VSM and using NCD, however we only tested five preprocessing procedures, and we feel there is still scope left to explore the use of this technique to increase document-similarity performance by testing other available preprocessing procedures such as noun phrase chunking for grouping phrases into their *conceptual* constituents instead of matching them at a token-by-token-basis. For instance, this would transform 'the master of ceremonies' $\Leftrightarrow$ 'the master_of_ceremonies', which could provide a more conceptual match than matching at the word level.

In content-based document recommendation better results were achieved by using methods which scored lower in a purely document-similarity setting (Okapi BM25 and LMs). It might be fruitful to research this problem further, in order to develop better content-based document recommender algorithms.

# Appendices

# Appendix A

# WordSimilarity-353 Concept Pairs

**Table A.1:** The list of fifty WS-353 concept-pairs (1–25 shown) and their human-assigned ratings which were used for the inter-concept similarity experiments.

| Word 1 | Word 2 | $\bar{r}$ | $r_1$ | $r_2$ | $r_3$ | $r_4$ | $r_5$ | $r_6$ | $r_7$ | $r_8$ | $r_9$ | $r_{10}$ | $r_{11}$ | $r_{12}$ | $r_{13}$ | $r_{14}$ | $r_{15}$ | $r_{16}$ |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| FBI | fingerprint | 6.94 | 8 | 6 | 8 | 5 | 5 | 9 | 7 | 7 | 6 | 6 | 6 | 8 | 9 | 7 | 6 | 8 |
| Mars | scientist | 5.63 | 8 | 1 | 7 | 4 | 6 | 8 | 1 | 6 | 5 | 6 | 2 | 9 | 7 | 5 | 7 | 8 |
| Mexico | Brazil | 7.44 | 8 | 8.5 | 8 | 6 | 7 | 7 | 8 | 9 | 7 | 7.5 | 7 | 8 | 10 | 2 | 8 | 8 |
| announcement | news | 7.56 | 8 | 7 | 8 | 8 | 8 | 9 | 8 | 6 | 6 | 8 | 8 | 8 | 6 | 8 | 9 | 6 |
| arrangement | accommodation | 5.41 | 5 | 6 | 7 | 6 | 4 | 4 | 4 | 6 | 3 | 7.5 | 5 | 5 | 7 | 6 | 7 | 4 |
| arrival | hotel | 6.00 | 7 | 8 | 6.5 | 1 | 6 | 9 | 7 | 5 | 6 | 7.5 | 4 | 6 | 7 | 4 | 6 | 6 |
| attempt | peace | 4.25 | 7 | 4 | 7 | 2 | 7 | 4 | 2 | 3 | 3 | 4 | 3 | 5 | 5 | 4 | 5 | 3 |
| benchmark | index | 4.25 | 5 | 5 | 2 | 2 | 7 | 4 | 5 | 7 | 3 | 1 | 7 | 2 | 6 | 4 | 5 | 3 |
| board | recommendation | 4.47 | 6 | 4 | 2 | 1 | 7 | 8 | 3 | 4 | 1 | 7.5 | 2 | 7 | 7 | 0 | 7 | 5 |
| boxing | round | 7.61 | 6 | 8.25 | 7.5 | 7 | 8 | 7 | 8 | 9 | 4 | 8 | 7 | 8 | 10 | 9 | 9 | 6 |
| century | year | 7.59 | 8 | 9 | 8 | 9 | 7 | 6 | 8 | 8 | 7 | 7.5 | 7 | 9 | 8 | 5 | 9 | 6 |
| competition | price | 6.44 | 7 | 8 | 7.5 | 5 | 6 | 7 | 8 | 4 | 6 | 7.5 | 5 | 6 | 9 | 5 | 9 | 3 |
| computer | news | 4.47 | 5 | 1 | 7 | 6 | 5 | 5 | 1 | 4 | 6.5 | 4 | 2 | 5 | 3 | 7 | 6 | 4 |
| concert | virtuoso | 6.81 | 8 | 6 | 7.5 | 9 | 6 | 7 | 7 | 8 | 2 | 7.5 | 7 | 7 | 8 | 3 | 8 | 8 |
| credit | information | 5.31 | 7 | 5 | 2 | 3 | 7 | 4 | 1 | 5 | 7 | 1 | 2 | 8 | 10 | 9 | 8 | 6 |
| day | summer | 3.94 | 7 | 7 | 2 | 1 | 4 | 3 | 1 | 7 | 3 | 4 | 4 | 4 | 6 | 1 | 5 | 4 |
| death | inmate | 5.03 | 4 | 5 | 7.5 | 1 | 5 | 4 | 7 | 3 | 6 | 2 | 6 | 6 | 3 | 8 | 8 | 5 |
| discovery | space | 6.34 | 8 | 2 | 7.5 | 9 | 5 | 7 | 4 | 7 | 5 | 6 | 5 | 7 | 8 | 8 | 7 | 6 |
| dividend | payment | 7.63 | 6 | 9 | 7 | 4 | 9 | 8 | 8 | 8 | 7 | 8 | 7 | 8 | 8 | 9 | 9 | 7 |
| dollar | profit | 7.38 | 8 | 6 | 9 | 9 | 9 | 7 | 7 | 8 | 8 | 6 | 7 | 5 | 6 | 7 | 8 | 8 |
| energy | secretary | 1.81 | 1 | 0 | 4 | 2 | 4 | 5 | 1 | 1 | 1 | 0 | 1 | 1 | 4 | 0 | 2 | 2 |
| experience | music | 3.47 | 6 | 1 | 7 | 1 | 5 | 3 | 1 | 1 | 1 | 7.5 | 3 | 8 | 3 | 1 | 5 | 2 |
| family | planning | 6.25 | 7 | 5 | 7 | 3 | 7 | 8 | 4 | 6 | 7 | 8 | 7 | 6 | 8 | 1 | 8 | 8 |
| fighting | defeating | 7.41 | 8 | 8 | 8 | 8 | 5 | 7 | 6 | 9 | 7 | 7.5 | 8 | 9 | 8 | 5 | 9 | 6 |
| focus | life | 4.06 | 5 | 0 | 7.5 | 1 | 7 | 6 | 0 | 3 | 4 | 7.5 | 3 | 2 | 2 | 7 | 5 | 5 |
| game | team | 7.69 | 8 | 8.5 | 8.5 | 8 | 5 | 8 | 6 | 9 | 7 | 9 | 7 | 7 | 9 | 8 | 8 | 7 |

**Table A.2:** The list of fifty WS-353 concept-pairs (26–50 shown) and their human-assigned ratings which were used for the inter-concept similarity experiments.

| Word 1 | Word 2 | $\bar{r}$ | $r_1$ | $r_2$ | $r_3$ | $r_4$ | $r_5$ | $r_6$ | $r_7$ | $r_8$ | $r_9$ | $r_{10}$ | $r_{11}$ | $r_{12}$ | $r_{13}$ | $r_{14}$ | $r_{15}$ | $r_{16}$ |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| gender | equality | 6.41 | 8 | 6 | 5 | 4 | 4 | 8 | 5 | 8 | 6 | 7.5 | 2 | 8 | 8 | 7 | 7 | 9 |
| governor | interview | 3.25 | 4 | 0 | 4 | 1 | 7 | 6 | 0 | 5 | 0 | 2 | 3 | 5 | 3 | 3 | 4 | 5 |
| grocery | money | 5.94 | 8 | 5 | 7.5 | 2 | 7 | 7 | 7 | 5 | 6.5 | 6 | 6 | 8 | 6 | 2 | 5 | 7 |
| hundred | percent | 7.38 | 9 | 5 | 9 | 3 | 10 | 9 | 6 | 7 | 7 | 6 | 6 | 7 | 10 | 9 | 8 | 7 |
| image | surface | 4.56 | 7 | 1 | 5 | 1 | 1 | 5 | 4 | 3 | 4 | 4 | 5 | 5 | 7 | 7 | 8 | 6 |
| life | death | 7.88 | 9 | 9.5 | 9.5 | 5 | 10 | 8 | 8 | 8 | 6.5 | 7.5 | 8 | 9 | 10 | 0 | 9 | 9 |
| line | insurance | 2.69 | 5 | 2 | 2 | 1 | 6 | 3 | 3 | 4 | 1 | 2 | 2 | 2 | 0 | 0 | 9 | 1 |
| lover | quarrel | 6.19 | 7 | 2 | 8.5 | 3 | 7 | 7 | 6 | 9 | 5 | 7.5 | 4 | 6 | 8 | 7 | 7 | 5 |
| man | woman | 8.30 | 8 | 9.75 | 9 | 10 | 9 | 9 | 8 | 9 | 7.5 | 7.5 | 9 | 9 | 10 | 1 | 9 | 8 |
| minister | party | 6.63 | 8 | 7.5 | 8 | 7 | 6 | 7 | 8 | 8 | 5 | 7.5 | 6 | 1 | 8 | 8 | 8 | 3 |
| news | report | 8.16 | 9 | 6 | 8.5 | 8 | 7 | 9 | 7 | 8 | 7 | 8 | 7 | 9 | 10 | 9 | 9 | 9 |
| opera | industry | 2.63 | 7 | 0 | 1 | 1 | 7 | 2 | 3 | 3 | 2 | 2 | 1 | 2 | 3 | 1 | 4 | 3 |
| peace | atmosphere | 3.69 | 6 | 5 | 5 | 1 | 0 | 6 | 3 | 5 | 2 | 6 | 1 | 3 | 3 | 0 | 7 | 6 |
| preservation | world | 6.19 | 7 | 8 | 7 | 6 | 7 | 7 | 2 | 8 | 4 | 6 | 6 | 8 | 8 | 4 | 6 | 5 |
| problem | challenge | 6.75 | 7 | 7.5 | 7.5 | 8 | 7 | 7 | 1 | 10 | 5 | 6 | 6 | 8 | 8 | 8 | 7 | 5 |
| reason | hypertension | 2.31 | 4 | 1 | 1 | 2 | 6 | 1 | 0 | 1 | 3 | 2 | 2 | 7 | 0 | 2 | 3 | 2 |
| record | number | 6.31 | 8 | 6 | 8 | 5 | 7 | 7 | 3 | 4 | 5 | 8 | 5 | 8 | 8 | 5 | 8 | 6 |
| shower | flood | 6.03 | 8 | 7 | 8.5 | 6 | 7 | 8 | 2 | 7 | 6 | 6 | 7 | 8 | 8 | 0 | 4 | 4 |
| start | match | 4.47 | 5 | 2 | 6.5 | 1 | 9 | 3 | 3 | 8 | 1 | 2 | 3 | 5 | 5 | 9 | 6 | 3 |
| start | year | 4.06 | 5 | 5.5 | 6.5 | 2 | 5 | 3 | 0 | 6 | 4 | 6 | 2 | 2 | 3 | 9 | 4 | 2 |
| sugar | approach | 0.88 | 3 | 0 | 0 | 1 | 5 | 1 | 0 | 1 | 0 | 0 | 1 | 0 | 0 | 0 | 1 | 1 |
| television | film | 7.72 | 8 | 5 | 8.5 | 8 | 9 | 8 | 8 | 9 | 5 | 8 | 7 | 8 | 9 | 9 | 8 | 6 |
| theater | history | 3.91 | 5 | 6 | 6 | 4 | 5 | 3 | 0 | 3 | 6.5 | 6 | 1 | 5 | 1 | 1 | 7 | 3 |
| victim | emergency | 6.47 | 8 | 7 | 7.5 | 4 | 5 | 6 | 6 | 7 | 6 | 4 | 5 | 6 | 9 | 9 | 9 | 5 |

# Appendix B

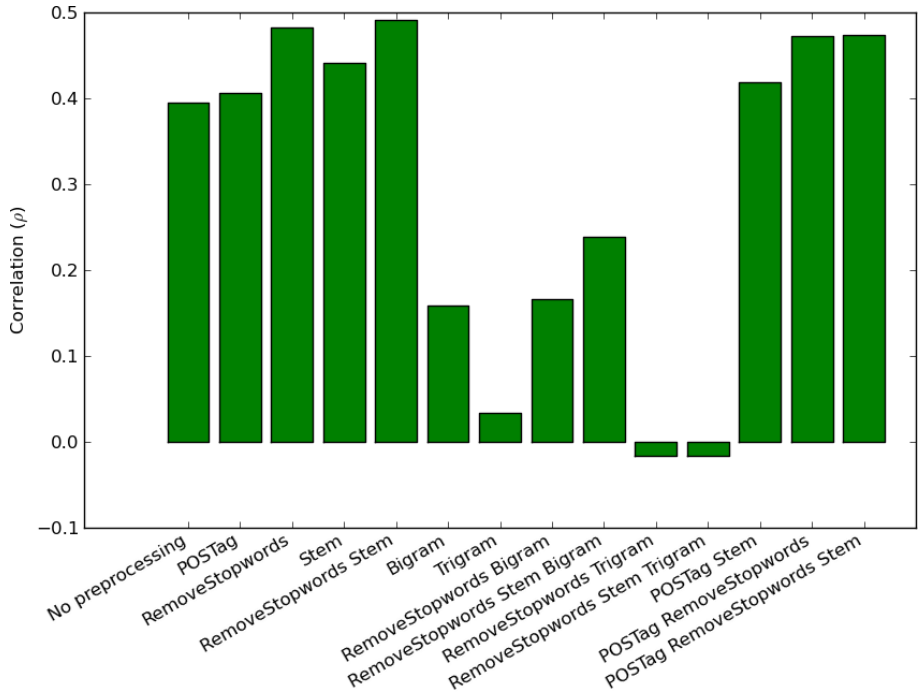
# Results of document preprocessing experiments

In the following figures, we present the results obtained for the document preprocessing experiments discussed in Section 7.2.

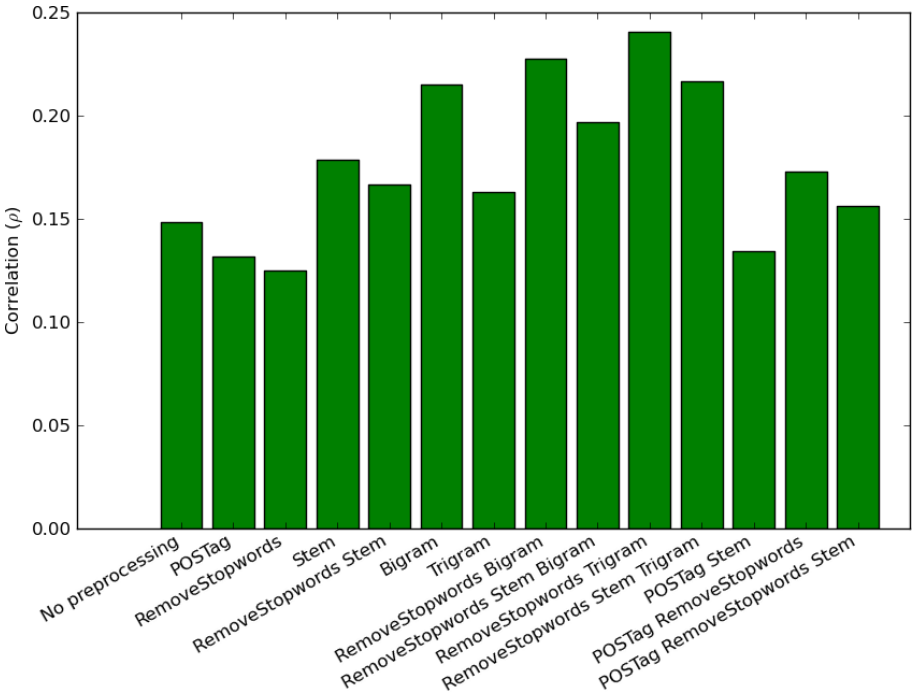
Please refer to Table B.1 for the meaning of the preprocessing codes underneath each figure. All correlations are the Pearson linear product-moment correlation.

**Table B.1:** Different preprocessing techniques and the codes they are represented by.

| **Code** | **Meaning** |
|---|---|
| No preprocessing | raw document text used |
| RemoveStopwords | stopwords removed |
| Stem | words stemmed |
| Bigram | terms converted to bigrams, e.g. ‘‘`she said so`’’ → ‘‘`she_said said_so`’’ |
| Trigram | terms converted to trigrams, e.g. ‘‘`the party was great`’’ → ‘‘`the_party_was party_was_great`’’ |
| POSTag | part-of-speech tags appended to terms, e.g. ‘‘`the car`’’ → ‘‘`the_ART car_NN`’’ |

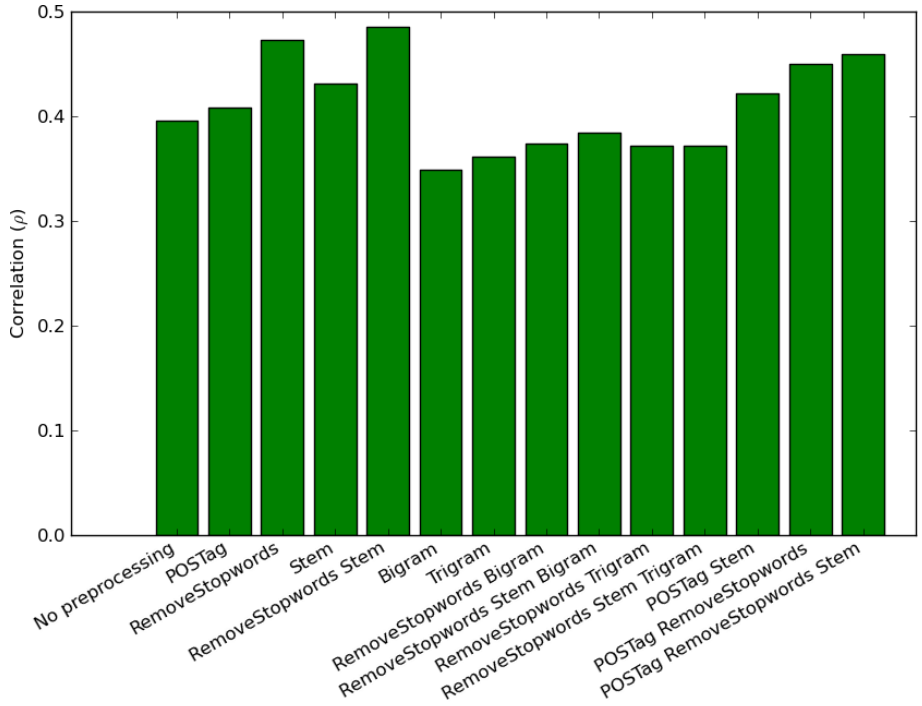



(a) Cosine similarity metric.

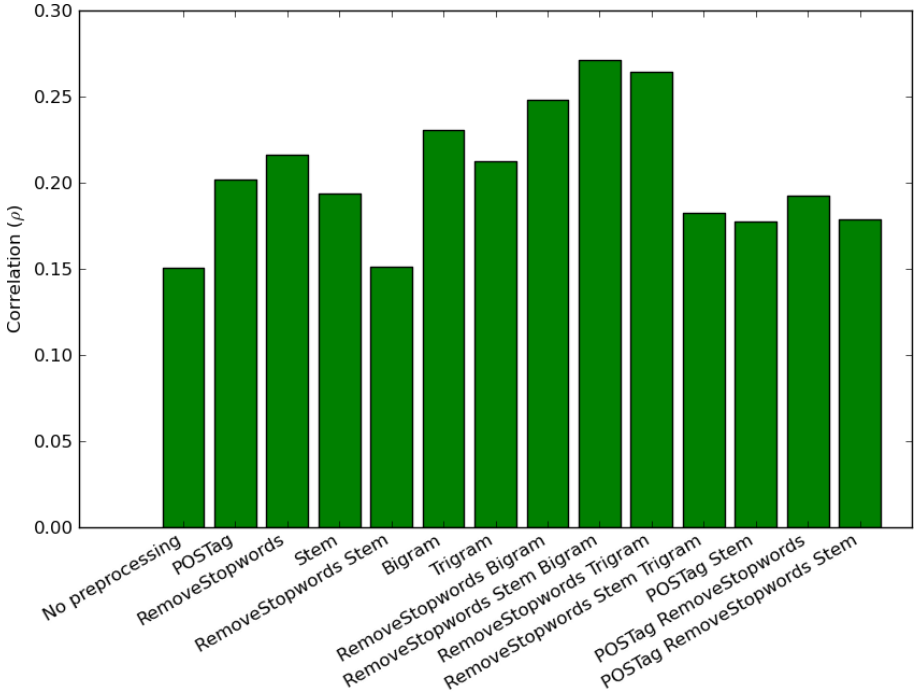



(b) Normalised Compression Distance.

**Figure B.1:** Performance of cosine VSM and NCD metrics using Term Space Substitution (TSS) model over Health24 corpus.

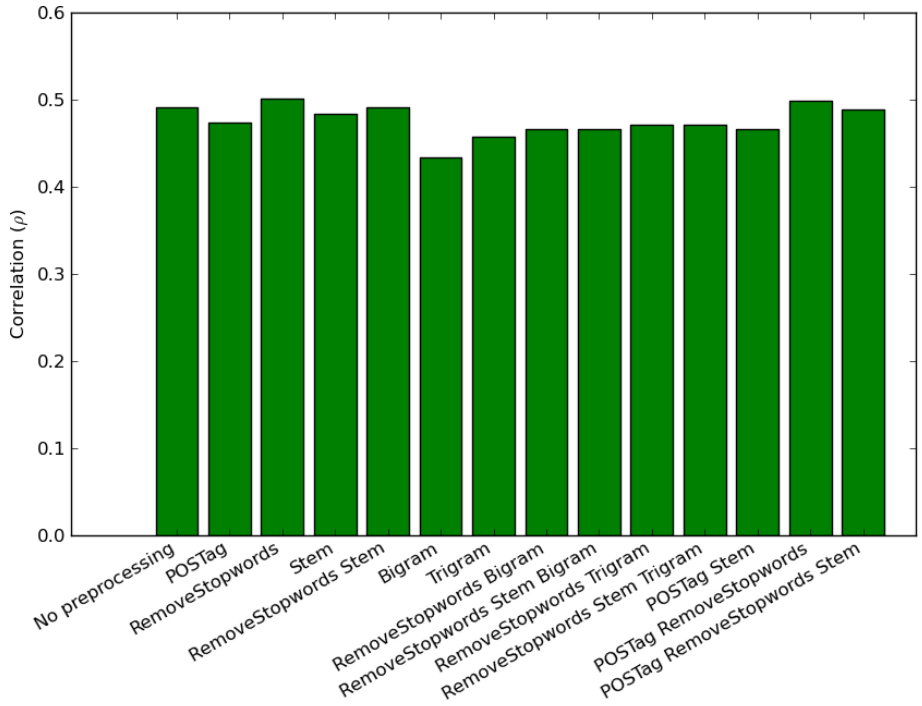



(a) Cosine similarity metric.

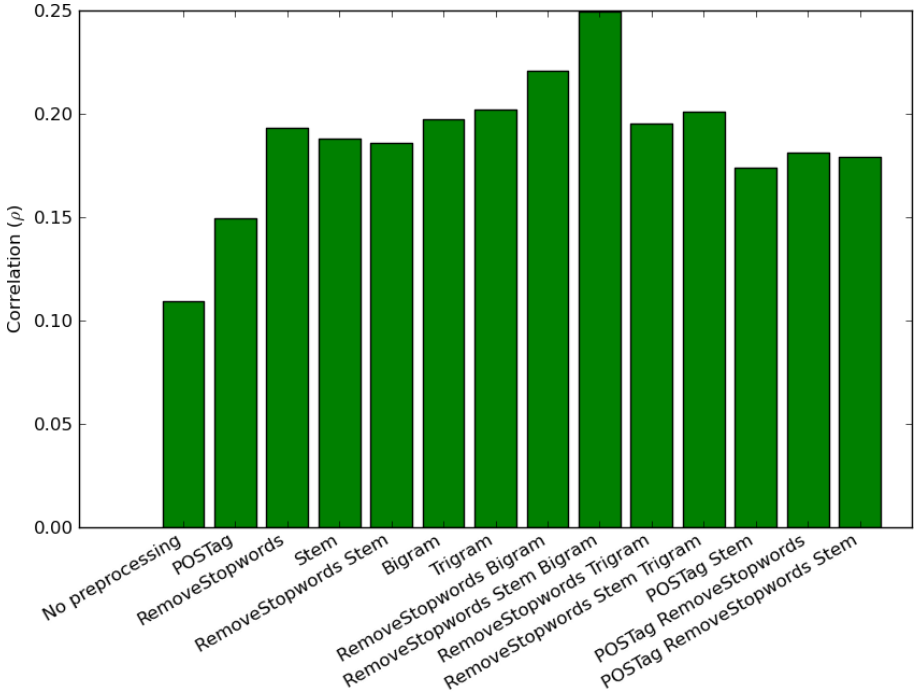



(b) Normalised Compression Distance.

**Figure B.2:** Performance of cosine and NCD metrics using Term Space Augmentation (TSA) model over Health24 corpus.

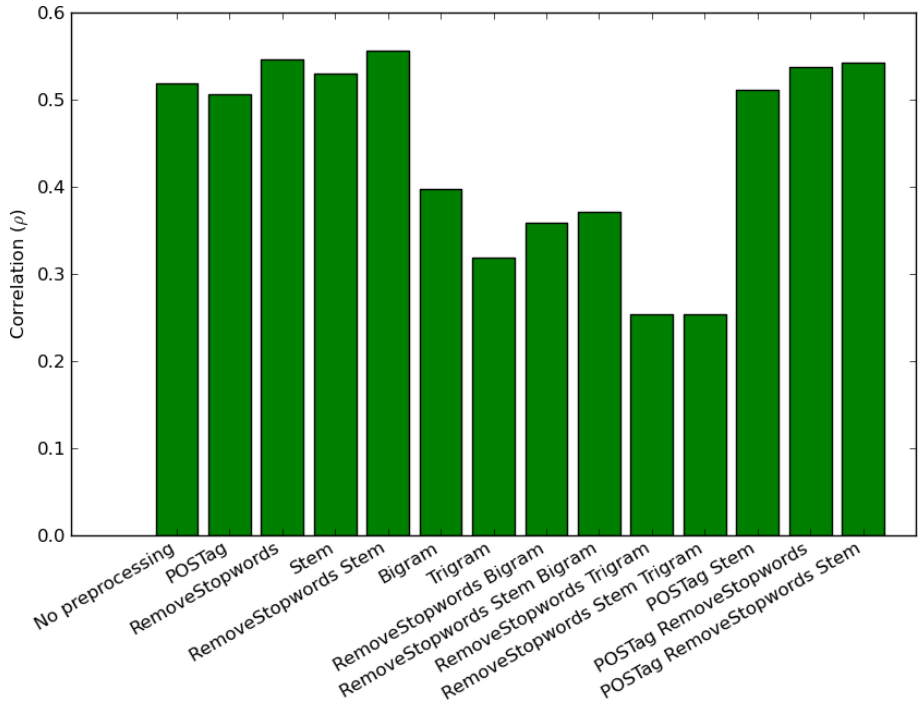



(a) Cosine similarity metric.

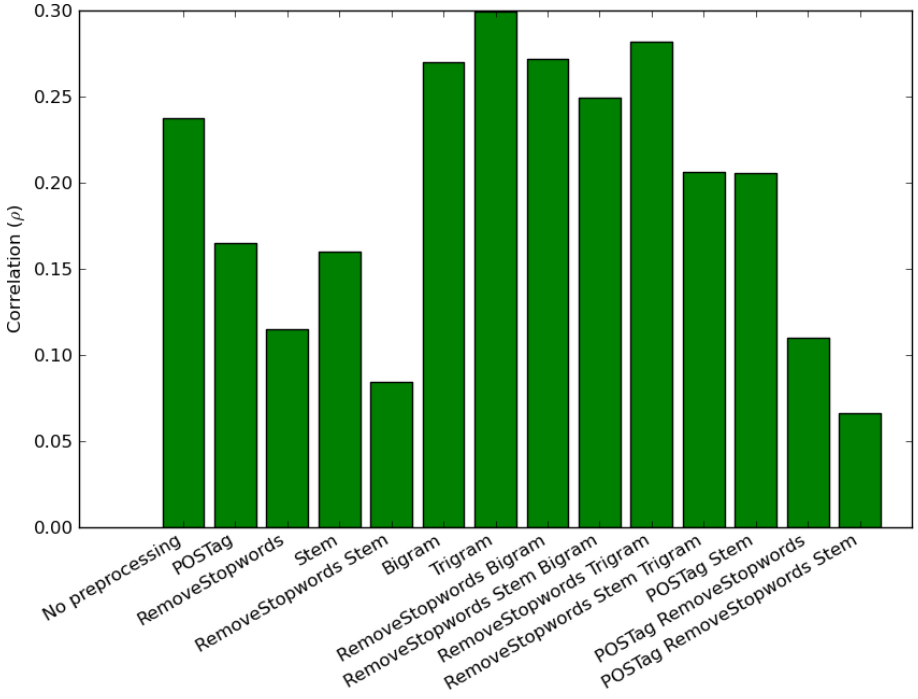



(b) Normalised Compression Distance.

**Figure B.3:** Performance of cosine and NCD metrics using TSA with stopped and stemmed base over Health24 corpus.

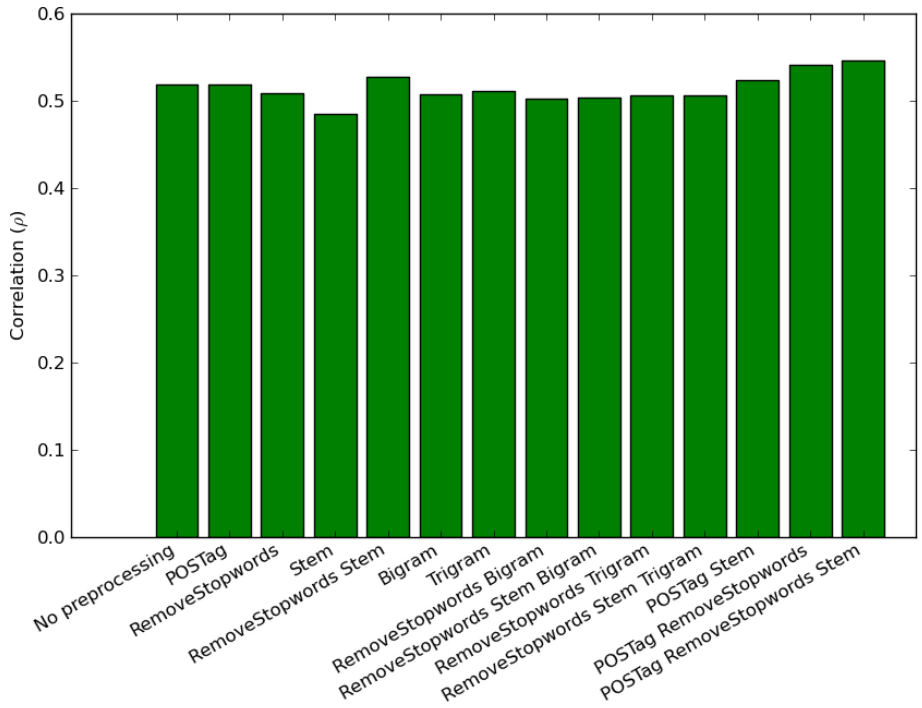



(a) Cosine similarity metric.

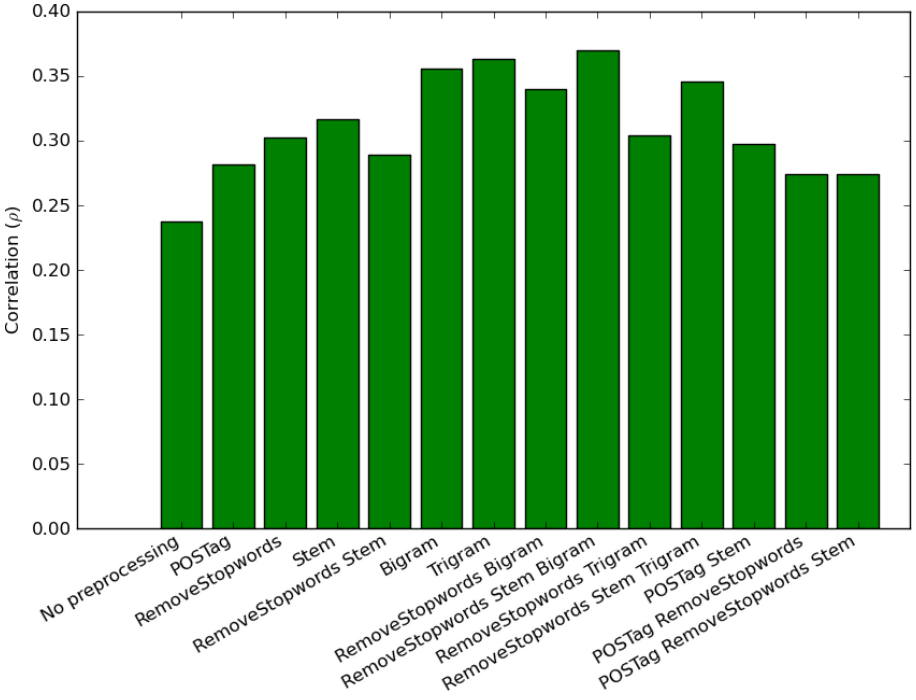



(b) Normalised Compression Distance.

**Figure B.4:** Performance of cosine and NCD metrics using TSS model over Lee & Pincombe corpus.

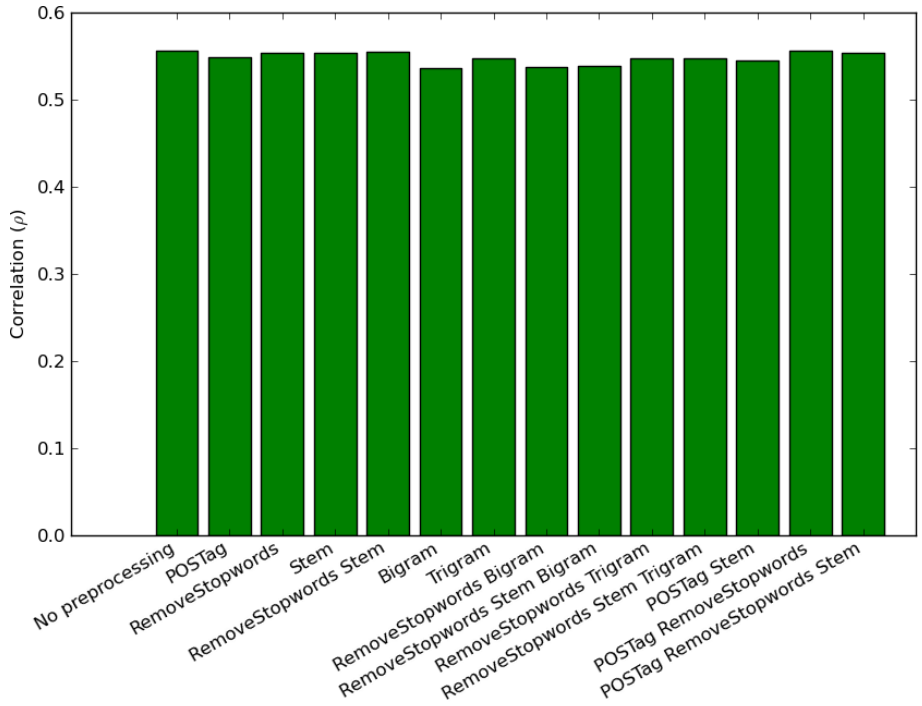



(a) Cosine similarity metric.

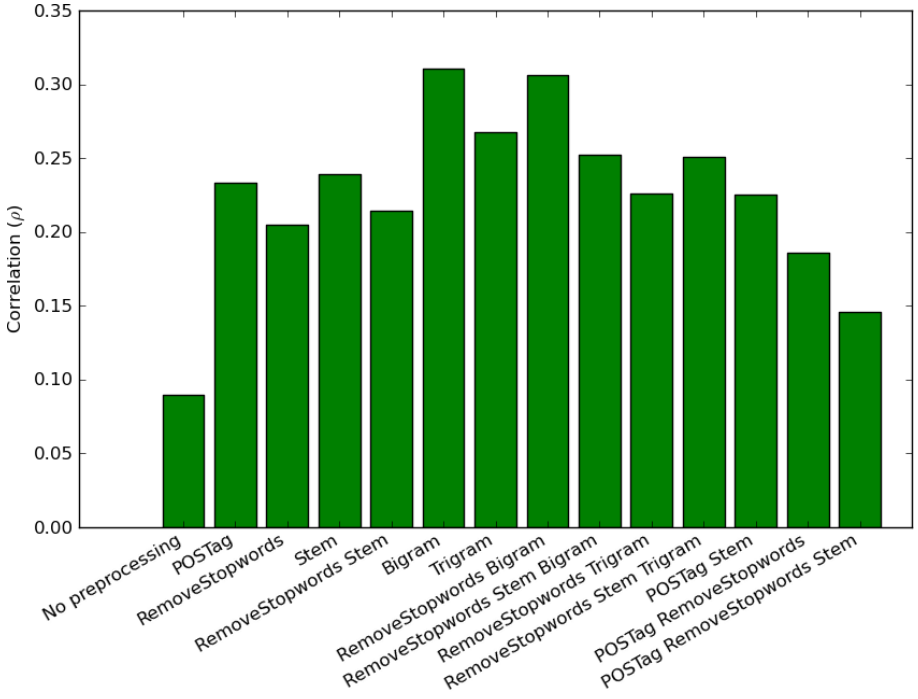



(b) Normalised Compression Distance.

**Figure B.5:** Performance of cosine and NCD metrics using TSA model over Lee & Pincombe corpus.




(a) Cosine similarity metric.




(b) Normalised Compression Distance.

**Figure B.6:** Performance of cosine and NCD metrics using TSA model with stopped and stemmed base over Lee & Pincombe corpus.

# List of References


[1] E. Gabrilovich and S. Markovitch, "Computing Semantic Relatedness Using Wikipedia-based Explicit Semantic Analysis," *Proceedings of the 20th International Joint Conference on Artificial Intelligence*, pp. 6–12, 2007.

[2] I. Witten and D. Milne, "An Effective, Low-Cost Measure of Semantic Relatedness Obtained From Wikipedia Links," in *Proceeding of AAAI Workshop on Wikipedia and Artificial Intelligence: an Evolving Synergy, AAAI Press, Chicago, USA*, 2008, pp. 25–30.

[3] B. Pincombe, *Comparison of Human and Latent Semantic Analysis (LSA) Judgements of Pairwise Document Similarities for a News Corpus.* Storming Media, 2004.

[4] P. Lyman and H. Varian, "How much information?" http://www2.sims.berkeley.edu/research/projects/how-much-info-2003/index.htm, 2003, accessed: May, 2010.

[5] M. Shamsfard, A. Nematzadeh, and S. Motiee, "Orank: An Ontology-based System for Ranking Documents," *International Journal of Computer Science*, vol. 1, no. 3, pp. 225–231, 2006.

[6] C. Manning, P. Raghavan, and H. Schütze, *Introduction to Information Retrieval.* Cambridge University Press, 2008.

[7] `http://www.google.com`.

[8] `http://www.bing.com`.

[9] W. Boswell, "The most popular search engines on the web," http://websearch.about.com/od/enginesanddirectories/tp/most-popular-search-engines.htm, accessed: May 2010.

[10] S. Brin and L. Page, "The Anatomy of a Large-Scale Hypertextual Web Search Engine," *Computer Networks and ISDN Systems*, vol. 30, no. 1-7, pp. 107–117, 1998.

[11] D. Metzler and W. B. Croft, "Beyond bags of words: Modeling implicit user preferences in information retrieval," *AAAI'06: Proceedings of the 21st National Conference on Artificial Intelligence*, pp. 1646–1649, 2006.

[12] T. Tran, P. Cimiano, S. Rudolph, and R. Studer, "Ontology-based Interpretation of Keywords for Semantic Search," *The Semantic Web*, pp. 523–536, 2008.

[13] "Owl web ontology language," `http://www.w3.org/TR/owl-features/`, accessed: May, 2010.

[14] "Resource description framework (rdf)," `http://www.w3.org/RDF/`, accessed: May, 2010.

[15] `http://en.wikipedia.org/`.

[16] M. Lee, B. Pincombe, and M. Welsh, "A Comparison of Machine Measures of Text Document Similarity with Human Judgments," in *27th Annual Meeting of the Cognitive Science Society (CogSci2005)*, 2005, pp. 1254–1259.

[17] G. Salton and M. McGill, *Introduction to Modern Information Retrieval*. McGraw-Hill New York, 1983.

[18] F. Crestani, "Application of Spreading Activation Techniques in Information Retrieval," *Artificial Intelligence Review*, vol. 11, no. 6, pp. 453–482, 1997.

[19] D. Metzler, S. Dumais, and C. Meek, "Similarity Measures for Short Segments of Text," *Advances in Information Retrieval*, pp. 16–27, 2007.

[20] V. Levenshteiti, "Binary Codes Capable of Correcting Deletions, Insertions, and Reversals," *Soviet Physics-Doklady*, vol. 10, no. 8, 1966.

[21] D. Lin, "An Information-Theoretic Definition of Similarity," in *Machine Learning: Proceedings of the Fifteenth International Conference (ICML'98)*. Morgan Kaufmann Pub, 1998, pp. 296–394.

[22] R. Cilibrasi and P. Vitanyi, "The Google Similarity Distance," *IEEE Transactions on Knowledge and Data Engineering*, vol. 19, no. 3, pp. 370–383, 2007.

[23] `http://www.gzip.org/`.

[24] `http://www.bzip.org/`.

[25] R. Rousseau, "George Kingsley Zipf: Life, Ideas, His Law and Informetrics," *Glottometrics*, vol. 3, pp. 11–18, 2002.

[26] M. Maron, "Mechanized Documentation: The Logic Behind a Probabilistic Interpretation." 1964.

[27] C. Van Rijsbergen, *Information Retrieval*, 1979, vol. 2.

[28] M. Gatford, M. Hancock-Beaulieu, S. Jones, S. Walker, and S. Robertson, "Okapi at TREC-3," in *The Third Text Retrieval Conference (TREC-3)*, 1995, pp. 109–126.

[29] G. Bennett, F. Scholer, and A. Uitdenbogerd, "A Comparative Study of Probabilistic and Language Models for Information Retrieval," in *Proceedings of the Nineteenth Conference on Australasian Database Volume 75*. Australian Computer Society, Inc., 2008, pp. 65–74.

[30] J. Ponte and W. Croft, "A Language Modeling Approach to Information Retrieval," in *Proceedings of the 21st Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*. ACM New York, NY, USA, 1998, pp. 275–281.

[31] M. Quillian, "Semantic Memory," *Semantic Information Processing*, pp. 227–270, 1968.

[32] D. Rumelhart and D. Norman, *Representation in Memory.* Storming Media, 1983.

[33] E. Gabrilovich, "The WordSimilarity-353 Test Collection," *Using Information Content to Evaluate Semantic Similarity in a Taxonomy*, 2002.

[34] F. Crestani, M. Lalmas, C. Van Rijsbergen, and I. Campbell, "'Is this document relevant? Probably': A Survey of Probabilistic Models in Information Retrieval," *ACM Computing Surveys (CSUR)*, vol. 30, no. 4, p. 552, 1998.

[35] A. Tversky *et al.*, "Features of Similarity," *Psychological Review*, vol. 84, no. 4, pp. 327–352, 1977.

[36] M. Li and P. Vitanyi, *An Introduction to Kolmogorov Complexity and Its Applications.* Springer Verlag, 1997.

[37] C. Bennet, P. Gács, M. Li, P. Vitanyi, and W. Zurek, "Information Distance," *IEEE Transactions on Information Theory*, vol. 44, no. 4, pp. 1407–1423, 1998.

[38] K. Jones *et al.*, "A Statistical Interpretation of Term Specificity and its Application in Retrieval," *Journal of Documentation*, vol. 60, pp. 493–502, 2004.

[39] S. Deerwester, S. T. Dumais, G. W. Furnas, T. K. Landauer, and R. Harshman, "Indexing by latent semantic analysis," *Journal of the American Society for Information Science*, vol. 41, pp. 391–407, 1990.

[40] T. Landauer, *Handbook of Latent Semantic Analysis.* Lawrence Erlbaum, 2007.

[41] W. Croft and D. Harper, "Using Probabilistic Models of Document Retrieval without Relevance Information," *Journal of Documentation*, vol. 35, no. 4, pp. 285–295, 1979.

[42] C. Zhai, *Statistical Language Models for Information Retrieval.* Morgan & Claypool Publishers, 2008, vol. 1, no. 1.

[43] B. Croft, D. Metzler, and T. Strohman, *Search Engines: Information Retrieval in Practice.* Addison-Wesley Publishing Company, USA, 2009.

[44] S. Javanmardi, J. Gao, and K. Wang, "Optimizing Two Stage Bigram Language Models for IR," *Proceedings of the 19th International Conference on World Wide Web*, pp. 1125–1126, 2010.

[45] C. Fellbaum *et al.*, *WordNet: An Electronic Lexical Database.* MIT Press, Cambridge, USA, 1998.

[46] D. Cruse, *Lexical Semantics.* Cambridge University Press, 1986.

[47] C. Leacock, G. Miller, and M. Chodorow, "Using Corpus Statistics and WordNet Relations for Sense Identification," *Computational Linguistics*, vol. 24, no. 1, pp. 147–165, 1998.

[48] Z. Wu and M. Palmer, "Verb Semantics and Lexical Selection," in *Proceedings of the 32nd Annual Meeting on Association for Computational Linguistics*. Association for Computational Linguistics, Morristown, USA, 1994, pp. 133–138.

[49] P. Resnik, "Using Information Content to Evaluate Semantic Similarity in a Taxonomy," in *International Joint Conference on Artificial Intelligence*, vol. 14, 1995, pp. 448–453.

[50] H. Liu and P. Singh, "ConceptNet - A Practical Commonsense Reasoning Tool-Kit," *BT Technology Journal*, vol. 22, no. 4, p. 211, 2004.

[51] M. Strube and S. Ponzetto, "WikiRelate! Computing Semantic Relatedness Using Wikipedia," in *Proceedings of the National Conference on Artificial Intelligence*, vol. 21, no. 2. MIT Press; 1999, 2006, p. 1419.

[52] `http://sourceforge.net/WikipediaMiner/`.

[53] `http://wdm.cs.waikato.ac.nz:8080/`.

[54] J. Sowa and A. Borgida, *Principles of Semantic Networks: Explorations in the Representation of Knowledge*. Morgan Kaufmann San Mateo, CA, 1991.

[55] U. Schiel, "Abstractions in Semantic Networks: Axiom Schemata for Generalization, Aggregation and Grouping," *ACM SIGART Bulletin*, no. 107, p. 26, 1989.

[56] S. Dolan, "Six Degrees of Wikipedia," *Retrieved from: http://www.netsoc.tcd.ie/ mu/wiki/*, 2008.

[57] R. Sedgewick and M. Schidlowsky, *Algorithms in Java, Part 5: Graph Algorithms*. Addison-Wesley Longman Publishing Co., Inc. Boston, MA, USA, 2003.

[58] K. Beck, *Test-Driven Development: By Example*. Addison-Wesley Professional, 2003.

[59] `http://blog.prashanthellina.com/2008/01/07/interfacing-python-with-c-using-ctypes/`.

[60] D. Metzler and W. Croft, "A Markov Random Field Model for Term Dependencies," in *Proceedings of the 28th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*. ACM New York, NY, USA, 2005, pp. 472–479.

[61] M. Kouylekov and B. Magnini, "Recognizing Textual Entailment with Tree Edit Distance Algorithms," *Recognizing Textual Entailment*, p. 17, 2006.

[62] P. Pantel and D. Lin, "Discovering Word Senses from Text," in *Proceedings of the Eighth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. ACM New York, NY, USA, 2002, pp. 613–619.

[63] P. Malakasiotis, "Paraphrase Recognition Using Machine Learning to Combine Similarity Measures," *ACL-IJCNLP 2009*, p. 27.

[64] `http://www.nltk.org/`.

[65] S. Bird, E. Klein, and E. Loper, *Natural Language Processing with Python.* O'reilly & Associates Inc, 2009.

[66] R. Mihalcea and A. Csomai, "Wikify!: Linking Documents to Encyclopedic Knowledge," in *Proceedings of the Sixteenth ACM Conference on Information and Knowledge Management.* ACM New York, NY, USA, 2007, pp. 233–242.

[67] D. Milne and I. H. Witten, "Learning to link with wikipedia," *CIKM '08: Proceeding of the 17th ACM Conference on Information and Knowledge Management*, pp. 509–518, 2008.

[68] G. Adomavicius and A. Tuzhilin, "Toward the Next Generation of Recommender Systems: A Survey of the State-of-the-Art and Possible Extensions," *IEEE Transactions on Knowledge and Data Engineering*, vol. 17, no. 6, pp. 734–749, 2005.

[69] `http://nusoap.sourceforge.net/`.

[70] `http://www.w3.org/TR/wsdl/`.

[71] `http://www.lemurproject.org/`.

[72] S. M. McNee, N. Kapoor, and J. A. Konstan, "Don't look stupid: Avoiding pitfalls when recommending research papers," in *CSCW '06: Proceedings of the 2006 20th anniversary Conference on Computer Supported Cooperative Work.* New York, NY, USA: ACM, 2006, pp. 171–180. [Online]. Available: http://dx.doi.org/10.1145/1180875.1180903

[73] D. Lewis, Y. Yang, T. Rose, and F. Li, "RCV1: A New Benchmark Collection for Text Categorization Research," *The Journal of Machine Learning Research*, vol. 5, pp. 361–397, 2004.

[74] J. Allan, B. Carterette, J. Aslam, V. Pavlu, B. Dachev, and E. Kanoulas, "Million Query Track 2007 Overview," 2007.

[75] I. Witten and E. Frank, *Data Mining: Practical Machine Learning Tools and Techniques.* Morgan Kaufmann, 2005.

[76] `http://divisi.media.mit.edu/`.

[77] M. W. Berry, "Large scale sparse singular value computations," *International Journal of Supercomputer Applications*, vol. 6, pp. 13–49, 1992.

[78] T. Bogers and A. Van den Bosch, "Comparing and Evaluating Information Retrieval Algorithms for News Recommendation," in *Proceedings of the 2007 ACM Conference on Recommender Systems.* ACM, 2007, p. 144.