Determining Whether a Document Changes in Subject

by

Colin J. Nicholson

(Under the direction of Michael Covington)

Abstract

This thesis describes a method for determining whether a document is composed of text related to a single subject or text that changes subjects. The algorithm involves dividing the document into five equal parts and measuring the similarity of the different sections with one another. Documents that drift in subject are shown to have a higher standard deviation of similarity values than documents that remain on one subject. This method requires a threshold value that is specific to the domain to work properly.

Index words: Coherence, Subject drift, Topic drift, Text classification

Determining Whether a Document Changes in Subject

by

Colin J. Nicholson

A.B. The University of Georgia, 2005

A Thesis Submitted to the Graduate Faculty
of The University of Georgia in Partial Fulfillment
of the
Requirements for the Degree

Master of Science

Athens, Georgia

2009

© 2009

Colin J. Nicholson

All Rights Reserved

Determining Whether a Document Changes in Subject

by

Colin J. Nicholson

Approved:	
Major Professor:	Michael Covington
Committee:	W. Don Potter Charles Cross

Electronic Version Approved:

Maureen Grasso
Dean of the Graduate School
The University of Georgia
May 2009

Dedication

To my family.

Acknowledgments

First, I have to thank Dr. Michael Covington for starting this project and improving it from that point until the end. Also, I have to thank Dr. Don Potter, Dr. Khaled Rasheed, and Dr. Donald Nute for providing me with an education that enabled me to complete this thesis. Dr. William Hollingsworth provided great advice along the way and I thank him as well. I thank my parents for supporting me all these years. Finally, I have to thank Kumar Ujjwal for providing me with countless amounts of advice during my time in graduate school.

Table of Contents

Appendix

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Background

Coherence in a text is semantic unity. Discourse made of parts that seem connected in subject matter has a high level of coherence; text that jumps around in subject matter has a low level of coherence. No universally accepted method for measuring coherence exists since coherence relates to a subjective interpretation of how well subjects connect with one another. In fact, individuals in different fields have defined different types of coherence before devising algorithms to measure them [8] [13].

Knowledge of a document's coherence level can help with various tasks. Computerized coherence measurement provides assistance to teachers with grading of essays [15]. Research groups have found coherence measurement to be an important step for developing systems that can locate topically-related material in streams of broadcast speech [16]. Also, determining whether a website is on one or multiple topics can assist search engines return the most relevant pages for queries; and search engines can be quite lucrative [4]. Coherence level is information that can assist computational linguists to discover different styles of writing which can help accomplish tasks such as determining authorship of text. Along with many other measurements of text, coherence level can give computers a better indication of the nature of the text they are working with.

Medical researchers also benefit from computerized coherence level measurement. Some cite speech abnormalities as an indicator of certain mental disorders [2]. Currently, software is being developed to advance medical knowledge by finding connections in seemingly unrelated texts [9]. Others have attempted to measure topic drift. These researchers represent different fields; I will discuss them in the next section.

1.2 Previous Work

1.2.1 Todd's Concept Mapping Work

Todd, Thienpermpool, and Keyuravong developed a method for measuring coherence that involves mapping of relationships between concepts; the method helps teachers assess students' work [15]. Todd's measure of coherence is called topic-based analysis; this measure identifies key concepts through frequency and determines logical relationships between them [14].

Todd's algorithm looks at the order in which concepts appear in the document and tries to determine whether that order follows a hierarchy built by earlier steps; coherence is ranked with how well a given piece of discourse follows the diagram. Identifying key concepts, in Todd's algorithm, involves considering the nouns and noun phrases in a document. Concepts are considered more important to the document if they repeat more often than others or if they are highlighted in a fashion such as appearing in the title of sections or being underlined.

To identify relationships between concepts, Todd uses two key associations. The first is called the inclusion relationship, developed by McCarthy in 1988 [10]. It covers a range of superordinate/subordinate relationships. The second relationship used by Todd is cause/effect; it looks for phrases such as "storms *cause* flooding" and draws relationships. Hierarchies are easy to form with superordinate/subordinate relationships; cause/effect relationships do not form hierarchies but are linked.

1.2.2 Latent Semantic Analysis

Latent Semantic Analysis is a document classification technique developed by Landauer, Foltz, and Laham, which extracts words from a document and links those words to topics based on their co-occurrence in other, previously analyzed documents [7]. LSA can help computers infer concepts about text that are not in the text at all but are related to words in the text. Early LSA work used the technique to estimate coherence, learnability of passages by individual students, and the quality and quantity of knowledge contained in an essay.

Latent Semantic Analysis attempts to build a relationship between terms and concepts relating to those terms; the goal is for the LSA system to make the same category judgments on words that humans use. To use LSA, one must start with a large corpus of documents, each on a single subject. First, the corpus text is represented as a matrix where each row stands for a unique word and each column represents a text passage where the word occurred. The matrix is decomposed to become the product of three other matrices. The first matrix describes the original row entities as vectors of derived orthogonal factor values, the second describes the original column entities in the same way, and the third is a diagonal matrix containing scaling values such that when the three components are matrix-multiplied, the original matrix is reconstructed. In the end, we are left with connections between terms and documents, which can be called concepts.

Elvevaag, Foltz, Weinberger, and Goldberg used Latent Semantic Analysis to analyze coherence in speech to draw connections between coherence in discourse and schizophrenia [3]. The team used LSA to measure coherence of speech from schizophrenic patients as well as healthy controls. Ultimately, their LSA program was able to successfully measure the level of incoherence in speech as well as determine whether a given piece of discourse belonged to a patient or control.

1.2.3 Brown's Link Detection Work

DARPA's Topic Detection and Tracking (TDT) program uses news stories (collected from newspaper and broadcasts) as its data and attempts to use computers to extract meaningful information from those data. Much of the work for the TDT program focuses on enabling computers to set boundaries to determine the coherence and cohesion of news stories. For example, for the 1998 DARPA Topic Detection and Tracking program a group led by Paul van Mulbregt developed a Hidden Markov Model approach to infer story boundaries and determine if certain topics were being covered by the news stories [11]. Link detection is an aspect of the DARPA Topic Detection and Tracking program; it is the section that most closely resembles the work in this thesis. The link detection task requires a computer to determine if two stories are topically "linked," that is, if they are about the same news subject. Those who worked on the TDT program's link detection task used a corpus of stories transcribed from news broadcasts for testing.

A group led by Ralph Brown from Carnegie Mellon University developed and tested two story link detection systems [1]. The systems compare the two documents using a standard text similarity measure (TF*IDF weighted cosine similarity) and determine if the similarity result is above a predetermined threshold; if the value exceeds the threshold the stories are called linked while if not the stories are said to be not linked. Both systems were tested on training data that was considered when building the systems and evaluation data full of story pairs unseen by the systems. The systems performed well on training data but very poorly on the evaluation data. It seems as if the systems were unable to find thresholds for similarity that would cover all documents rather than just the few documents the system was being optimized for.

1.3 Disadvantages of Past Methods

The method proposed by this thesis has a big advantage over most of the previous work on this subject: simplicity. For example, Todd's work involves a complicated multi-step process that identifies key concepts in a text, maps relationships between these concepts, and builds a hierarchy to be used with the text to try to measure coherence. Potential for error can be high in multi-step processes, especially when later steps hinge on tasks such as relationship-mapping which can be difficult for computers to accomplish with a high level of accuracy. Elvevaag's Latent Semantic Analysis work is difficult to replicate due to the size of the training corpus; LSA projects usually require hundreds of thousands to millions of documents to train on in order to be effective in distinguishing between subjects. It also suffers from an issue that most coherence-measuring algorithms are stuck with: it does not know what to do with text on a subject that was not trained by the system beforehand.

Brown's work on link detection provided a start in a direction that could be quite worthwhile. The concept is simple: compute the text similarity of two documents and determine if the similarity resides above a predetermined threshold. This system for computing coherence is fast, simple, and works on any domain; it needs no knowledgebase of subject documents to map the current text with. Unfortunately the method produced unreliable results when tested for the TDT program's link detection task and the research ended with a report on the failed experiment and no usable algorithm. I have tested Brown's basic method myself and it seems as if no blanket similarity threshold exists that would apply to several different documents and adequately classify them based on coherence.

I do not believe the general idea of Brown's work should be abandoned because of all the benefits the method would possess if it worked properly. It seems as if one key change needs to be made: classification needs to be based on a number other than the text similarity score. The method proposed in this thesis involves a measurement for classification that is more stable, even under conditions where the writing style might skew similarity values one way or another. I will discuss this method in the next chapter.

Chapter 2

Computing Text Similarity Scores

A good way to compare two texts is to compare their vocabularies. This can be done by counting how many times each term appears in each document, and then comparing the lists of numbers by treating them as vectors. The angle between the vectors reflects the similarity of the relative proportions of different words but is not affected by the absolute number of words or the length of the text. The idea of representing text documents as vectors was introduced by Salton et al. in 1975 [12]. For example, let's take a look at three documents: A, B, and C (Figure 2.1), which contain nine, six, and six terms, respectively.

These documents can be compared using word frequency tables. Table 2.1 shows three columns that represent the three documents while rows indicate the frequency of each term within a document. One thing to note is that the order of the terms in the document is not represented by the table; sentences that use the same terms but in a different sequence will appear to be the same in a word frequency table. This can cause a loss of information as "dog bites man" matches perfectly with "man bites dog."

Document A: The new car is parked in the new garage.

Document B: Where is the new car parked?

Document C: The dog is in the doghouse.

Figure 2.1: Documents A, B, and C

Table 2.1: Term frequencies for each document

Term	**A**	**B**	**C**
the	2	1	2
new	2	1	0
car	1	1	0
is	1	1	1
parked	1	1	0
in	1	0	1
garage	1	0	0
where	0	1	0
dog	0	0	1
doghouse	0	0	1

2.1 Vector Comparisons

The columns of the term frequency (TF) table can be seen as three lists of numbers; these three lists can be converted to vectors (reading downward along each column), shown below.

TF Vector A: (2, 2, 1, 1, 1, 1, 1, 0, 0, 0)

TF Vector B: (1, 1, 1, 1, 1, 0, 0, 1, 0, 0)

TF Vector C: (2, 0, 0, 1, 0, 1, 0, 0, 1, 1)

The advantage of turning these text documents into vectors is we can now apply linear algebra methods to the documents. For example: to measure the similarity of two documents with each other, we can calculate the dot product of the vectors. The dot product is computed by multiplying each element in one vector with the corresponding element in the other vector and then adding the products together. The following equation demonstrates how to calculate the dot product of vectors A and B, each containing n elements.

$$(a_1\ a_2\ a_3\ \cdots\ a_n) \cdot (b_1\ b_2\ b_3\ \cdots\ b_n) = a_1b_1 + a_2b_2 + a_3b_3 +\ \cdots\ + a_nb_n$$

One problem with using the vector dot product to measure text similarity is that it is influenced by the length of the documents. For example, a lengthy document that repeated the term “the” 100 times would show a higher similarity with document A in our example than documents B or C would, even if it shared no other terms in common with document A. A more effective similarity-measuring algorithm would need an extra step to adjust the equation to prevent such a length bias.

2.2 Cosine of Angle Between Vectors

If we calculate the cosine of the angle between the vectors, we produce a value that is normalized for document length. The size of the angle between the vectors will depend on how closely the vectors run together; vectors with many terms in common will run closer together than vectors with few terms in common. As the angle between the vectors decreases, we will receive a higher cosine value; this is indicative of a higher level of similarity between the texts. The range of the cosine of the angle between vectors is 0 to 1.

The cosine of the angle between vectors can be computed by dividing the dot product of the vectors (calculated in the previous section) by the product of the length of the vectors. The calculation is illustrated by the following equation, where a_1, a_2, ... a_n and b_1, b_2, ... b_n represent elements of vectors A and B.

$$\cos\theta = \frac{(a_1 b_1)+(a_2 b_2)+\ \cdots\ +(a_n b_n)}{\sqrt{(a_1{}^2+a_2{}^2+\ \cdots\ +a_n{}^2)}\sqrt{(b_1{}^2+b_2{}^2+\ \cdots\ +b_n{}^2)}}$$

2.3 Term Weighting Schemes

It is possible to produce more accurate similarity scores by weighting each term by its Inverse Document Frequency (IDF) score. This will give higher values to more important words in the text while giving lower values to trivial words such as “the” or “and” which can appear in any document without giving clues as to the document’s subject. The IDF score of a term is log $\frac{T}{N}$ where T is the total number of documents under consideration and N is the number of documents containing the term [5]. In this thesis we will always use the logarithm with base 10. If a word appears in every document, its IDF score will be the logarithm of 1, namely 0. Taking the logarithm of the IDF score also prevents the IDF value from getting to be too high. The fewer documents a term appears in, the higher its IDF score. Table 2.2 shows the terms in the documents from Figure 2.1 with their IDF scores.

The Term Frequency * Inverse Document Frequency (TF*IDF) score is the IDF score for a term multiplied by how many times that term appears in a given document. The TF*IDF score will give more weight to terms that repeatedly appear in a document and can be indicative of the document's topic (remember, since we are using IDF scores, terms that appear repeatedly in ALL documents have no value). We will use the TF*IDF method in this thesis when generating vectors from documents of terms.

Table 2.3 shows the terms of the documents from Figure 2.1 with their TF*IDF values. Note the value for the term "new" in document A is twice the IDF score for that term since that term appears twice in that document. With the columns of this table we can make TF*IDF vectors for each document, shown below.

TF*IDF Vector A: (0, 0.3520, 0.1760, 0, 0.1760, 0.1760, 0.4771, 0, 0, 0)
TF*IDF Vector B: (0, 0.1760, 0.1760, 0, 0.1760, 0, 0, 0.4771, 0, 0)
TF*IDF Vector C: (0, 0, 0, 0, 0, 0.1760, 0, 0, 0.4771, 0.4771)

As Table 2.4 shows, the cosine of the angle between vectors using the TF*IDF vectors shows a stronger distinction between the documents than the TF dot product scores, the TF*IDF dot product scores, or the cosine of the angle between the vectors using the TF vectors.

Table 2.2: IDF values

Term	**IDF Value**
the	0
new	0.1760
car	0.1760
is	0
parked	0.1760
in	0.1760
garage	0.4771
where	0.4771
dog	0.4771
doghouse	0.4771

Table 2.3: TF*IDF values

Term	**A**	**B**	**C**
the	0	0	0
new	0.3520	0.1760	0
car	0.1760	0.1760	0
is	0	0	0
parked	0.1760	0.1760	0
in	0.1760	0	0.1760
garage	0.4771	0	0
where	0	0.4771	0
dog	0	0	0.4771
doghouse	0	0	0.4771

Table 2.4: Similarity scores

Method	**A and B**	**A and C**	**Difference**
TF dot product	7	6	1.17 times
TF*IDF dot product	0.1239	0.0309	4 times
TF cosine of angle between vectors	0.7926	0.5880	1.35 times
TF*IDF cosine of angle between vectors	0.3419	0.0664	5.15 times

Chapter 3

Document Division and Similarity Calculation Method

3.1 Example Document

For the research in this thesis, I will be considering one document at a time and attempting to determine whether it is all on one subject or drifts in topic; text similarity calculation is an important part of this process. In order to measure text similarity, a query document must be compared with other documents to return some kind of measurable value. With only one document to work with, the document must be divided up into sections, or subdocuments, which can then be treated as separate documents. A small collection of sentences can illustrate the process that can be used for determining similarity within a document. For instance, let's consider the five sets of sentences, all on the same subject, in Figure 3.1.

The five groups were formed by their placement; they represent the beginning, end, and three middle sections of the document. These will be our five subdocuments. First, we must assign values to each word; we do this using the IDF values, described earlier. The values

1. Dogs are nice pets. Many people own them.
2. They usually like people. Dogs are usually loyal.
3. Dogs are good companions. People really love dogs.
4. I own two dogs. They are quite energetic.
5. Dogs eat many things. They are always hungry.

Figure 3.1: Document that stays on subject

Table 3.1: IDF values for each term

Term	**IDF Value**
dogs	0
are	0
nice	0.6989
pets	0.6989
many	0.3979
people	0.2218
own	0.3979
them	0.6989
they	0.0969
usually	0.6989
like	0.6989
loyal	0.6989
good	0.6989
companions	0.6989
really	0.6989
love	0.6989
I	0.6989
two	0.6989
quite	0.6989
energetic	0.6989
eat	0.6989
things	0.6989
hungry	0.6989

obtained for each unique term can be seen in Table 3.1; these values will be multiplied by each term's frequency within a subdocument to make TF*IDF vectors out of each subdocument.

Now that each term has a value we can make comparisons of the five subdocuments. We will use the dot product of the vectors to produce similarity values. It is not necessary to use the cosine of the angle between vectors value since the subdocuments are all the same length. The similarity values of the first subdocument with the other four subdocuments can be seen in Table 3.2.

Table 3.2: Similarity values for coherent document

Texts	**Similarity**
S(1,2)	0.0586
S(1,3)	0.0492
S(1,4)	0.0586
S(1,5)	0.0586

1. Dogs are nice pets. Many people own them.
2. They usually like people. Dogs are usually loyal.
3. Dogs are good companions. People really love dogs.
4. Some people teach math. Everyone must learn math.
5. Math can be difficult. Kids usually dislike math.

Figure 3.2: Document that drifts in subject

Now let's calculate the vector dot product for a similar document, only one that drifts in subject. Consider the set of sentences in Figure 3.2.

The IDF values for each unique term in the document that drifts in subject can be seen in Table 3.3. Using the first subdocument as the query document, we get the similarity values seen in Table 3.4.

Do the similarity scores distinguish between the document that is all on one subject and the document that drifts? As it turns out, the average of the similarity values for the document that stayed on subject was the same as the average of the similarity values for the document that changed subjects: 0.0562. Here we run into a problem like the one Brown faced: no clear similarity distinction between documents with different coherence levels.

The distribution of the values, however, is different. On the document that is all on one subject, the similarity values are all fairly close in value. However, the document that drifts

Table 3.3: IDF values for each term

Term	**IDF Value**
dogs	0.2218
are	0.2218
nice	0.6989
pets	0.6989
many	0.6989
people	0.0969
own	0.6989
them	0.6989
they	0.3979
usually	0.3979
like	0.6989
loyal	0.6989
good	0.6989
companions	0.6989
really	0.6989
love	0.6989
some	0.6989
teach	0.6989
math	0.3979
everyone	0.6989
must	0.6989
learn	0.6989
can	0.6989
be	0.6989
difficult	0.6989
kids	0.6989
dislike	0.6989

Table 3.4: Similarity values for drifting document

Texts	**Similarity**
S(1,2)	0.1077
S(1,3)	0.1077
S(1,4)	0.0094
S(1,5)	0

in subject shows a strong lowering in similarity values in the subdocuments that are on a different subject than the input document. If we were to use the fifth subdocument as the query, we would see a similar trend in the opposite direction.

As it turns out, the best indicator of a document's coherence level is not the similarity score itself, but the difference between the similarity values of the different subdocuments with each other. The standard deviation of the similarity values for the document that stays on subject is 0.0047; the document that drifts has a standard deviation that is over ten times higher: 0.0595. A document that is all on the same subject will show little similarity value drift when compared with documents that have a sharp change in subject and whose subdocuments are more likely to be full of different terms. The idea of using changes in term frequency to extract information from text is not new; in her work on document cohesion, Hearst looked at the sharpest boundaries where changes of words occur in a document to draw lines on where different sections of a document begin and end [6].

3.2 The Algorithm

3.2.1 Why Five?

The first step in the algorithm for determining a document's coherence is to divide the document into five subdocuments; this was shown on a small scale in the example document discussed earlier. It is important to understand why the document is divided into five sections. Just as in the previous section, we are going to calculate the similarity values of the subdocuments with one another to get standard deviation values. It is possible to lose information if we have too few or too many subdocuments.

Consider Figure 3.3 which shows one document divided into three, five and ten subdocuments. Here, the document switches subject halfway through as the color in the figure changes. The document with three sections would only provide two similarity scores to work with (one of the subdocuments would be compared with itself and just return a 1 every time so we disregard its similarity score). Let's assume the middle subdocument would return a middle-range similarity value when the first subdocument is used as the query since half its terms are on the same subject as the first subdocument, and the third subdocument would return a low score since it is on a different subject altogether. If the similarity scores for subdocuments 2 and 3 were 0.5 and 0.1, the standard deviation of those two values would be 0.2828. However, if the third subdocument were on the same subject as the first subdocument and the similarity scores were 0.5 and 0.9, we would get the same standard deviation value: 0.2828. Clearly, with only two subdocument values, we do not have enough information to calculate a standard deviation value that distinguishes between documents on one subject and documents that drift.

It is also possible to have too many subdocument values for our purposes. Consider the rightmost image in Figure 3.3; it is one document divided into ten subdocuments, changing subjects midway just as the previous one did. By having too many sections, we run the risk of having sections that are too small and might not show similarity with other sections on the same subject just because they do not contain an adequate sample of that topic's terms.

So, finding the proper number of subdocuments turns into a balancing act; we don't want too few or too many. Three subdocuments was shown above to be inadequate; we attempted

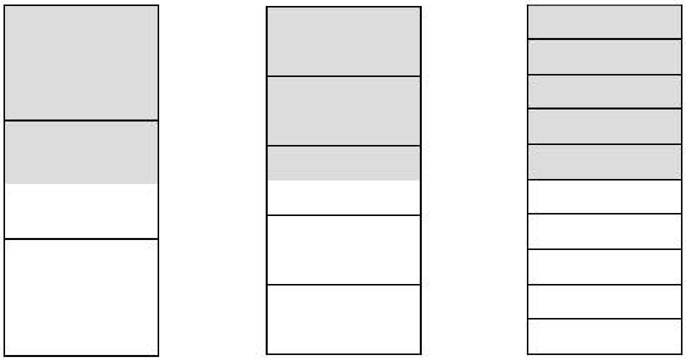

Figure 3.3: One document divided into three, five, and ten subdocuments

to classify documents that changed around halfway through using four subdocuments but five subdocuments produced better results for the data used in this thesis. When four subdocuments were used, some documents had subjects that changed at around halfway through the second subdocument (seen in Figure 3.4). These documents produced higher standard deviation values with five subdocuments than four since using five subdocuments enabled one subdocument to show a high similarity score when the first subdocument was used as input while four subdocuments produces moderate or low similarity values and leads to a lower standard deviation (seen in Table 3.5). Like the crossover or mutation rate of a genetic algorithm, the number of subdocuments is a parameter that needs to be tested on data before being set. Now that we have the sections, we transform each subdocument into a vector of TF*IDF values.

3.2.2 Getting the Average Standard Deviation

Now we will find the similarity scores of the first subdocument with the other four subdocuments. In this thesis, we use the dot product of the vectors to calculate similarity (we do not need to use the cosine of the angle between the vectors since the subdocuments are all the

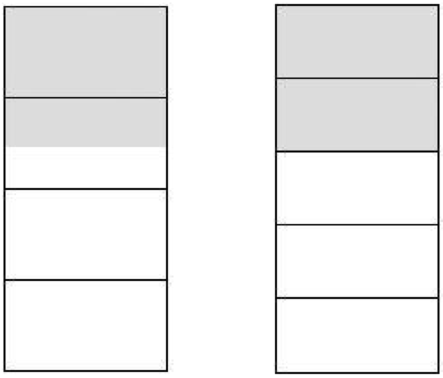

Figure 3.4: One document divided into four and five subdocuments

Table 3.5: Similarity values for documents in Figure 3.2

Four Subdocuments	**Five Subdocuments**
S(1,2) = 0.6	S(1,2) = 0.9
S(1,3) = 0.1	S(1,3) = 0.1
S(1,4) = 0.1	S(1,4) = 0.1
	S(1,5) = 0.1
Std Dev = 0.29	**Std Dev = 0.4**

same length). When comparing different documents to determine which has a higher trend of coherence, we must normalize the values so that all similarity scores can be compared evenly on a scale from 0 (no similarity) to 1 (perfect similarity). This is done by dividing the four other subdocument similarity scores by the input subdocument's similarity score with itself.

At this point, we have four similarity scores for the first subdocument with values ranging between 0 and 1; now we take the standard deviation of these values. The standard deviation tells us if the different sections are producing similarity scores that have a great range or are close together in value. Now we have a value that represents the diversity of the similarity of the first subdocument with the other four subdocuments. We will repeat this same process, only with the second, forth, and fifth subdocuments as the inputs. We do not consider the middle subdocument as input because in documents that change around the halfway mark (many of the documents used in this thesis), it will contain terms from both topics and will give high similarity values to all other subdocuments and provide no useful information. Figure 3.5 shows the order of the similarity comparisons of the different sections within a document.

Now that we have four standard deviation values, we take the average of these four values to attain our final value; it represents the average difference in similarity among the different sections of the document. This value is the one that will be used to determine if the document drifts in subject or not. Testing has shown documents that drift in subject generally have a higher average standard deviation value than documents that remain on one subject.

3.3 Example Calculation

Let's run a 3,000 word document on the subject of Mars (retrieved from the website Wikipedia on June 27, 2008) through the algorithm to further illustrate the process. Initially, we make five groups of 600 words each: the beginning, three middle fifths, and end of the document. After calculating the TF*IDF scores, we can calculate the similarity values of sections 1,2,4, and 5 with each other section using the dot product of the vectors (Table 3.6). The normalized similarity values are calculated by dividing the similarity scores by the query document's similarity score with itself (Table 3.7); the standard deviations are calculated using these values (Table 3.8).

The average of these four standard deviations is 0.0477. This is our final value for the document. It should be noted that subdocuments nearest the query document usually give

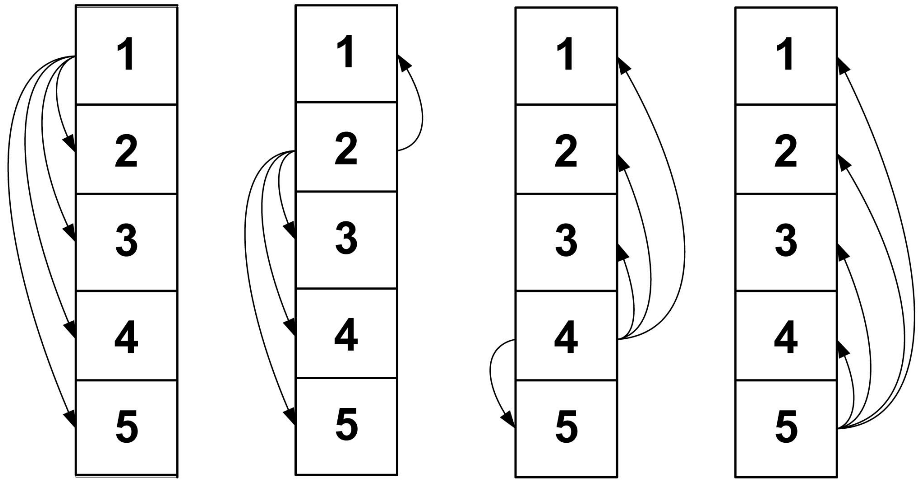

Figure 3.5: Subdocument comparison steps

Table 3.6: Similarity values of x sections with y sections

		y				
		1	2	3	4	5
x	1	57.7	8.9	3.5	3.1	2.4
	2	8.9	52.9	9.3	3.8	3.5
	3	3.5	9.3	—	5.4	2.4
	4	3.1	3.8	5.4	66.8	10.8
	5	2.4	3.5	2.4	10.8	155.7

Table 3.7: Normalized similarity values of x sections with y sections

		y				
		1	2	3	4	5
x	1	1	0.1551	0.0619	0.0541	0.0423
	2	0.1713	1	0.1795	0.0728	0.0680
	4	0.0467	0.0569	0.0813	1	0.1627
	5	0.0156	0.0228	0.0160	0.0698	1

Table 3.8: Standard deviation values

Query	**Standard Deviation**
1	0.0518
2	0.0607
4	0.0525
5	0.0260

higher similarity scores than the other subdocuments. This occurs in documents that are all on the same subject, such as this one, because these sections are right next to each other in the document and usually share a more specific topic than the subdocuments that are not as close; however, in this example, the size difference in similarity is slight. In a document that drifts in topic, the similarity scores for nearby subdocuments would be much greater and lead to standard deviation values that are, on average, at least twice as high as the one in this document.

3.4 How To Use The Algorithm

This algorithm works better on certain domains than others. In short, the algorithm works well on texts that stay on the same general subject, repeating terms, such as a scientific essay on a certain subject. It works less well on domains that have a lot of text that might fall under the same category, but have sections that have little in common with one another and might not repeat terms very much throughout the document, such as a website that is a collection of articles on sports where each article might mention a different sport or team.

To use the algorithm, one must first take a small sample of documents (half of which stay on subject and half of which drift) from the domain and find the average standard deviation value of the documents using the method of dividing each document into subdocuments and finding the standard deviation of the similarity scores. When one is comfortable enough to have standard deviation values that seem to show a consistent trend in the one-subject and multi-subject documents, one can now use the average standard deviation score of all of the one subject documents and the average standard deviation score of all of the multi-subject documents to find a threshold value for coherence: the midpoint of these two values.

To classify documents, we can run the algorithm on a document and if its average standard deviation score is above that threshold value it is classified as drifting in subject; if it is below the threshold it is classified as being on one subject.

3.5 Effects as Document Size Increases

In section 3.1 we looked at an example of how a small document behaves when similarity values are calculated within it using a part of the document as the query. This leaves open the question of how larger documents will behave under similar tests. In the example document mentioned in section 3.1, all terms have an IDF value of 0.2218 or above, even terms unrelated to the subjects.

1. **Dogs are nice pets. Many people own dogs.**
2. **They usually like people. Dogs are usually loyal.**
3. **Dogs offer good companionship. I've always loved dogs.**
4. **Some people teach math. Everyone must learn math.**
5. **Math can be difficult. Kids usually dislike math.**

Figure 3.6: Document from section 3.1

In this small document, words that relate to the topics are given high values, but so are arbitrary words that happen to be in one or two sections of the document but not in other sections. However, as we examine larger documents that change in subject, common words that are not specific to the subject are more and more likely to appear in other parts of the document. This leads to an interesting effect where words related to the subject maintain their high values but words that are less related decline in value. In Figure 3.7, the words in bold are the words expected to not appear in documents on the alternate subject, and would most likely have higher IDF values in lengthy documents.

1. **...Dogs** are nice **pets.** Many people **own dogs...**
2. ...They usually like people. **Dogs** are usually **loyal...**
3. **...Dogs** offer good **companionship.** I've always loved **dogs...**
4. ...Some people **teach math.** Everyone must **learn math...**
5. **...Math** can be difficult. Kids usually dislike **math...**

Figure 3.7: Document from section 3.1 with key words emphasized

Also, as the number of words in the document increases, more words related to a subject are found in each subdocument. The combined effects of lowering the value of trivial words and adding more words related to the subject produce much stronger distinctions between subdocuments that are on the same subject as the query and subdocuments that are on a different subject than the query.

Chapter 4

News Blog Values

How will real documents respond to the algorithm proposed in this thesis? Will the average standard deviation values show distinct trends depending on whether the documents drift in subject? In this chapter, we will take documents containing blog entries on the same subject and compare the documents' average standard deviation values with the values of documents made of blog entries on different subjects to answer that question.

The pages used in this section are from the website Reuters (http://blogs.Reuters.com). The documents were retrieved from the Reuters website on June 3, 2008, and were the most recent three or four pages of blogs on three different subjects. Four documents contained groups of news blog entries about Pakistan, four documents contained groups of news blog entries about the Olympics, and three documents contained groups of news blog entries about Africa. Documents ranged in size from around 2,000 to around 5,000 words each. Also, eleven documents were created using the first half of one document and the second half of a document on a different subject.

The average standard deviation values for one-subject and two-subject blog documents can be seen in Table 4.1. The average value for all one-subject blog documents was 0.0538 while the average value for all two-subject blog documents was 0.0947; the threshold value found by averaging these values is 0.0742.

Looking at the data, the average standard deviation values for the documents show two clear groups: those above and below the threshold value. With a couple of exceptions, the two groups are identical to the groups one would also make if distinguishing the documents based on coherence. Clearly, the method proposed in this thesis is able to produce values on

Table 4.1: Average standard deviation values of one-subject and two-subject news blog pages

Subject	**Avg Std Dev**	**Subject**	**Avg Std Dev**
Africa Page 1	0.0516	Africa Page 1/Olympics Page 1	0.1056
Africa Page 2	0.0324	Africa Page 1/Pakistan Page 4	0.0885
Africa Page 3	0.0654	Africa Page 2/Olympics Page 2	0.0763
Pakistan Page 1	0.0838	Olympics Page 2/Africa Page 1	0.0923
Pakistan Page 2	0.0425	Olympics Page 2/Africa Page 2	0.0966
Pakistan Page 3	0.0422	Olympics Page 3/Africa Page 3	0.0951
Pakistan Page 4	0.0527	Olympics Page 4/Africa Page 2	0.0967
Olympics Page 1	0.0468	Pakistan Page 1/Olympics Page 1	0.1073
Olympics Page 2	0.0671	Pakistan Page 2/Olympics Page 2	0.1180
Olympics Page 3	0.0578	Pakistan Page 3/Africa Page 3	0.0540
Olympics Page 4	0.0490	Pakistan Page 4/Olympics Page 4	0.1119
AVERAGE:	**0.0538**	**AVERAGE:**	**0.0947**

real news blog data that show distinction based on coherence. In the next chapter, we will see how accurately large amounts of data can be classified.

Chapter 5

Wikipedia Test

In this test, 156 pages from the website Wikipedia (http://www.Wikipedia.org), retrieved on June 27, 2008, were used. Wikipedia is a website that contains pages on various subjects, much like an online dictionary. Pages on each subject are usually divided into sections related to subtopics of the general topic; the text on a given Wikipedia page is often times written by more than one author.

Each page used was the main page for a different country (they are listed in Table 5.2). Each page on a certain country is composed of different sections relating to a different aspect of the country (language, history, etc.). For these pages, the length of the documents ranged from 1,449 words (Andorra) to 16,074 (Cuba). Most documents were in the middle of that range. The goal of this test is to determine if the algorithm proposed in this thesis is able to successfully categorize which files are composed of text from one article on a certain country and which files are composed of text from two articles on two different countries.

5.1 Step One

The first step in such a process is to find a threshold value that is appropriate for the given domain. Pages on the first 80 articles (Abkhazia to Japan) were used to represent the average standard deviation value for a document all on one subject while documents that drift in subject were represented by documents that were made of the first half of an article on one country and the second half of the article on the next country, starting with the first article.

The average standard deviation values of the first 80 articles can be seen in Table 5.1. The average value for these documents is 0.0483. The average standard deviation values for the mixed articles can be seen in Table 5.2. The average value for these documents is 0.1149. To get the threshold, we compute the midpoint of the average standard deviation values for single subject and the average standard deviation values for mixed subject documents: 0.0816.

5.2 Step Two

The threshold value can be used to classify the remaining articles. Two groups are used here: the first is the remaining 76 single-subject documents (from Jordan to Zimbabwe). The other group is made up of the first half of each document combined with the second half of each document below it, starting with the second document. The average standard deviation values for each document is computed, and each document is classified according to whether it is above or below the threshold; documents with values above the threshold are said to drift in subject while documents with values below the threshold are said to remain on subject.

Table 5.3 shows the values for one-subject documents; values in bold were misclassified. Table 5.4 shows the values for two subject documents; values in bold were misclassified. In the one-subject documents, three documents out of 76 were misclassified; in the two-subject documents, 8 documents out of 78 were misclassified.

5.3 Why Were Some Misclassified?

5.3.1 Iran/Iraq

The document composed of half of Iran and half of Iraq's articles was given a low standard deviation value for similarity. This is understandable if one looks through the articles and sees that both contain sections for the Iran-Iraq war and both articles make references to the other article's country.

Table 5.1: Average standard deviation values of one-subject documents

Subject	Avg SD	Subject	Avg SD
Abkhazia	0.0581	Croatia	0.0682
Afghanistan	0.0572	Cuba	0.0416
Albania	0.0239	Cyprus	0.0349
Algeria	0.0283	Czech Republic	0.0483
American Samoa	0.0292	Denmark	0.0725
Andorra	0.0309	Dominican Republic	0.0359
Angola	0.0349	East Timor	0.0492
Argentina	0.0472	Ecuador	0.0474
Armenia	0.0542	Egypt	0.0443
Aruba	0.0376	El Salvador	0.0398
Australia	0.0354	Estonia	0.0348
Austria	0.0329	Ethiopia	0.0688
Azerbaijan	0.0380	Fiji	0.0466
Bahamas	0.0319	Finland	0.0335
Bahrain	0.0460	France	0.0317
Bangladesh	0.0442	Gambia	0.0479
Barbados	0.0396	Georgia	0.0621
Belarus	0.0530	Germany	0.0513
Belgium	0.0481	Ghana	0.0594
Belize	0.0434	Greece	0.0502
Benin	0.0508	Greenland	0.0590
Bermuda	0.0364	Grenada	0.0598
Bhutan	0.0382	Guam	0.0227
Bolivia	0.0599	Guatemala	0.0839
Bosnia and Herzegovina	0.0525	Guinea	0.0706
Botswana	0.1416	Guyana	0.0542
Brazil	0.0434	Haiti	0.0541
Bulgaria	0.0468	Honduras	0.0329
Burma	0.0615	Hong Kong	0.0343
Cambodia	0.0484	Hungary	0.0553
Cameroon	0.0358	Iceland	0.0334
Canada	0.0400	India	0.0341
Central African Republic	0.0631	Indonesia	0.0441
Chad	0.0413	Iran	0.0513
Chile	0.0642	Iraq	0.0572
China	0.0747	Ireland	0.0552
Colombia	0.0664	Israel	0.0387
Congo DR	0.0596	Italy	0.0366
Costa Rica	0.0307	Jamaica	0.0339
Côte d'Ivoire	0.0664	Japan	0.0490

Table 5.2: Average standard deviation values of two-subject documents

Subject	Avg SD	Subject	Avg SD
Abkhazia/Afghanistan	0.1751	Jamaica/Japan	0.1162
Albania/Algeria	0.0975	Jordan/Kazakhstan	0.1404
American Samoa/Andorra	0.1142	Kenya/Korea, North	0.1809
Angola/Argentina	0.0268	Korea, South/Kosovo	0.1825
Armenia/Aruba	0.1467	Kuwait/Kyrgyzstan	0.1165
Australia/Austria	0.0934	Laos/Latvia	0.1384
Azerbaijan/Bahamas	0.0357	Lebanon/Macedonia	0.0856
Bahrain/Bangladesh	0.1190	Madagascar/Malaysia	0.0790
Barbados/Belarus	0.1210	Mali/Malta	0.1056
Belgium/Belize	0.0725	Mauritania/Mexico	0.0305
Benin/Bermuda	0.0930	Moldova/Mongolia	0.1460
Bhutan/Bolivia	0.1157	Montenegro/Morocco	0.1231
Bosnia and H./Botswana	0.1443	Mozambique/Nambia	0.1579
Brazil/Bulgaria	0.1191	Nepal/Netherlands	0.1481
Burma/Cambodia	0.1175	New Zealand/Nicaragua	0.1946
Cameroon/Canada	0.1378	Niger/Nigeria	0.0775
Central African Republic/Chad	0.0765	Norway/Oman	0.1017
Chile/China	0.0959	Pakistan/Panama	0.1318
Colombia/Congo DR	0.1180	Papua New Guinea/Paraguay	0.1150
Costa Rica/Côte d'Ivoire	0.1411	Peru/Philippines	0.1056
Croatia/Cuba	0.0363	Poland/Portugal	0.1189
Cyprus/Czech Republic	0.1312	Puerto Rico/Qatar	0.1473
Denmark/Dominican Republic	0.2060	Romania/Russia	0.0993
East Timor/Ecuador	0.1238	Rwanda/Saudi Arabia	0.1536
Egypt/El Salvador	0.1232	Senegal/Serbia	0.0607
Estonia/Ethiopia	0.1716	Sierra Leone/Singapore	0.1412
Fiji/Finland	0.0557	Slovakia/Slovenia	0.1221
France/Gambia	0.0614	Somalia/South Africa	0.1634
Georgia/Germany	0.1376	Spain/Sri Lanka	0.2040
Ghana/Greece	0.0628	Sudan /Sweden	0.1276
Greenland/Grenada	0.0990	Switzerland/Syria	0.1207
Guam /Guatemala	0.1445	Taiwan/Tanzania	0.0898
Guinea/Guyana	0.1235	Thailand/Tunisia	0.1032
Haiti/Honduras	0.0837	Turkey/Uganda	0.0549
Hong Kong/Hungary	0.2122	U.A.E./United Kingdom	0.0951
Iceland/India	0.1246	United States/Uruguay	0.0732
Indonesia/Iran	0.0923	Uzbekistan/Venezuela	0.0740
Iraq/Ireland	0.0875	Vietnam/Yemen	0.1168
Israel/Italy	0.1370	Zambia/Zimbabwe	0.0483

Table 5.3: Average standard deviation values of one-subject documents (misclassified documents in bold)

Subject	**Avg SD**	**Subject**	**Avg SD**
Jordan	0.0279	Poland	0.0543
Kazakhstan	0.0585	Portugal	0.0340
Kenya	0.0434	Puerto Rico	0.0590
Korea, North	0.0399	Qatar	0.0534
Korea, South	0.0381	Romania	0.0440
Kosovo	0.0459	Russia	0.0437
Kuwait	0.0371	Rwanda	0.0580
Kyrgyzstan	0.0332	Saudi Arabia	0.0444
Laos	0.0540	Senegal	0.0671
Latvia	0.0249	Serbia	0.0609
Lebanon	0.0517	Sierra Leone	0.0518
Macedonia	0.0444	Singapore	0.0408
Madagascar	0.0538	Slovakia	0.0672
Malaysia	0.0725	Slovenia	0.0753
Mali	**0.0870**	Somalia	0.0521
Malta	0.0532	South Africa	0.0662
Mauritania	0.0539	**Spain**	**0.0820**
Mexico	0.0651	Sri Lanka	0.0231
Moldova	0.0726	Sudan	0.0677
Mongolia	0.0495	Sweden	0.0494
Montenegro	0.0310	Switzerland	0.0680
Morocco	0.0345	Syria	0.0549
Mozambique	0.0376	Taiwan	0.0423
Nambia	0.0534	Tanzania	0.0280
Nepal	**0.0981**	Thailand	0.0369
Netherlands	0.0577	Tunisia	0.0618
New Zealand	0.0517	Turkey	0.0551
Nicaragua	0.0422	Uganda	0.0454
Niger	0.0594	United Arab Emirates	0.0368
Nigeria	0.0451	United Kingdom	0.0393
Norway	0.0552	United States	0.0497
Oman	0.0519	Uruguay	0.0485
Pakistan	0.0535	Uzbekistan	0.0399
Panama	0.0246	Venezuela	0.0275
Papua New Guinea	0.0327	Vietnam	0.0362
Paraguay	0.0570	Yemen	0.0402
Peru	0.0552	Zambia	0.0747
Philippines	0.0519	Zimbabwe	0.0390

Table 5.4: Average standard deviation values of two-subject documents (misclassified documents in bold)

Subject	Avg SD	Subject	Avg SD
Afghanistan/Albania	0.1243	Japan/Jordan	0.1232
Algeria/American Samoa	0.1252	Kazakhstan/Kenya	0.1434
Andorra/Angola	0.1069	**Korea, N./Korea, S.**	**0.0675**
Argentina/Armenia	0.1685	Kosovo/Kuwait	0.2033
Aruba/Australia	0.1390	Kyrgyzstan/Laos	0.1125
Austria/Azerbaijan	0.1431	Latvia/Lebanon	0.1370
Bahamas/Bahrain	0.0894	Macedonia/Madagascar	0.1406
Bangladesh/Barbados	0.0910	Malaysia/Mali	0.1216
Belarus/Belgium	0.1104	Malta/Mauritania	0.0911
Belize/Benin	0.0946	Mexico/Moldova	0.0960
Bermuda/Bhutan	0.1308	Mongolia/Montenegro	0.1111
Bolivia/Bosnia and H.	0.1224	Morocco/Mozambique	0.1030
Botswana/Brazil	0.1407	**Nambia/Nepal**	**0.0814**
Bulgaria/Burma	0.1912	Netherlands/New Zealand	0.1364
Cambodia/Cameroon	0.1102	Nicaragua/Niger	0.1235
Canada/Central African R.	0.1162	Nigeria/Norway	0.1068
Chad/Chile	0.0927	Oman/Pakistan	0.1093
China/Colombia	0.1678	Panama/Papua New Guinea	0.1792
Congo DR/Costa Rica	0.1329	**Paraguay/Peru**	**0.0707**
Côte d'Ivoire/Croatia	0.1015	Philippines/Poland	0.1314
Cuba/Cyprus	0.1567	Portugal/Puerto Rico	0.1715
Czech Republic/Denmark	0.1260	**Qatar/Romania**	**0.0628**
Dominican Rep./East Timor	0.1233	Russia/Rwanda	0.1160
Ecuador/Egypt	0.0966	Saudi Arabia/Senegal	0.0914
El Salvador/Estonia	0.1533	Serbia/Sierra Leone	0.2067
Ethiopia/Fiji	0.1221	Singapore/Slovakia	0.0954
Finland/France	0.1145	Slovenia/Somalia	0.1167
Gambia/Georgia	**0.0273**	South Africa/Spain	0.1793
Germany/Ghana	**0.0648**	Sri Lanka/Sudan	0.1610
Greece/Greenland	0.1635	Sweden/Switzerland	0.1791
Grenada/Guam	0.1206	Syria/Taiwan	0.1211
Guatemala/Guinea	0.0884	Tanzania/Thailand	0.1297
Guyana/Haiti	**0.0791**	Tunisia/Turkey	0.0984
Honduras/Hong Kong	0.1938	Uganda/U.A.E.	0.1014
Hungary/Iceland	0.1829	United Kingdom/U. S.	0.1222
India/Indonesia	0.0946	Uruguay/Uzbekistan	0.1435
Iran/Iraq	**0.0660**	Venezuela/Vietnam	0.1149
Ireland/Israel	0.1930	Yemen/Zambia	0.0918
Italy/Jamaica	0.1127	Zimbabwe/Abkhazia	0.1362

5.3.2 North Korea/South Korea

The document composed of half of North Korea and half of South Korea's articles was also given a low standard deviation value for similarity. This is understandable because both countries have the same key term, "Korea," and reference the other article's country frequently, which leads to a high overall similarity level.

5.3.3 Paraguay/Peru

The average standard deviation value of the Paraguay/Peru document was 0.0707; this was not too far below our threshold, but the document was misclassified nonetheless. It turns out these documents contained many of the same terms as they were both on South American countries that shared some of the same neighbors.

5.3.4 The Others

Of the five remaining two-subject documents that were misclassified, Guyana/Haiti and Nambia/Nepal just barely missed our threshold value. There is an interesting trend in the remaining three misclassified two-subject documents that may explain what happened with them: one subject is represented by much more text than the other.

For instance, the Gambia/Georgia document gave a very low value; however, the Gambia section was less than 20% of the document. Remember: the documents are first divided into five parts. When one subject is less than one fifth of the total document, it can not produce high similarity values when it is compared with the other sections, only low ones. This causes a low standard deviation in all of the comparisons when that section is used as a query. Perhaps for these three documents, dividing into five subdocuments was not the best choice and a higher number should have been used.

Additionally, three of the one-subject documents were misclassified. Spain just barely missed our threshold value. Nepal and Mali were further away, but still below the average value for two-subject documents. This can happen when there are distinct sections in the article that use many terms unique to that section.

5.4 Conclusion

Using the average standard deviation values of similarity scores proved to be a reliable method; 96% of the single-subject documents were classified correctly along with 89% of the two-subject documents.

Chapter 6

Cosine Wikipedia Test

Now, let's run the same test using the more straightforward method of computing coherence that Brown attempted in his TDT program report. In this test, I will divide the documents from the previous chapter into two parts of equal length and compute the cosine of the angle between the vectors of the first and second half with one another; just as before, half of the documents that drift in subject and half of the documents that stay on subject will be used to find a threshold value for classification that will test the accuracy of the method on the remaining documents.

When using the cosine of the angle between the vectors method on a single document, an important question to ask is: how do we get our IDF values? We do not want to have to resort to calculating IDF values from a large corpus of data; one big advantage of the method proposed in this paper is the fact that it does not rely on such a corpus. Having to use a body of text for computing IDF values each time would make the program larger, slower, and unable to perform when it encounters terms that are not in the corpus. When we divide a document into two parts, computing IDF scores is useless since if a term is in both parts its IDF score will be log $\frac{2}{2}$= 0. Since we can't use TF*IDF vectors, we will just use TF vectors for this test.

6.1 Step One

Just as before, the first step in classifying the documents is to determine a threshold level on a sample of the documents. We will use the same samples as before. Table 6.1 shows the cosine of the angle between vectors values of the first set of single subject documents while

Table 6.2 shows the cosine of the angle between vectors values of the first set of two-subject documents. Their averages are 0.2912 and 0.2578, respectively. The threshold value we will use for classification is the midpoint of these values: 0.2745.

6.2 Step Two

Using the threshold value of 0.2745, we will classify a document as being on one subject if its cosine of the angle between vectors value is above the threshold or on two subjects if its value is below the threshold. Tables 6.3 and 6.4 show the results for classification for single and double subject documents, respectively. Misclassified documents are in bold.

6.3 Results

Using the cosine of the angle between the vectors method, 15 out of 76 of the one-subject documents were misclassified while 27 out of 78 of the two-subject documents were misclassified for accuracy ratings of 80% and 65%, respectively. On the same data, my method achieved accuracy ratings of 96% for single-subject documents and 89% for two-subject documents; my method was clearly more effective at classifying these documents.

Interestingly, much of the misclassification by my method seemed to make sense in that a document showed low coherence scores if it drifted, but the two subjects contained similar sections and terminology. However, the misclassification of the cosine of the angle between the vectors method appears to be more random; without being able to use IDF scores, arbitrary words can push similarity values up and down.

Table 6.1: Cosine of the angle values of one-subject documents

Subject	**Cosine**	**Subject**	**Cosine**
Abkhazia	0.3092	Croatia	0.2367
Afghanistan	0.3163	Cuba	0.3325
Albania	0.2852	Cyprus	0.2519
Algeria	0.3012	Czech Republic	0.3070
American Samoa	0.3034	Denmark	0.2544
Andorra	0.2314	Dominican Republic	0.3094
Angola	0.2744	East Timor	0.2972
Argentina	0.2969	Ecuador	0.2551
Armenia	0.3103	Egypt	0.3376
Aruba	0.2669	El Salvador	0.2894
Australia	0.2655	Estonia	0.3316
Austria	0.2889	Ethiopia	0.2909
Azerbaijan	0.2504	Fiji	0.2803
Bahamas	0.2525	Finland	0.3214
Bahrain	0.2646	France	0.3157
Bangladesh	0.2737	Gambia	0.2953
Barbados	0.2779	Georgia	0.3315
Belarus	0.2915	Germany	0.2669
Belgium	0.2788	Ghana	0.2702
Belize	0.2859	Greece	0.2666
Benin	0.2603	Greenland	0.2764
Bermuda	0.3011	Grenada	0.2840
Bhutan	0.2559	Guam	0.2708
Bolivia	0.3330	Guatemala	0.2874
Bosnia and Herzegovina	0.3022	Guinea	0.2883
Botswana	0.3149	Guyana	0.2614
Brazil	0.3197	Haiti	0.2698
Bulgaria	0.3091	Honduras	0.2863
Burma	0.3533	Hong Kong	0.3164
Cambodia	0.2874	Hungary	0.2978
Cameroon	0.2846	Iceland	0.3142
Canada	0.3256	India	0.2502
Central African Republic	0.3110	Indonesia	0.3001
Chad	0.2832	Iran	0.2766
Chile	0.3246	Iraq	0.3096
China	0.3124	Ireland	0.3007
Colombia	0.3318	Israel	0.3203
Congo DR	0.2745	Italy	0.2858
Costa Rica	0.3120	Jamaica	0.2496
Côte d'Ivoire	0.3248	Japan	0.2636

Table 6.2: Cosine of the angle values of two-subject documents

Subject	Cosine	Subject	Cosine
Abkhazia/Afghanistan	0.2424	Jamaica/Japan	0.2366
Albania/Algeria	0.2668	Jordan/Kazakhstan	0.2689
American Samoa/Andorra	0.2063	Kenya/Korea, North	0.2736
Angola/Argentina	0.2519	Korea, South/Kosovo	0.2719
Armenia/Aruba	0.2653	Kuwait/Kyrgyzstan	0.2269
Australia/Austria	0.2306	Laos/Latvia	0.2574
Azerbaijan/Bahamas	0.2197	Lebanon/Macedonia	0.2796
Bahrain/Bangladesh	0.2287	Madagascar/Malaysia	0.2516
Barbados/Belarus	0.2558	Mali/Malta	0.2468
Belgium/Belize	0.3023	Mauritania/Mexico	0.2747
Benin/Bermuda	0.2518	Moldova/Mongolia	0.2387
Bhutan/Bolivia	0.2648	Montenegro/Morocco	0.2323
Bosnia and Herzegovina/Botswana	0.2807	Mozambique/Nambia	0.2868
Brazil/Bulgaria	0.2600	Nepal/Netherlands	0.2490
Burma/Cambodia	0.2772	New Zealand/Nicaragua	0.2905
Cameroon/Canada	0.2803	Niger/Nigeria	0.2725
Central African Republic/Chad	0.2413	Norway/Oman	0.2349
Chile/China	0.2661	Pakistan/Panama	0.2688
Colombia/Congo DR	0.2781	Papua New Guinea/Paraguay	0.2448
Costa Rica/Côte d'Ivoire	0.2741	Peru/Philippines	0.2424
Croatia/Cuba	0.2745	Poland/Portugal	0.2331
Cyprus/Czech Republic	0.2343	Puerto Rico/Qatar	0.2833
Denmark/Dominican Republic	0.2652	Romania/Russia	0.2829
East Timor/Ecuador	0.1980	Rwanda/Saudi Arabia	0.2521
Egypt/El Salvador	0.2461	Senegal/Serbia	0.2551
Estonia/Ethiopia	0.2468	Sierra Leone/Singapore	0.2590
Fiji/Finland	0.2601	Slovakia/Slovenia	0.2100
France/Gambia	0.2547	Somalia/South Africa	0.2591
Georgia/Germany	0.2406	Spain/Sri Lanka	0.2701
Ghana/Greece	0.2461	Sudan /Sweden	0.3127
Greenland/Grenada	0.2262	Switzerland/Syria	0.2725
Guam /Guatemala	0.2823	Taiwan/Tanzania	0.3018
Guinea/Guyana	0.2520	Thailand/Tunisia	0.2840
Haiti/Honduras	0.2259	Turkey/Uganda	0.2829
Hong Kong/Hungary	0.2549	U.A.E./United Kingdom	0.2788
Iceland/India	0.2465	United States/Uruguay	0.2884
Indonesia/Iran	0.2778	Uzbekistan/Venezuela	0.2216
Iraq/Ireland	0.2398	Vietnam/Yemen	0.2755
Israel/Italy	0.2234	Zambia/Zimbabwe	0.2974

Table 6.3: Cosine of the angle values of one-subject documents (misclassified documents in bold)

Subject	**Cosine**	**Subject**	**Cosine**
Jordan	0.3239	Poland	0.2892
Kazakhstan	0.3228	Portugal	0.2944
Kenya	**0.2622**	Puerto Rico	0.2961
Korea, North	0.3168	Qatar	0.3047
Korea, South	0.2914	Romania	0.3059
Kosovo	0.3378	Russia	0.2862
Kuwait	0.2783	Rwanda	0.3471
Kyrgyzstan	**0.2717**	Saudi Arabia	0.2871
Laos	**0.2536**	**Senegal**	**0.2450**
Latvia	0.2872	Serbia	0.3209
Lebanon	0.2788	Sierra Leone	0.3231
Macedonia	0.3309	Singapore	0.3130
Madagascar	**0.2574**	**Slovakia**	**0.2721**
Malaysia	0.3219	**Slovenia**	**0.2364**
Mali	**0.2126**	Somalia	0.3493
Malta	**0.2617**	South Africa	0.3139
Mauritania	0.2778	Spain	0.3389
Mexico	0.2903	Sri Lanka	0.3094
Moldova	0.3000	Sudan	0.3471
Mongolia	0.2776	Sweden	0.3416
Montenegro	0.2821	Switzerland	0.2853
Morocco	0.2876	Syria	0.3158
Mozambique	0.2989	Taiwan	0.3634
Nambia	0.3034	**Tanzania**	**0.2729**
Nepal	0.3035	Thailand	0.3025
Netherlands	0.2983	Tunisia	0.2928
New Zealand	0.2804	Turkey	0.2865
Nicaragua	0.2867	Uganda	0.2934
Niger	0.3060	United Arab Emirates	0.3102
Nigeria	0.2929	United Kingdom	0.3233
Norway	0.2902	United States	0.2863
Oman	0.2828	**Uruguay**	**0.2706**
Pakistan	**0.2743**	Uzbekistan	0.2900
Panama	**0.2436**	**Venezuela**	**0.2516**
Papua New Guinea	0.3149	Vietnam	0.3172
Paraguay	**0.2500**	Yemen	0.3013
Peru	0.3011	Zambia	0.3353
Philippines	0.2970	Zimbabwe	0.3240

Table 6.4: Cosine of the angle values of two-subject documents (misclassified documents in bold)

Subject	**Cosine**	**Subject**	**Cosine**
Afghanistan/Albania	**0.2982**	**Japan/Jordan**	**0.2797**
Algeria/American Samoa	0.2739	Kazakhstan/Kenya	0.2523
Andorra/Angola	0.2415	Korea, North/Korea, South	0.2665
Argentina/Armenia	0.**3038**	Kosovo/Kuwait	0.2644
Aruba/Australia	0.2230	Kyrgyzstan/Laos	0.2669
Austria/Azerbaijan	0.2482	Latvia/Lebanon	0.2311
Bahamas/Bahrain	0.2465	Macedonia/Madagascar	0.2551
Bangladesh/Barbados	0.2178	Malaysia/Mali	0.2507
Belarus/Belgium	0.2246	Malta/Mauritania	0.2560
Belize/Benin	0.2340	**Mexico/Moldova**	**0.2912**
Bermuda/Bhutan	0.2426	Mongolia/Montenegro	0.2471
Bolivia/Bosnia and Herzegovina	0.2222	Morocco/Mozambique	0.2555
Botswana/Brazil	**0.2808**	Nambia/Nepal	0.2321
Bulgaria/Burma	**0.2957**	Netherlands/New Zealand	0.2308
Cambodia/Cameroon	0.2291	**Nicaragua/Niger**	**0.2774**
Canada/Central African Rep.	**0.2862**	Nigeria/Norway	0.2470
Chad/Chile	**0.2836**	Oman/Pakistan	0.2727
China/Colombia	**0.3015**	Panama/Papua New Guinea	0.2317
Congo DR/Costa Rica	0.2561	Paraguay/Peru	0.2209
Côte d'Ivoire/Croatia	0.2228	Philippines/Poland	0.2395
Cuba/Cyprus	**0.2827**	Portugal/Puerto Rico	0.2538
Czech Republic/Denmark	0.2624	**Qatar/Romania**	**0.2842**
Dominican Republic/East Timor	**0.2827**	Russia/Rwanda	0.2550
Ecuador/Egypt	0.2480	**Saudi Arabia/Senegal**	**0.2770**
El Salvador/Estonia	**0.2918**	Serbia/Sierra Leone	0.2263
Ethiopia/Fiji	**0.2882**	**Singapore/Slovakia**	**0.2858**
Finland/France	**0.2945**	**Slovenia/Somalia**	**0.2829**
Gambia/Georgia	**0.2818**	**South Africa/Spain**	**0.2857**
Germany/Ghana	**0.2868**	**Sri Lanka/Sudan**	**0.2996**
Greece/Greenland	0.2181	**Sweden/Switzerland**	**0.2944**
Grenada/Guam	0.2426	**Syria/Taiwan**	**0.2873**
Guatemala/Guinea	0.2038	Tanzania/Thailand	0.2259
Guyana/Haiti	0.2547	Tunisia/Turkey	0.2405
Honduras/Hong Kong	0.2446	Uganda/U.A.E.	0.2414
Hungary/Iceland	**0.2865**	**United Kingdom/U. S.**	**0.2924**
India/Indonesia	0.2316	Uruguay/Uzbekistan	0.2364
Iran/Iraq	0.2612	Venezuela/Vietnam	0.2525
Ireland/Israel	0.2688	Yemen/Zambia	0.2646
Italy/Jamaica	0.2237	Zimbabwe/Abkhazia	0.2623

Chapter 7

Lower Bound For Accurate Classification

It was previously mentioned in this thesis that as document length increases, so does the accuracy of classifying documents based on coherence level. In this chapter, I will demonstrate this by comparing the results of documents that were run through the algorithm which were similar in subject, but different in length. I will also give a general suggestion of what might be a lower bound of length for a document to be classified correctly by the method proposed in this thesis. In this chapter, test documents were retrieved from the website Wikipedia on eleven distinct subjects; the documents were retrieved from the website on June 27, 2008. Every document was either 1,000 or 3,000 words long; the first 1,000 or 3,000 terms from the subject's Wikipedia page were used. First, let's compare documents on the two different lengths that are all on one subject.

Table 7.1 shows the average standard deviation of similarity scores for our eleven subjects on documents that are either 1,000 or 3,000 words long. At the bottom of Table 7.1 we have the average standard deviation scores for all documents of a given length. One thing that should be noted is how abnormally high the 1,000 word document for physics is ranked. After looking through the Wikipedia page for physics, one notices that the subject is so broad that the page is full of sections that are so distinct they have little in common with one another. Perhaps the method would not be incorrect to categorize the Wikipedia page on physics as being on more than one subject. Here, the average values for all pages' standard deviation scores is very similar. The 1,000 words document is slightly higher but that can be attributed to the score of the physics article.

Table 7.1: Average standard deviation values of one-subject documents

Subject	**1000 Words**	**3000 Words**
Wine	0.0331	0.0453
Beatles	0.0544	0.0884
Cows	0.0671	0.0569
Mars	0.0515	0.0477
Motorcycles	0.0601	0.0441
Physics	0.1142	0.0504
Robotics	0.0425	0.0306
Shakespeare	0.0476	0.0461
UGa	0.0517	0.0541
Ukraine	0.0517	0.0762
Water	0.0559	0.0363
AVERAGE:	**0.0573**	**0.0524**

Table 7.2: Average standard deviation values of two-subject documents

Subject	**1000 Words**	**3000 Words**
Water/Ukraine	0.1086	0.1315
Physics/Robotics	0.1284	0.0966
Shakespeare/Cows	0.1486	0.1560
Beatles/Mars	0.0907	0.1036
UGa/Motorcycles	0.0915	0.0787
Water/Mars	0.0616	0.0445
Beatles/Ukraine	0.0784	0.0904
UGa/Shakespeare	0.0866	0.1140
Physics/Cows	0.1346	0.1471
Water/Robotics	0.1240	0.1258
AVERAGE:	**0.1053**	**0.1088**

Table 7.3: Average standard deviation values of three-subject documents

Subject	**1000 Words**	**3000 Words**
Mars/Beatles/Ukraine	0.0651	0.1289
UGa/Beatles/Robotics	0.0650	0.0923
UGa/Water/Shakespeare	0.0867	0.1176
Cows/Water/Motorcycles	0.1299	0.1591
Cows/Mars/Physics	0.1774	0.1726
Beatles/Physics/Mars	0.0844	0.1371
Physics/Mars/Ukraine	0.1228	0.0872
Physics/UGa/Ukraine	0.0296	0.0839
Ukraine/UGa/Water	0.0373	0.2098
Robotics/Ukraine/UGa	0.0362	0.0917
AVERAGE:	**0.0834**	**0.1280**

Table 7.2 shows the average standard deviation values for documents that are on two subjects, changing in the middle, of 1,000 and 3,000 words. The 3,000 word document that starts out discussing water and then changes to a discussion of Mars is abnormally low. After a perusal of the document, I discovered that the Wikipedia section on Mars that was used actually discussed the possibility of water on the planet. The repetition of the term "water" and other similar terms in the Mars section caused the similarity value of the entire document to give low standard deviation value. Again, here the average standard deviation values for documents of 1,000 and 3,000 words are very similar. It seems as if, when there is one subject change, 1,000 and 3,000 word documents are equally capable of classification based on coherence.

Table 7.3 shows the average standard deviation values of documents that change subject twice (at around a third and two thirds of the way in) of length 1,000 and 3,000 words. Here, the average standard deviation of similarity is much lower for documents of 1,000 words than it is for documents of 3,000 words. Documents of 1,000 words that change subject twice seem to be greatly prone to misclassification. It is important to note that in those documents each subject is represented by only around 333 terms. It seems as if that is not a large enough sample of words related to the subjects to calculate similarity values accurately. This is a much worse performance than the 1,000 word document that changed subject once where subjects were represented by 500 terms. It seems as if the system starts to break down when the subjects have less than 500 terms to describe them.

Chapter 8

Documents With Multiple Topic Changes

8.1 Possible Solution

How would this system handle a lengthy document that changed subject multiple times? There are many possible courses of action to take for such a situation. For instance, instead of dividing the document into five subdocuments, one could divide the document into several subdocuments and draw inferences, based on the similarity levels between them, of where different subjects might exist.

One method mirrors the algorithm in this thesis directly: it involves dividing a document up into sections and then treating each section as if it were a unique document, dividing it up into five subdocuments and calculating its average standard deviation value as we have done before. Depending on the length of the original document, the document can be divided up into 2, 4, 8, or more sections. This can be seen in Figure 8.1.

If a document changes its subject with every sentence, it will be classified as being all on one subject by my system. Perhaps that is not incorrect: the document could be said to be all on the same subject: no subject at all.

8.2 Example

Let's look at the values for a 16,000 word document made up of four sections of 4,000 words from Wikipedia articles on cows, wine, Mars, and motorcycles. Table 8.1 shows the average standard deviation values of different sections of a document made up of 4 different subjects. The entire document's score is 0.1201. This value is high because when the four subjects are divided into five parts for measuring similarity, some of the words from the four subjects spill into two different sections, and when these sections are compared the similarity value is higher than for the other three sections. This increases the standard deviation value. The average standard deviation score when the document is divided into two pieces is 0.1563. This is a high value because each piece is two different subjects.

The average standard deviation score when the document is divided into four pieces is 0.0472. This value is low because now each section whose value is being computed is all on one subject. The average standard deviation score when the document is divided into eight

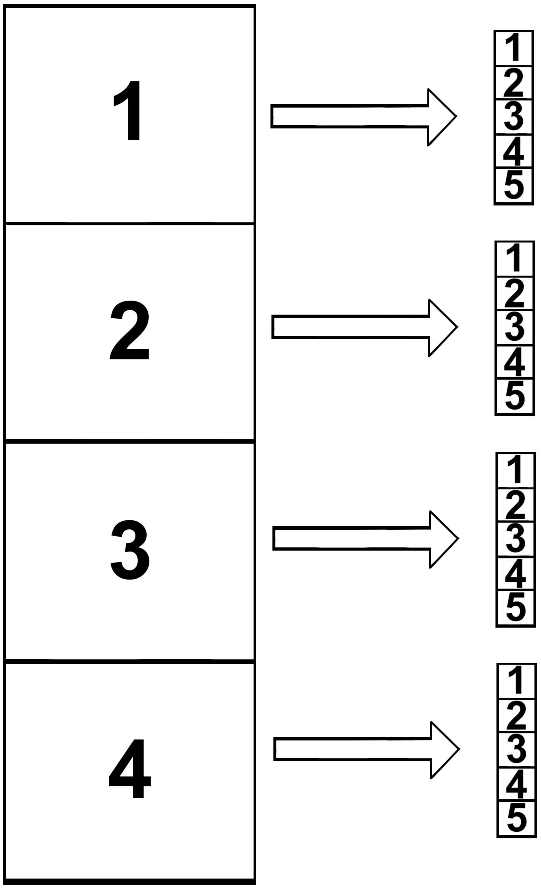

Figure 8.1: Division of one document into four documents with five subdocuments each

Table 8.1: Different sections of 16,000 word document and their standard deviation values

Section	**Avg std dev**
Entire Document	0.1201
First Half	0.1781
Second Half	0.1344
First Quarter	0.0587
Second Quarter	0.0420
Third Quarter	0.0437
Fourth Quarter	0.0444
First Eighth	0.0591
Second Eighth	0.0293
Third Eighth	0.0373
Fourth Eighth	0.0298
Fifth Eighth	0.0343
Sixth Eighth	0.0426
Seventh Eighth	0.0673
Eighth Eighth	0.0316

pieces is 0.0414. Here, the value is still low because the sections are still all on one subject each, just with half as many words as the previous set.

This demonstrates how the method described in this thesis could potentially handle large documents with multiple topic changes, not just documents that change topic once. This is just a general suggestion; to handle this problem accurately one must look at the domain of interest and test for an appropriate strategy to extract information accurately.

Chapter 9

Future Work

This thesis describes a new concept that is unfinished; much work could be done to improve it and apply it to different domains. For example, many documents that were misclassified were close to the threshold value. Perhaps instead of a standard classification, the project should take more of a fuzzy logic approach where documents are given scores on a scale of zero to one of how much drift is measured. This would show a distinction between articles that just barely miss the threshold and documents that were very far away from the threshold.

There may be other information in the documents that can be combined with the average standard deviation value to lead to more accurate classification. I considered other values, such as the average IDF score, which did not seem to show any difference for documents that were of different coherence levels; but there may be other values that can show distinction.

Bibliography

[1] Brown, R. D.; Pierce, T.; Yang, Y; and Carbonell, J. G. (2000). Link detection results and analysis. *1999 National Institute of Standards and Technology Topic Detection and Tracking Workshop.* Web: http://citeseer.ist.psu.edu/brown00link.html.

[2] Covington, M. A.; He, C.; Brown, C.; Naci, L.; McClain, J. T.; Fjordbak, B. S.; Semple, J.; and Brown, J. (2005). Schizophrenia and the structure of language: the linguists view. *Schizophrenia Research, 77*(1):85–98.

[3] Elvevaag, B.; Foltz, P.W.; Weinberger, D.R.; and Goldberg, T.E. (2006). Quantifying incoherence in speech: an automated methodology and novel application to schizophrenia. *Schizophrenia Research, 93*(1–3), 304–316.

[4] Gaffen, D. (2007, October 31). Google's surge would make casey kasem proud. *The Wall Street Journal.* Web: http://blogs.wsj.com/marketbeat/2007/10/31/googles-surge-would-make-casey-kasem-proud.

[5] Grossman, D.; and Frieder, O. (2004). *Information retrieval.* Dordrecht: Springer.

[6] Hearst, M. (1997, March). TextTiling: segmenting text into multi-paragraph subtopic passages. *Computational Linguistics*, *23*(1): 33–64.

[7] Landauer, T.K.; Foltz P.W.; and Laham, D. (1998). Introduction to latent semantic analysis. *Discourse Processes. 25*: 259–284.

[8] Mann, W. C.; and Thompson, S. A. (1988). Rhetorical structure theory: toward a functional theory of text organization. *Text, 8*(3), 243–281.

[9] Maskery, S.; Zhang, Y.; Hu, H.; Shriver, C.; Hooke, J.; and Liebman, M. (2006). Caffeine intake, race, and risk of invasive breast cancer lessons learned from data mining a clinical database. *19th IEEE Symposium on Computer-Based Medical Systems (CBMS'06)*, 714–718.

[10] McCarthy, M. (1988). Some vocabulary patterns in conversation. *Vocabulary and language teaching* London: Longman.

[11] Mulbregt, P. van; Carp, I.; Gillick, L; Lowe, S; and Yamron, J. (1998, December). Text segmentation and topic tracking on broadcast news via a hidden Markov model approach. In *Proceedings ICSLP-98*, paper 0116.

[12] Salton, G.; Wong, A; and Yang, C.S. (1975). A vector space model for automatic indexing. *Communications of the ACM, 18*(11), 613–620.

[13] Stubbs, M. (1983). *Discourse analysis: the sociolinguistic analysis of natural language.* Oxford: Blackwell.

[14] Todd, R. W. (1997). Textual patterns in teachers eliciting. *RELC Journal, 28*(1), 1–14.

[15] Todd, R. W.; Thienpermpool, P.; and Keyuravong, S. (2004). Measuring the coherence of writing using topic-based analysis. *Assessing Writing, 9,* 85–104.

[16] Wayne, C. (2000, May). Multilingual topic detection and tracking: successful research enabled by corpora and evaluation. In *Proceedings of the Second International Language Resources and Evaluation Conference*, Athens, Greece.

Appendix A

Mars Document

This is the document on Mars used in Section 3.3 (3,000 words of Wikipedia's Mars entry, retrieved from Wikipedia on June 27, 2008).

> Physical characteristics
>
> Size comparison of terrestrial planets (left to right): Mercury, Venus, Earth, and Mars.
>
> Mars has approximately half the radius of Earth and only one-tenth the mass, being less dense, but its surface area is only slightly less than the total area of Earth's dry land.[3] While Mars is larger and more massive than Mercury, Mercury has a higher density. This results in a slightly stronger gravitational force at Mercury's surface. The red-orange appearance of the Martian surface is caused by iron(III) oxide, more commonly known as hematite, or rust.[8]
>
> Geology
>
> Main article: Geology of Mars
>
> Based on orbital observations and the examination of the Martian meteorite collection, the surface of Mars appears to be composed primarily of basalt. Some evidence suggests that a portion of the Martian surface is more silica-rich than typical basalt, and may be similar to andesitic rocks on Earth; however, these observations may also be explained by silica glass. Much of the surface is deeply covered by a fine iron(III) oxide dust that has the consistency of talcum powder.[citation needed]
>
> Rock strewn surface imaged by Mars Pathfinder
>
> Although Mars has no intrinsic magnetic field, observations show that parts of the planet's crust have been magnetized and that alternating polarity reversals of its dipole field have occurred. This paleomagnetism of magnetically susceptible minerals has properties that are very similar to the alternating bands found on the ocean floors of Earth. One theory, published in 1999 and re-examined in October 2005 (with the help of the Mars Global Surveyor), is that these bands demonstrate plate tectonics on Mars 4 billion years ago, before the planetary dynamo ceased to function and caused the planet's magnetic field to fade away.[9]

Current models of the planet's interior imply a core region about 1,480 kilometres in radius, consisting primarily of iron with about 14–17% sulfur. This iron sulfide core is partially fluid, and has twice the concentration of the lighter elements than exist at Earth's core. The core is surrounded by a silicate mantle that formed many of the tectonic and volcanic features on the planet, but now appears to be inactive. The average thickness of the planet's crust is about 50 km, with a maximum thickness of 125 km.[10] Earth's crust, averaging 40 km, is only a third as thick as Mars crust relative to the sizes of the two planets.

The geological history of Mars can be split into many epochs, but the following are the three main ones:

Noachian epoch (named after Noachis Terra): Formation of the oldest extant surfaces of Mars, 3.8 billion years ago to 3.5 billion years ago. Noachian age surfaces are scarred by many large impact craters. The Tharsis bulge volcanic upland is thought to have formed during this period, with extensive flooding by liquid water late in the epoch.

Hesperian epoch (named after Hesperia Planum): 3.5 billion years ago to 1.8 billion years ago. The Hesperian epoch is marked by the formation of extensive lava plains.

Amazonian epoch (named after Amazonis Planitia): 1.8 billion years ago to present. Amazonian regions have few meteorite impact craters but are otherwise quite varied. Olympus Mons formed during this period along with lava flows elsewhere on Mars. A major geological event occurred on Mars on February 19, 2008, and was caught on camera by the Mars Reconnaissance Orbiter. Images capturing a spectacular avalanche of materials thought to be fine grained ice, dust, and large blocks are shown to have detached from a 2,300-foot (701 m) high cliff. Evidence of the avalanche is present in the dust clouds left above the cliff afterwards.[11]

Recent studies support a theory, first proposed in the 1980s, that Mars was struck by an Pluto-sized meteor about four billion years ago. The event, thought to be the cause of the Martian hemispheric dichotomy, distorted the planet's northern hemisphere.[12][13]

Hydrology

Photo of microscopic rock forms indicating past signs of water, taken by Opportunity Liquid water cannot exist on the surface of Mars with its present low atmospheric pressure, except at the lowest elevations for short periods[14][15] but water ice is in no short supply, with two polar ice caps made largely of ice.[16] In March 2007, NASA announced that the volume of water ice in the south polar ice cap, if melted, would be sufficient to cover the entire planetary surface to a depth of 11 metres.[17] Additionally, an ice permafrost mantle stretches down from the pole to latitudes of about 60.[16]

Much larger quantities of water are thought to be trapped underneath Mars's thick cryosphere, only to be released when the crust is cracked through volcanic action.[clarify] The largest such release of liquid water is thought to have occurred when the Valles Marineris formed early in Mars's history, enough water being released to form the massive outflow channels. A smaller but more recent event of the same kind may have occurred when the Cerberus Fossae chasm opened about 5 million years ago, leaving a supposed sea of frozen ice still visible today on the Elysium Planitia centered at Cerberus Palus.[18] However, the morphology of this region is more consistent with the ponding of lava flows causing a superficial similarity to ice flows.[19] These lava flows probably draped the terrain established by earlier catastrophic floods of Athabasca Valles.[20] Significantly rough surface texture at decimeter (dm) scales, thermal inertia comparable to that of the Gusev plains, and hydrovolcanic cones are consistent with the lava flow hypothesis.[20] Furthermore, the stoichiometric mass fraction of H2O in this area to tens of centimeter depths is only $\tilde{4}$%,[21] easily attributable to hydrated minerals[22] and inconsistent with the presence of near-surface ice.

More recently the high resolution Mars Orbiter Camera on the Mars Global Surveyor has taken pictures which give much more detail about the history of liquid water on the surface of Mars. Despite the many giant flood channels and associated tree-like network of tributaries found on Mars there are no smaller scale structures that would indicate the origin of the flood waters. It has been suggested that weathering processes have denuded these, indicating the river valleys are old features. Higher resolution observations from spacecraft like Mars Global Surveyor also revealed at least a few hundred features along crater and canyon walls that appear similar to terrestrial seepage gullies. The gullies tend to be in the highlands of the southern hemisphere and to face the Equator; all are poleward of 30 latitude.[23] The researchers found no partially degraded (i.e. weathered) gullies and no superimposed impact craters, indicating that these are very young features.

In a particularly striking example (see image) two photographs, taken six years apart, show a gully on Mars with what appears to be new deposits of sediment. Michael Meyer, the lead scientist for NASA's Mars Exploration Program, argues that only the flow of material with a high liquid water content could produce such a debris pattern and colouring. Whether the water results from precipitation, underground or another source remains an open question.[24] However, alternative scenarios have been suggested, including the possibility of the deposits being caused by carbon dioxide frost or by the movement of dust on the Martian surface.[25][26] Further evidence that liquid water once existed on the surface of Mars comes from the detection of specific minerals such as hematite and goethite, both of which sometimes form in the presence of water.[27]

Nevertheless, some of the evidence believed to indicate ancient water basins and flows has been negated by higher resolution studies taken at resolution about 30 cm by the Mars Reconnaissance Orbiter.[28]

Geography

Main articles: Geography of Mars, List of mountains on Mars, and List of craters on Mars

See also: Category:Surface features of Mars

This approximate true-color image, taken by the Mars Exploration Rover Opportunity, shows the view of Victoria Crater from Cape Verde. It was captured over a three-week period, from October 16 – November 6, 2006.

Although better remembered for mapping the Moon, Johann Heinrich Mdler and Wilhelm Beer were the first “areographers”. They began by establishing once and for all that most of Mars surface features were permanent, and determining the planet’s rotation period. In 1840, Mdler combined ten years of observations and drew the first map of Mars. Rather than giving names to the various markings, Beer and Mdler simply designated them with letters; Meridian Bay (Sinus Meridiani) was thus feature “a.”[29] Today, features on Mars are named from a number of sources. Large albedo features retain many of the older names, but are often updated to reflect new knowledge of the nature of the features. For example, Nix Olympica (the snows of Olympus) has become Olympus Mons (Mount Olympus).[30]

Mars equator is defined by its rotation, but the location of its Prime Meridian was specified, as was Earth’s (at Greenwich), by choice of an arbitrary point; Mdler and Beer selected a line in 1830 for their first maps of Mars. After the spacecraft Mariner 9 provided extensive imagery of Mars in 1972, a small crater (later called Airy-0), located in the Sinus Meridiani (“Middle Bay” or “Meridian Bay”), was chosen for the definition of 0.0 longitude to coincide with the original selection.

Olympus Mons

Since Mars has no oceans and hence no “sea level”, a zero-elevation surface or mean gravity surface also had to be selected. Zero altitude is defined by the height at which there is 610.5 Pa (6.105 mbar) of atmospheric pressure. This pressure corresponds to the triple point of water, and is about 0.6% of the sea level surface pressure on Earth (.006 atm).[31]

The dichotomy of Martian topography is striking: northern plains flattened by lava flows contrast with the southern highlands, pitted and cratered by ancient impacts. The surface of Mars as seen from Earth is thus divided into two kinds of areas, with differing albedo. The paler plains covered with dust and sand rich in reddish iron oxides were once thought of as Martian “continents” and given names like Arabia Terra (land of Arabia) or Amazonis Planitia (Amazonian plain). The dark features were thought to be seas, hence their names Mare Erythraeum, Mare Sirenum and Aurorae Sinus. The largest dark feature seen from Earth is Syrtis Major.[32]

The shield volcano, Olympus Mons (Mount Olympus), at 26 km is the highest known mountain in the Solar System. It is an extinct volcano in the vast upland region Tharsis, which contains several other large volcanoes. It is over three times the height of Mount Everest which in comparison stands at only 8.848 km.

Mars is also scarred by a number of impact craters: a total of 43,000 craters with a diameter of 5 km or greater have been found.[33] The largest of these is the Hellas impact basin, a light albedo feature clearly visible from Earth.[34] Due to the smaller mass of Mars, the probability of an object colliding with the planet is about half that of the Earth. However, Mars is located closer to the asteroid belt, so it has an increased chance of being struck by materials from that source. Mars is also more likely to be struck by short-period comets, i.e., those that lie within the orbit of Jupiter.[35] In spite of this, there are far fewer craters on Mars compared with the Moon because Mars's atmosphere provides protection against small meteors. Some craters have a morphology that suggests the ground was wet when the meteor impacted.

The large canyon, Valles Marineris (Latin for Mariner Valleys, also known as Agathadaemon in the old canal maps), has a length of 4000 km and a depth of up to 7 km. The length of Valles Marineris is equivalent to the length of Europe and extends across one-fifth the circumference of Mars. By comparison, the Grand Canyon on Earth is only 446 km long and nearly 2 km deep. Valles Marineris was formed due to the swelling of the Tharis area which caused the crust in the area of Valles Marineris to collapse. Another large canyon is Ma'adim Vallis (Ma'adim is Hebrew for Mars). It is 700 km long and again much bigger than the Grand Canyon with a width of 20 km and a depth of 2 km in some places. It is possible that Ma'adim Vallis was flooded with liquid water in the past.[36]

THEMIS image of cave entrances on Mars

Images from the Thermal Emission Imaging System (THEMIS) aboard NASA's Mars Odyssey orbiter have revealed seven possible cave entrances on the flanks of the Arsia Mons volcano.[37] The caves, named Dena, Chloe, Wendy, Annie, Abbey, Nikki and Jeanne after loved ones of their discoverers, are collectively known as the "seven sisters."[38] Cave entrances measure from 100 m to 252 m wide and they are believed to be at least 73 m to 96 m deep. Because light does not reach the floor of most of the caves, it is likely that they extend much deeper than these lower estimates and widen below the surface. Dena is the only exception; its floor is visible and was measured to be 130 m deep. The interiors of these caverns may be protected from micrometeoroids, UV radiation, solar flares and high energy particles that bombard the planet's surface.[39] Some researchers have suggested that this protection makes the caves good candidates for future efforts to find liquid water and signs of life. Mars has two permanent polar ice caps: the northern one at Planum Boreum and the southern one at Planum Australe.

Atmosphere Main article: Atmosphere of Mars

Mars's thin atmosphere, visible on the horizon in this low-orbit photo. Mars lost its magnetosphere 4 billion years ago, so the solar wind interacts directly with the Martian ionosphere, keeping the atmosphere thinner than it would otherwise be by stripping away atoms from the outer layer. Both Mars Global Surveyor and Mars Express have detected these ionised atmospheric particles trailing off into space behind Mars.[40][41] The atmosphere of Mars is now relatively thin. Atmospheric pressure on the surface varies from around 30 Pa (0.03 kPa) on Olympus Mons to over 1155 Pa (1.155 kPa) in the depths of Hellas Planitia, with a mean surface level pressure of 600 Pa (0.6 kPa). This is less than 1% of the surface pressure on Earth (101.3 kPa). Mars's mean surface pressure equals the pressure found 35 km above the Earth's surface. The scale height of the atmosphere, about 11 km, is higher than Earth's (6 km) due to the lower gravity.

The atmosphere on Mars consists of 95% carbon dioxide, 3% nitrogen, 1.6% argon, and contains traces of oxygen and water.[3] The atmosphere is quite dusty, containing particulates about 1.5 m in diameter which give the Martian sky a tawny color when seen from the surface.[42]

Several researchers claim to have detected methane in the Martian atmosphere with a concentration of about 10 ppb by volume.[43][44] Since methane is an unstable gas that is broken down by ultraviolet radiation, typically lasting about 340 years in the Martian atmosphere,[45] its presence would indicate a current or recent source of the gas on the planet. Volcanic activity, cometary impacts, and the presence of methanogenic microbial life forms are among possible sources. It was recently pointed out that methane could also be produced by a non-biological process called serpentinization[b] involving water, carbon dioxide, and the mineral olivine, which is known to be common on Mars.[46]

During a pole's winter, it lies in continuous darkness, chilling the surface and causing 25–30% of the atmosphere to condense out into thick slabs of CO2 ice (dry ice).[47] When the poles are again exposed to sunlight, the frozen CO2 sublimes, creating enormous winds that sweep off the poles as fast as 400 km/h. These seasonal actions transport large amounts of dust and water vapor, giving rise to Earth-like frost and large cirrus clouds. Clouds of water-ice were photographed by the Opportunity rover in 2004.[48]

Climate Main article: Climate of Mars

Mars from Hubble Space Telescope October 28, 2005 with dust storm visible. Of all the planets, Mars's seasons are the most Earth-like, due to the similar tilts of the two planets' rotational axes. However, the lengths of the Martian seasons are about twice those of Earth's, as Mars greater distance from the Sun leads to the Martian year being about two Earth years in length. Martian surface temperatures vary from lows of about -140 C (-220 F) during the polar winters to highs of up to 20 C (68 F) in summers.[14] The wide range in temperatures is due to the thin atmosphere which cannot store much solar heat, the low atmospheric pressure, and the low thermal inertia of Martian soil.[49]

If Mars had an Earth-like orbit, its seasons would be similar to Earth's because its axial tilt is similar to Earth's. However, the comparatively large eccentricity of the Martian orbit has a significant effect. Mars is near perihelion when it is summer in the southern hemisphere and winter in the north, and near aphelion when it is winter in the southern hemisphere and summer in the north. As a result, the seasons in the southern hemisphere are more extreme and the seasons in the northern are milder than would otherwise be the case. The summer temperatures in the south can be up to 30 C (54 F) warmer than the equivalent summer temperatures in the north.[50]

Mars's northern ice cap.

Mars also has the largest dust storms in our Solar System. These can vary from a storm over a small area, to gigantic storms that cover the entire planet. They tend to occur when Mars is closest to the Sun, and have been shown to increase the global temperature.[51]

The polar caps at both poles consist primarily of water ice. However, there is dry ice present on their surfaces. Frozen carbon dioxide (dry ice) accumulates as a thin layer about one metre thick on the north cap in the northern winter only, while the south cap has a permanent dry ice cover about eight metres thick.[52] The northern polar cap has a diameter of about 1,000 kilometres during the northern Mars

(3,000 words of document reached)

Appendix B

Code

B.1 Driver class (compiled using Microsoft Visual C# 2008)

```
using System;
using System.Collections;
using System.Collections.Generic;
using System.Windows.Forms;
using System.IO;
using System.Text.RegularExpressions;
using CoherenceMeasure;


class Test
{
    public static string[] Tokenize(string equation)
    {
        Regex RE = new Regex(@"\n");
        return (RE.Split(equation));
    }

    public static string[] Tokenize2(string equation)
    {
        Regex RE = new Regex(@" ");
        return (RE.Split(equation));
    }

    public static void Main()
    {
        int Sections = 5;  //number of "subdocuments" to compare,
        //this should be divisible by 5
        int cuts = 2;
        //number of times we run a given document though
        //if this number is 2 we just run the entire document
        //once, if it is 3 we run the document, then the first half,
        //then the second half. If it is 7 we run the document,
        // first half, second half, first quarter... etc until the fourth
        //quarter...

        ArrayList al = new ArrayList();
        int counter = 0;

        try
        {
            //reading the text file in
            using (StreamReader sr = new StreamReader("TestFile.txt"))
            {
                String line2;
                line2 = sr.ReadToEnd();

                foreach (string token in Tokenize2(line2))
                {
                    al.Add(token.Trim());
                    counter++;
                }
            }
        }
        catch (Exception e)
        {
            Console.WriteLine("The file could not be read:");
            Console.WriteLine(e.Message);
        }

        ArrayList documentCuts = new ArrayList();

        double placemarker1 = 1;
        double placemarker2 = 1;
        double placemarker3 = 1;

        ArrayList pm1 = new ArrayList();
        ArrayList pm2 = new ArrayList();

        for (int j = 1; j < cuts; j++)
        {
            ArrayList AlTemp = new ArrayList();
            placemarker3 = j;
            documentCuts.Add(AlTemp);
            pm1.Add(placemarker1);
            pm2.Add(placemarker2);

            if (placemarker2 >= placemarker1)
            {
                placemarker2 = 1;
                placemarker1 = placemarker1 * 2;
            }
            else
            {
                placemarker2++;
            }
        }

        //run this loop once if we want to run the entire document once,
        //multiple times if we want to look at different sections
        for (int bbb = 0; bbb < pm1.Count; bbb++)
        {
            double start = 0;
            int intstart = 0;
            ArrayList tempal = new ArrayList();
            double docLength = (al.Count / (double)pm1[bbb]);
            double finish = 0;
            int intfinish = 0;
            finish = ((double)pm2[bbb] / (double)pm1[bbb]) * (al.Count);

            if ((double)pm2[bbb] != 1)
            {
                 start = ((double)pm2[bbb - 1] / (double)pm1[bbb]) * al.Count;
            }

            intfinish = (int)finish;
            intstart = (int)start;

            for (int a = intstart; a < intfinish; a++)
            {
                tempal.Add(al[a]);
            }

            ArrayList al2 = new ArrayList();
            SimCo s = new SimCo();

            //getting rid of punctuation, making it all lowercase
            al2 = s.makeTermsSame(tempal);
            al2 = s.makeTermsSame(al2);
            ArrayList al3 = new ArrayList();

            //getting rid of blank entries
            for (int aa = 0; aa < al2.Count; aa++)
```

```
{
    if (al2[aa].Equals(""))
    {
    }
    else
    {
        al3.Add(al2[aa]);
    }
}

System.Collections.ArrayList[] all;
all = new ArrayList[Sections];
for (int i = 0; i < Sections; i++)
{
    all[i] = new ArrayList();
}

int textlength = al3.Count;
int div = textlength / Sections;
int textplace = 0;
int div2 = div;

for (int count = 0; count < Sections; count++)
{
    for (int c = textplace; c < div2; c++)
    {
        all[count].Add(al3[c]);
    }

    textplace = textplace + div;
    div2 = div2 + div;
}

ArrayList uniqueTerms = new ArrayList();
uniqueTerms = s.getTermList(al2);
ArrayList IDFS = new ArrayList();

//get IDF scores
IDFS = s.getIDFList(al2, uniqueTerms, Sections);

int fifth = Sections / 5;
double denom = 1;
double num = 1;
double stddev = 1;
ArrayList stddevs = new ArrayList();
```

```
double[] vals = new double[Sections - 1];
int valsPlace = 0;
ArrayList sims = new ArrayList();

//get first standard deviation value
for (int i = 0; i < fifth; i++)
{
    for (int h = 0; h < Sections; h++)
    {
        denom = s.getSim(all[i], all[i], uniqueTerms, IDFS);
        num = s.getSim(all[i], all[h], uniqueTerms, IDFS);

        if (i != h)
        {
            sims.Add(num / denom);
            vals[valsPlace] = (num / denom);
            valsPlace++;
        }
    }
    valsPlace = 0;
    stddev = s.getStandardDev(vals);
    stddevs.Add(stddev);
}
//get second standard deviation value
for (int i = fifth; i < (fifth * 2); i++)
{
    for (int h = 0; h < Sections; h++)
    {
        denom = s.getSim(all[i], all[i], uniqueTerms, IDFS);
        num = s.getSim(all[i], all[h], uniqueTerms, IDFS);

        if (i != h)
        {
            sims.Add(num / denom);
            vals[valsPlace] = (num / denom);
            valsPlace++;
        }
    }
    valsPlace = 0;
    stddev = s.getStandardDev(vals);
    stddevs.Add(stddev);
}
//get third standard deviation value
for (int i = (fifth * 3); i < (fifth * 4); i++)
{
    for (int h = 0; h < Sections; h++)
    {
        denom = s.getSim(all[i], all[i], uniqueTerms, IDFS);
        num = s.getSim(all[i], all[h], uniqueTerms, IDFS);

        if (i != h)
        {
            sims.Add(num / denom);
            vals[valsPlace] = (num / denom);
            valsPlace++;
        }
    }
    valsPlace = 0;
    stddev = s.getStandardDev(vals);
    stddevs.Add(stddev);
}
//get forth standard deviation value
for (int i = (fifth * 4); i < (fifth * 5); i++)
{
    for (int h = 0; h < Sections; h++)
    {
        denom = s.getSim(all[i], all[i], uniqueTerms, IDFS);
        num = s.getSim(all[i], all[h], uniqueTerms, IDFS);

        if (i != h)
        {
            sims.Add(num / denom);
            vals[valsPlace] = (num / denom);
            valsPlace++;
        }
    }

    valsPlace = 0;
    stddev = s.getStandardDev(vals);
    stddevs.Add(stddev);
}

double sumStdDev = 0;
double avgSD = 0;

//computing the average of the standard deviations
for (int k = 0; k < stddevs.Count; k++)
{
    sumStdDev = sumStdDev + (double)stddevs[k];
}
```

```
            avgSD = (sumStdDev / stddevs.Count);
            Console.WriteLine("Average std dev is " + avgSD);
        }
    }
}
```

B.2 SimCo class (compiled using Microsoft Visual C# 2008)

```
using System;
using System.Collections.Generic;
using System.Text;
using System.Collections;


namespace CoherenceMeasure
{
    class SimCo
    {
        public string newLineTerm1(string newline)
        {
            int f = newline.IndexOf("\n");
            string newline2 = newline.Substring(0, f - 1);
            string newline3 = newline.Substring(f + 1);
            newline3 = newline3.TrimStart();
            return newline2;
        }

        public string newLineTerm2(string newline)
        {
            int f = newline.IndexOf("\n");
            string newline2 = newline.Substring(0, f - 1);
            string newline3 = newline.Substring(f + 1);
            newline3 = newline3.TrimStart();
            return newline3;
        }

        //get the standard deviation of a given arraylist
        public double getStandardDev(double[] l)
        {
            double stddev = 0;
            double sum = 0;
            for (int i = 0; i < l.Length; i++)
            {
                sum = sum + l[i];
            }

            double avg;
            avg = sum / l.Length;
            ArrayList diffs = new ArrayList();
            double tempdiff = 0;

            for (int i = 0; i < l.Length; i++)
            {
                tempdiff = (l[i]) - avg;
                diffs.Add(tempdiff * tempdiff);
            }

            double sumdiffs = 0;

            for (int i = 0; i < diffs.Count; i++)
            {
                sumdiffs = sumdiffs + ((double)(diffs[i]));
            }

            double avgdiffs = (sumdiffs / (diffs.Count - 1));
            stddev = Math.Sqrt(avgdiffs);

            return stddev;
        }

        //make all terms lowercase, remove punctuation
        public ArrayList makeTermsSame(ArrayList ar)
        {
            ArrayList terms = new ArrayList();
            string term2 = " ";
            bool isTerm2 = false;

            for (int aa = 0; aa < ar.Count; aa++)
            {
                string tempstring = (string)ar[aa];
                tempstring = tempstring.ToLower();
                tempstring = tempstring.Replace(".", "");
                if (tempstring.EndsWith("."))
                {
                    tempstring = tempstring.Replace(".", "");
                }
```

```
if (tempstring.EndsWith("’"))
{
    tempstring = tempstring.Replace("’", "");
}

if (tempstring.EndsWith(":"))
{
    tempstring = tempstring.Replace(":", "");
}

if (tempstring.EndsWith(")"))
{
    tempstring = tempstring.Replace(")", "");
}


if (tempstring.StartsWith("("))
{
    tempstring = tempstring.Replace("(", "");
}

if (tempstring.EndsWith("?"))
{
    tempstring = tempstring.Replace("?", "");
}

if (tempstring.EndsWith("!"))
{
    tempstring = tempstring.Replace("!", "");
}

if (tempstring.EndsWith(","))
{
    tempstring = tempstring.Replace(",", "");
}

if (tempstring.Contains("\n"))
{
    SimCo s = new SimCo();
    string term1 = s.newLineTerm1(tempstring);
    term2 = s.newLineTerm2(tempstring);
    isTerm2 = true;
    tempstring = term1;
}
```

```
        if (tempstring.EndsWith(";"))
        {
            tempstring = tempstring.Replace(";", "");
        }

        if (tempstring.EndsWith("\""))
        {
            tempstring = tempstring.Replace("\"", "");
        }

        terms.Add(tempstring);

        if (isTerm2 == true)
        {
            terms.Add(term2);
            isTerm2 = false;
        }
    }
    return terms;
}

//get all unique terms in one arraylist
public ArrayList getTermList(ArrayList doc)
{
    ArrayList terms = new ArrayList();
    terms.Add(doc[0]);

    for (int mm = 0; mm < doc.Count; mm++)
    {
        bool isPresent = false;

        for (int rr = 0; rr < terms.Count; rr++)
        {
            if (doc[mm].Equals(terms[rr]))
            {
                isPresent = true;
            }
        }
        if (isPresent == false)
        {
            terms.Add(doc[mm]);
        }
    }
    return terms;
```

```
}

//get list of IDF values that corresponds with list of unique terms
public ArrayList getIDFList(ArrayList doc, ArrayList uniqueTerms,
int Sections)
{
    ArrayList IDFS = new ArrayList();
    ArrayList freqs = new ArrayList();
    System.Collections.ArrayList[] all;

    all = new ArrayList[Sections];
    for (int i = 0; i < Sections; i++)
    {
        all[i] = new ArrayList();
    }

    int textlength = doc.Count;
    int div = textlength / Sections;
    int textplace = 0;
    int div2 = div;

    for (int count = 0; count < Sections; count++)
    {
        for (int c = textplace; c < div2; c++)
        {
            all[count].Add(doc[c]);
        }
        textplace = textplace + div;
        div2 = div2 + div;
    }

    for (int i = 0; i < uniqueTerms.Count; i++)
    {
        int tempFreq = 1;
        bool isIN = false;
        bool foundAlready = false;

        for (int c = 0; c < Sections; c++)
        {
            for (int d = 0; d < all[c].Count; d++)
            {
                if (uniqueTerms[i].Equals(all[c][d]))
                {
                    isIN = true;
                }
            }
            if (isIN == true && foundAlready == true)
            {
                tempFreq++;
                isIN = false;
            }
            if (isIN == true && foundAlready == false)
            {
                foundAlready = true;
            }
        }
        freqs.Add(tempFreq);
    }

    for (int rr = 0; rr < freqs.Count; rr++)
    {
        double tempIDF1 = 1;
        double tempIDF2 = 1;
        tempIDF1 = (int)freqs[rr];
        tempIDF2 = (Sections / tempIDF1);
        tempIDF2 = Math.Log10(tempIDF2);
        IDFS.Add(tempIDF2);
    }
    return IDFS;
}

//get the similarity score of two text inputs
public double getSim(ArrayList query, ArrayList doc, ArrayList
uniqueTerms, ArrayList IDFS)
{
    ArrayList queryMatrix = new ArrayList();
    ArrayList docMatrix = new ArrayList();

    for (int i = 0; i < uniqueTerms.Count; i++)
    {
        double tempValue = 0;

        for (int c = 0; c < query.Count; c++)
        {
            if (uniqueTerms[i].Equals(query[c]))
            {
                double tempIDF = (double)IDFS[i];
                tempValue = tempValue + tempIDF;
            }
        }
```

```
                queryMatrix.Add(tempValue);
            }

            for (int i = 0; i < uniqueTerms.Count; i++)
            {
                double tempValue = 0;

                for (int c = 0; c < doc.Count; c++)
                {
                    if (uniqueTerms[i].Equals(doc[c]))
                    {
                        double tempIDF = (double)IDFS[i];
                        tempValue = tempValue + tempIDF;
                    }
                }

                docMatrix.Add(tempValue);
            }

            double SimCo = 0;

            for (int f = 0; f < queryMatrix.Count; f++)
            {
                double temp1 = (double)queryMatrix[f];
                double temp2 = (double)docMatrix[f];
                double temp3 = 0;

                temp3 = temp1 * temp2;
                SimCo = SimCo + temp3;
            }
            return SimCo;
        }
    }
}
```