Leximancy describes the art of science of distilling knowledge from an analysis of a large body of text. Patterns of symbols, for example, must be identified and their properties used to infer the power of their content. This is essentially the process of abstraction from data to knowledge. Each concept must be a set of symbols, something which cannot be named. A name, a description, can be called a concept only if it is practical. If not, it cannot be named. The number of documents is almost countless, and it is hard to imagine how we could practically use them.
Table Of Contents

Section 1: An Introduction To Leximancer .............................................. 4
  What is Leximancer? ........................................................................ 4
  An Example: Romeo and Juliet ......................................................... 4
  Applications of Leximancer ............................................................. 10
Section 2: Running Leximancer .............................................................. 11
  System Requirements ..................................................................... 11
  Supported File Types .................................................................... 11
  Installing Leximancer ..................................................................... 11
  Starting Leximancer ...................................................................... 12
  Creating a New Project .................................................................. 12
  Creating an Automatic Map of your Documents .............................. 13
  Phases of Processing in Leximancer ................................................ 14
  Using the Map to Understand and Browse your Documents .......... 16
Section 3: Content Analysis .................................................................. 19
  What is Content Analysis? .............................................................. 19
  Types of Content Analysis .............................................................. 20
Section 4: Concepts and the Conceptual Map in Leximancer .................. 24
  What is a Concept? ....................................................................... 24
  Concept Seed Words ..................................................................... 25
  Concepts and the Relevancy Metric ................................................ 25
  How are Concepts Learned? ............................................................ 26
Section 5: Text Preprocessing and Automatic Concept Identification ....... 28
  Configuring Preprocessing (Simple Menu Options) ......................... 29
  Configuring Automatic Concept Identification (Simple Menu Options) 31
  Tutorial Questions ........................................................................ 32
Section 6: Concept Editing and Thesaurus Learning ............................... 35
  Concept Editing ............................................................................ 35
  Tutorial Questions ......................................................................... 38
  Thesaurus Learning ....................................................................... 39
  Tutorial Questions ......................................................................... 42
  The Thesaurus .............................................................................. 43
  Tutorial Exercises ......................................................................... 43
Section 7: Classification and Indexing ..................................................... 44
Section 1: An Introduction To Leximancer

What is Leximancer?

Leximancer is a data-mining tool that can be used to analyse the content of collections of textual documents and to visually display the extracted information. The information is displayed by means of a conceptual map that provides a birds eye view of the material, representing the main concepts contained within the text and how they are related. Apart from viewing the conceptual structure of the information, this map allows users to perform a directed search of the documents in order to explore instances of the concepts or their interrelations. That is, Leximancer provides a means of both quantifying and displaying the conceptual structure of a document set, as well as a means of using this information to explore interesting conceptual features.

An Example: Romeo and Juliet

After processing the script “Romeo and Juliet” by William Shakespeare, the following map was extracted:
This map displays five important sources of information about the text:

- The main concepts discussed in the document set
- The relative frequency of each concept
- How often concepts co-occur within the text
- The centrality of each concept
- The similarity in contexts in which the concepts occur – thematic groups.

These sources of information will be discussed in the following sections.

**Concepts and their Frequency**

Leximancer automatically extracts the most important concepts discussed in the documents being processed. In *Romeo and Juliet*, such concepts include “Romeo”, “Juliet”, “Tybalt”, “death” and “love.” On the map, the strength of a concept’s label relates to its relative frequency in the text, varying from black (highly frequent) to light grey (infrequent). The size of the concept point indicates its connectedness. The colour indicates its thematic group. Thematic groups are also surrounded by circles. In the full interface, a list of the concepts ranked by their frequency can be viewed by clicking on the “Ranked Concept List” below the map:

In the Romeo and Juliet map, the five most frequent concepts were “love”, “Romeo”, “Juliet”, “man”, “night”, and “death.” Inspecting the frequently occurring concepts can thus by itself sometimes provide insightful overviews of the content of the text.
**Relationships between Concepts**

Apart from measuring the frequency of occurrence of the main extracted concepts, Leximancer also measures how often concepts occur close together within the text. Such direct relationships between the concepts can be revealed on the map by left-clicking on concepts. For example, in the large map, clicking on *Romeo* reveals associations with *Juliet*, *dead*, *man*, *Tybalt*, *time* and *love*. The strength of direct associations between concepts is indicated by the brightness of the link between concepts on the map, as well as bar graph on the interface:

The co-occurrence between concepts in the text is an important measure of the degree of association between them.

**Centrality of Concepts**

On the map, it is possible to change the number of concepts that are visible by using the top slider below the map. This allows users to view only the most central concepts contained within the text. The centrality of a concept is defined in terms of the number of times a concept co-occurs with other defined concepts. That is, a concept will be central if it is frequent (as it is more likely to co-occur with other concepts), and appears in contexts surrounded by the other concepts that Leximancer has extracted.

In *Romeo and Juliet*, the most central concept is *love*, followed by *Romeo*, *night*, *Juliet*, *death* and *man*:
**Contextual Similarity**

The last source of information that is directly perceptible from the map is the contextual similarity between concepts. The map is constructed by initially placing the concepts randomly on the grid. Each concept exerts a pull on each other concept with a strength related to their co-occurrence value. That is, concepts can be thought of as being connected to each other with springs of various lengths. The more frequently two concepts co-occur, the stronger will be the force of attraction (the shorter the spring), forcing frequently co-occurring concepts to be closer on the final map. However, because there are many forces of attraction acting on each concept, it is impossible to create a map in which every concept is at the expected distance away from every other concept. Rather, concepts with similar attractions to all other concepts will become clustered together. That is, concepts that appear in similar contexts (i.e., co-occur with the other concepts to a similar degree) will appear in similar regions in the map. For example, in an analysis of political speeches, parties that talk about similar issues will appear closer together than the parties with different agendas.

An example of contextual similarity can be seen clearly in the Romeo and Juliet case. Although the concepts *eye* and *eyes* do not have a direct link in the text (they never appear together), they appear adjacent on the map due to the similarities in the
contexts in which they appear. Furthermore, these words appear clustered close together with other “luminous” words, such as *light*, and *sun* that appear in similar contexts in the play. A thematic grouping of *night*, *day*, *to-morrow*, and also *to-night* is also apparent:

**Summary**

In summary, Leximancer’s conceptual map provides three main sources of information about the content of documents:

1. The main concepts contained within the text and their relative importance;
2. The strengths of links between concepts (how often they co-occur);
3. The similarities in the context in which they occur.

As will be discussed later, these sources of information provide a powerful metric for statistically comparing different documents. Such a comparison is useful for document retrieval (e.g., to find documents similar to a given document) as well as typical content analysis applications (e.g., determining if there exist significant differences in the communicational content of suicidal versus non-suicidal patients). Furthermore, this map can be used for simply providing a bird’s eye view of a large amount of material.
**Browsing Information using Leximancer**

As stated earlier, the conceptual map that Leximancer extracts can be used to browse for instances of specific concepts and relationships. This facility allows users to directly explore the text for instances where interesting concepts occur or co-occur with other concepts. For example, clicking on Romeo in the above map will display the co-occurring concepts in a separate frame:

![Concept Map]

Clicking on the “navigate location” button beside one of the related concepts allows you to browse through the instances of where this concept co-occurs with Romeo. For example, navigating the concept “Tybalt” will display instances in the text where “Romeo” and “Tybalt” co-occur:

![Text Browser]
Thus, apart from being a powerful content analysis tool, Leximancer can be used to provide a general overview of the content of textual documents as well as a means of exploring the material in a directed fashion. Such uses range from web site navigation to research applications where the user only wishes to explore parts of the documents relating to their area of interest.

**Applications of Leximancer**

**Text Analysis**
One important application of Leximancer is in the analysis of the differences in textual content between various sources (discussed in more detail in Section 3). For example, it can be used as a tool for determining if there exist significant differences in the communicational content of suicidal versus non-suicidal patients, or to determine how issues of political speeches change over a given period of time. Apart from allowing these differences to be explored visually, Leximancer explicitly stores the data relating to the frequency of concept use and frequency of concept co-occurrence in a spreadsheet file. This file can be loaded by programs such as SPSS to statistically explore or verify such differences in textual content.

**Coding Open-Ended Surveys**
Leximancer can be used for coding the answers to open-ended questions asked in surveys. This can be used for organisational, clinical and social psychology experiments, where the differences in written or verbal answers between different groups need to be evaluated.

**Site and Archive Concept Navigation**
It is an efficient alternative to manually maintained site maps and search engines, and you can select the concepts you choose. This is also a lifesaver for navigating mountains of litigation evidence if it is in electronic form.

**Media Analysis**
Given a collection of electronic media articles from a news agency, Leximancer can rapidly show you a media profile of you or your opposition.

**Customer Relationship Management (CRM)**
Whether you are an IT service provider, or a politician, you may have a large collection of e-mail from your customers. Leximancer can show you the spectrum of current issues and concerns, including any sleepers. This is very useful for policy and campaign development.

**Students and Academics**
Leximancer is an excellent tool for supporting researchers in History, Literature, Media Studies, Sociology, Politics, or any extensive library work.

If you would like to see Leximancer in action at this stage, example maps can be found in the “Gallery” section of the Leximancer homepage, located at http://www.leximancer.com/.
Section 2: Running Leximancer

The aim of the following section is to give you an overview and hands-on introduction to using Leximancer. In this section, you will be creating a map of the book “Alice’s Adventures in Wonderland”. The book has been provided in the folder C:\Leximancer 2.xx\Tutorial\Alice_In_Wonderland (or the comparable folder where Leximancer was installed). Please replace 2.xx with your version number.

System Requirements

In order to run the Windows version of Leximancer 2, you will need the following:

- Windows 98 or above
- Approximately 100Mb of Hard Drive space per concept map

It is recommended that you have at least 128Mb of RAM

Supported File Types

Currently .doc, .html, .htm, .txt, .xml and .pdf files are supported. Files with no extension are also supported, but are treated as .txt formatted.

Generally, if there are other formats of text data you wish to process, it is recommended that you look into pre-converting them into HTML. There are quite a few filters available, such as Latex2HTML etc.

Installing Leximancer

Using Windows Explorer, double click on the file “Install.exe” from the CD provided. This will guide you through the set-up process and create an icon on your desktop from which you can run Leximancer. Generally, you can accept the defaults given during the setup process unless you wish to install Leximancer in a particular location.
Starting Leximancer

To Start Leximancer either:

- double-click on the icon “Leximancer 2.0” on your desktop
- choose “Programs⇒Leximancer 2.0⇒Leximancer 2.0” from your Start menu.

In a few seconds, a screen will appear allowing you to create new projects, or load or delete existing projects:

The aim of Leximancer is to create a conceptual map of collections of documents. Projects in Leximancer are a way of organising your information, with a single map being the product of each project. You can create several different maps of the same data set by creating different projects.

If you have not generated Leximancer projects in the past, the above project list will be empty.

Creating a New Project

When creating a new project, you will be asked to enter a project name. As a project file is created using this name, please stick to typical conventions for naming files (e.g., don't include the characters ".", ",", "*", or "/"). Names such as "alice in wonderland" and "cinderella map 1" are appropriate.

Click on “New” to create a new project. The following dialog will appear:
Type “Alice” and click on OK.

Creating an Automatic Map of your Documents

When you create a new project or open an existing project, an interface similar to the following will appear:

The above interface contains the following:

1) A main pull-down menu that enables you to save and load previous maps
2) A workspace that contains a flow diagram of the various phases of processing required to generate a conceptual map.
3) A start button, allowing you to generate an automatic map with a single click

Click on “START” to create an automatic map of your data. The following dialog will appear:
Clicking on the browse button allows you to select the files or folder full of files that you would like to process. **Note:** holding down `<ctrl>` while selecting files will allow you to select multiple files within the same folder. Once you have selected the files, clicking on OK in the above dialog will automatically create a map of your documents.

- Click on the browse button and select the folder C:\Leximancer 2.xx\Tutorial\Alice_In_Wonderland (or the comparable folder where Leximancer was installed).
- Clicking on OK in the above dialog will automatically create a map of your documents.

**Phases of Processing in Leximancer**

Once you click on “OK”, Leximancer will automatically run the following phases in sequence:

<table>
<thead>
<tr>
<th>Preprocess Text</th>
<th>(1) <strong>Text Preprocessing</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Text preprocessing is the first phase of processing that is run from Leximancer’s main menu. This phase converts the raw documents into a useful format for processing, such as identifying sentence and paragraph boundaries.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Automatic Concept Identification</th>
<th>(2) <strong>Automatic Seed Extraction</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>In this phase, important concepts are automatically identified from the text. As this stage, concepts are simply keywords, such as “dog” or “alice”.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Concept Editing</th>
<th>(3) <strong>Concept Editing</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>In this phase of processing, users have the option of deleting automatically identified concepts that are not of interest, adding extra concepts, or merging concepts that refer to the same thing.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Thesaurus Learning</th>
<th>(4) <strong>Concept Thesaurus Learning</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Concepts in Leximancer are collections of words that travel together throughout the text. For example, a term such as “rifle” may have other terms such as “ammunition” or “bullet” that frequently appear with it, and rarely appear elsewhere. The learning phase identifies such clusters of words that surround the main terms given by the preceding two phases.</td>
</tr>
</tbody>
</table>
(5) Classification
Once the concept definitions have been learned, each block of text is tagged with the names of the concepts that it contains. This process is similar to manual coding used in “content analysis.” However, the benefit of using Leximancer for this task is that it is fast (compared to the time taken for human coders), and is more objective (as opposed to humans, where there is much variability in coding performance).

(6) Mapping
The last phase of processing is “mapping” in which the conceptual map that displays the relationship between variables is constructed.

Leximancer should now be creating a map of Alice in Wonderland. On the interface, icons that are green have been completed, and those in red are yet to run. The progress bar at the bottom of the interface displays the progress of the current phase.

When all the stages have run, a map will appear on your screen.
Using the Map to Understand and Browse your Documents

When the map phase has been run successfully, a window similar to the following will appear:

Concepts on the map

- The map (1) contains the main concepts that occur within your documents, enclosed in thematic circles.
- The brightness of a concept is related to its frequency (i.e. the brighter the concept, the more often it appears in the text).
- Green concepts represent proper names (such as people or locations), whereas white concepts refer to other objects, locations, or actions.
- Use the slider (2) to change the number of concepts visible on the map. Sliding to the left will show you the most central concepts that occur within your documents. Sliding all the way left will hide the concepts.
- Use the slider (3) to change the size, or generality, of the thematic circles. Moving the slider all the way left turns off theme circles.
- A list of concepts ranked by their frequency can be revealed by clicking on “View Ranked Concept List” (4).
- Concepts are contextually clustered on the map. That is, concepts that appear together frequently in the text or in similar situations will be close together on the map.
**Browsing the Information**

On the Map:

- You can reveal hidden concepts by moving the “# of points” scrollbar (located underneath the map) to the right.
- To reveal only the most important concepts, slide this bar to the left.

**Question 1.**

- Using the scrollbar, reveal the single most important concept.

What is this concept?

__________________________________________________________________

**Question 2.**

You can also find out the most important concepts by clicking on the “View Ranked Concept List” button below the map.

- Click on the “View Ranked Concept List” Button below the map

What are the top three concepts?

1. ______________________
2. ______________________
3. ______________________

**Question 3.**

On the map, clicking on a concept will reveal the strength of co-occurrence with other concepts. This is displayed in terms of the brightness of the links. This information is also displayed in the right window, as a ranked co-occurrence list of the concept you selected.

- Find and click on the concept “Alice” on the map, and inspect the related entity list. You will need to use the slider to see the links if Alice is the only visible concept.

Looking at the ranked co-occurrence list, what are the top three concepts that co-occur with Alice?

1. ______________________
2. ______________________
3. ______________________
**Question 4.**

From the co-occurrence list, you can browse the locations in the document for where the concepts actually co-occur by clicking in the Navigate Locations button.

- By browsing the co-occurrence of “Alice” and “thought”, explain why “thought” is such a strong concept.

Under the map, there is a “View Document Summary” button. This summary will give you blocks of sentences containing the most important concepts that are contained on the map. That is, they are crucial segments of text showing you the relationship between the defined concepts.

- Close the text browser, click on the “View Document Summary” button below the map and read the paragraph provided for chapter 2.
- Use the Click Here icon beside “Summary” to view a list of important sections of text, then use the Full text option to view a particular sample where it occurs in the full text. The following dialog will appear:

The above dialog allows gives you a list of the most important sections of text in the document (in the left window), and shows you where they appear within the document (in the right widow).

- Browse through the important sections of Chapter 2 in the left window, and click on the “Full Text” links to show where they appear in the text.

When you have finished playing with the browser and the Map, you can close them down.
Section 3: Content Analysis

What is Content Analysis?

Content analysis is a research tool used for determining the presence of words or concepts in collections of textual documents. It is used for breaking down the material into manageable categories and relations that can be quantified and analyzed. These extracted measurements can be used to make valid inferences about the material contained within the text (such as the presence of propaganda), properties of the writer or speaker (such as his or her psychological state), the audience to which the material is presented, or properties of the culture of the time in which the material was written. Many approaches to content analysis have produced highly reliable and valid results.

Content analysis is an important research methodology as it can be used to analyze any form of verbal communication from written to spoken forms. Thus it can be used, for example, by social psychologists to compare groups of individuals non-invasively by analyzing their natural social interactions without the need for creating artificial settings or scenarios. Furthermore, as text documents tend to exist over long periods of time, the technique can be used to extract valuable historical and cultural insights.

As content analysis can be performed on numerous forms of data ranging from political speeches and open-ended interviews to newspaper articles and historical documents, it is invaluable to many researchers. Such uses include:

- historical analysis of political speeches
- detecting the existence and level of propaganda
- coding surveys that ask open-ended questions
- determining the psychological state of the writers
- assesses textual content against measures (e.g. censoring)
- assess cultural differences in populations
Types of Content Analysis

In general, approaches to content analysis fall into two major categories: conceptual analysis and relational analysis. In conceptual analysis, documents are measured for the presence and frequency of concepts. Such concepts can be words or phrases, or more complex definitions, such as collections of words representing each concept. Relational analysis, by contrast, measures how such identified concepts are related to each other within the documents.

One of the strengths of the Leximancer system is that it conducts both forms of analysis, measuring the presence of defined concepts in the text as well as how they are interrelated. The following sections describe both forms of content analysis, and how the Leximancer system deals with the various issues concerning them.

**Conceptual Analysis**

Conceptual analysis (also known as thematic analysis) is the most common form of content analysis. Conceptual analysis involves the detection and quantification of the presence of predefined concepts within the text. Concepts can either be explicit (i.e. particular words or phrases) or implicit (i.e. implied, but not explicitly stated in a set of predefined terms). As expected, explicit concepts are the easiest to define and measure, with the quantification of implicit concepts requiring the use of a specialized dictionaries or contextual translation rules.

**Issues in Conceptual Analysis**

Although conceptual analysis may seem pretty straightforward, there are many decisions that need to be made by the researcher, each of which highlights important issues about the approach. Some of the main issues and how they are dealt within Leximancer are discussed below.

**What is the appropriate level of analysis?**

Conceptual analysis is aimed at quantifying the presence of concepts in a given text. The first main issue raised is what exactly constitutes a concept? Coding may be attempted at many levels, including the detection of particular words or phrases, the coding of the sense of the word (taking into account synonyms or idioms), or the categorization of sentences, paragraphs or whole bodies of text.

One popular approach to defining concepts is in the creation and application of general dictionaries. Such dictionaries explicitly define the possible coding categories, providing rules for assigning words to categories or providing the actual words to code for. Although the creation of general dictionaries minimizes the need for dictionary construction and validation, they are problematic in that they are specifically tailored to particular domains.
Assumed versus Inferred Categories

One problem in using general dictionaries for conceptual analysis is that they are generally useless in coding for documents outside their domain. Also, it is often argued that assumed category schemes impose the reality of the investigator on the text rather than measuring the categories used by the writers of the text themselves. That is, using different dictionaries, multiple descriptions of the same textural reality can be extracted with vastly different results, rather than the actual description meant by the authors.

It is often argued that the categories present in the text can be inferred from the co-variation amongst the high-frequency words in the text. One of Leximancer’s main features is that it can automatically extract its own dictionary of terms for each document set using this information. That is, it is capable of inferring the concept classes that are contained within the text, explicitly extracting a thesaurus of terms for each concept. This approach also relieves the user of the task of needing to formulate their own coding scheme, allowing dictionaries to be defined automatically for any topic. Furthermore, this process can be interactive, allowing users with knowledge of the domain to influence the concept definitions, focusing on the concepts that they are interested in quantifying. The process of how concepts are automatically extracted by Leximancer is described later in this document.

How do you distinguish concepts?

One decision that commonly needs to be addressed in conceptual analysis is whether or not categorical judgments are mutually exclusive. This is a problem because many statistical analysis procedures require variables that are not confounded (i.e. correlated). However, schemes that are capable of coding for the meaning of words in the sentences (rather than coding for particular words) are flexible by necessity, often leading to ambiguities in categorical judgments. This problem is confounded by the fact that having one attribute does not necessarily exclude the possession of another. Furthermore, in many coding schemes, the qualities by which concepts are judged are continuous, reducing the reliability of category membership decisions made by human coders.

To deal with the issue of continuous attributes in concept definitions, Leximancer weights each word depending of how well it connotes the category. In this approach, sentences may contain multiple concepts, but with the sentences needing to contain enough of the crucial keywords for a concept to be identified. The weighting of each word in the definition of a concept is automatically extracted by Leximancer during its learning phase (described later). As there is a strict algorithm for describing concept membership, the coding can be done with a computer, leading to perfect scoring reliability.

Reliability

In content analysis, generally there are considered to be two forms of reliability that are pertinent: stability and reproducibility. Stability generally refers to the tendency of a coder to consistently re-code the same information in the same way over time. However, in using human coders, there are often inconsistencies that arise due to
various factors such as the ambiguity of the coding rules, ambiguities in the text, simple coding errors, or cognitive changes within the coder (such as the presence of a good essay affecting the rating of subsequent ones). As Leximancer’s approach is automated and deterministic, many such inconsistencies are avoided, leading to a high level of coding stability.

The second form of reliability, reproducibility, refers to the consistency of classification given by several coders, given the same marking scheme. In Leximancer, this issue is most relevant to the generation of the conceptual map. That is, the process of map generation is stochastic, leading to the possibility of different final positions for the extracted concepts each time the map is generated. However, although the process is stochastic, there typically exist consistent trends in the spatial positioning of concepts relative to each other. Thus, to promote a high level of reliability of the analysis, the map should be constructed several times, with the interpretations of the data being focused on the stable features.

**Relational Analysis**

Relational analysis (also known as semantic analysis) goes one step beyond conceptual analysis, measuring the relationships between the identified concepts. Approaches to relational analysis generally fall within 3 categories: affect extraction, proximity analysis and cognitive mapping. Affect extraction, which aims at providing an emotional evaluation of the emotional and psychological state of the speaker, is a quite specific form of content analysis that is beyond the scope of this document.

The second form of relational analysis, proximity analysis, measures the co-occurrence of concepts found within the text. In this approach, a length of words or sentences called a window is specified. The window is moved sequentially through the text, with concepts that occur together within the window being noted. The result of this calculation is called a concept co-occurrence matrix, in which the frequency of co-occurrence of all concepts against all others is explicitly stored. This matrix gives a convenient metric for comparing the content of different document sets. As discussed earlier, Leximancer automatically extracts this information, explicitly storing it in a spreadsheet ready for analysis. The file is called spreadsheet.txt and is generated in each project’s Map folder.

The third form of relational analysis, Cognitive Mapping, is an extension of the first two, representing the information visually for comparison. As the output of relational analysis is generally a large matrix of values, the cognitive mapping approach tries to compress this information into two dimensions, displaying the main relationships between concepts. Again, as stated earlier, this form of analysis is also performed by the Leximancer software.
Interested in Learning More about Content Analysis?

If you are interested in learning more about the issues relating to content analysis and the various techniques that are used, we recommend the following book:

Section 4: Concepts and the Conceptual Map in Leximancer

What is a Concept?

Concepts in Leximancer are collections of words that generally travel together throughout the text. For example, a concept "rifle" may contain the keywords "rifle", "telescopic", "serial", "ammunition", "stock", "western cartridge", etc. These terms are weighted according to how frequently they occur in sentences containing the concept compared to how frequently they occur elsewhere. Sentences are tagged as containing a concept if enough accumulated evidence is found. Terms are weighted so the presence of each word in a sentence provides an appropriate contribution to the accumulated evidence for the presence of a concept. That is, a sentence (or group of sentences) is only tagged as containing a concept if the accumulated evidence (the sum of the weights of the keywords found) is above a set threshold. For example, the concept “violence” may contain central keywords (such as “war” and “terrorism”) that provide strong evidence for the concept in a sentence, but also may contain a collection of peripheral items (such as “army” or “troops”) that provide weaker evidence:

Apart from detecting the overall presence of a concept in the text (which is indicated by the brightness of the concept in the conceptual map), the concept definitions are also used to determine the frequency of co-occurrence between concepts. This co-occurrence measure is what is used to generate the concept map.
Concept Seed Words

In Leximancer, the definition of each concept (i.e. the set of weighted terms) is automatically learned from the text itself. **Concept seed words** represent the starting point for the definition of such concepts, with each concept definition containing one or more such seeds. These seed words can either be provided by the user, or can be automatically extracted from the text (explained below). They are called seeds as they represent the starting point of the concept, with more terms being added to the definition through learning. Occasionally, more appropriate central terms may be discovered, pushing the seeds away from the centre of the concept definition.

Concepts and the Relevancy Metric

Concepts in Leximancer are defined as collections of terms that provide evidence for the use of the concept in the text. The terms themselves have a “relevancy value” that determines how central each word is to the concept. For example, in the illustration below, *word 1* will have a high relevancy value to *concept X* as it appears often in sentences containing the concept, and rarely in sentences not containing the concept. In contrast, *word 2* will have a low relevancy to *concept X* as it co-occurs rarely with the concept, but frequently in the rest of the text.

Leximancer identifies concept seeds (the starting point of concept definitions) by looking for candidates through the most frequently appearing words in the text that are not “stop words” (such as “and” or “very”). The potential seeds are next evaluated by calculating the number of strongly relevant terms for each seed candidate. This measurement selects important and central concepts that characterise the text. For example, the word “dog” may be considered a seed as it has many strongly related items such as “barks”, “fleas” etc., that often co-occurs with it and occur not so often alone.
How are Concepts Learned?

During learning, Leximancer generates a thesaurus of terms for each concept. This learning is an iterative process in which the collection of terms defining a concept is updated. The aim of concept learning is to discover clusters of words which, when taken together as a concept, maximise the relevancy values of all the other words in the document. For example, the concept definition (“dog”, “hound”, “puppy”) is a strong concept definition as a large collection of other words (such as “fleas”, “bites”, “barks”, etc) frequently co-occurs with at least one of the concept keywords, and not so frequently elsewhere. This concept definition is considered stronger than a concept containing just “dog”, as the co-occurring words (“fleas”, “bites” and “barks”) may also appear frequently in other contexts (i.e. close to “hound” and “puppy”).

Learning occurs as follows:

Given the seed word(s), the relevancies of all other words in the document are calculated (i.e. how often they co-occur with the seed item as opposed to how often they appear without it). For example if the initial seed word of a concept is “dog” the relevancies of the words “fleas” and “bites” might be high because they appear relatively frequently in blocks of text containing the concept (i.e. containing the word “dog”):

Words are added to the concept definition if their relevancies fall above a certain threshold. For example if the concept seed is “dog”, the words “fleas”, “bites” and “barks” may be added to the definition due to their frequency of co-occurrence, leading to the new concept definition (“dog”, “fleas”, “bites”, “barks”).

The process then continues, calculating the relevancy of other words in the document compared to the new concept definition (i.e., how often does a word co-occur with any element in the concept definition as opposed to without any of them). These relevancy values are normalised, with the words above a set threshold being added to the definition. For example, given the concept definition (“dog”, “fleas”, “bites”, “barks”), the items “hound” and “puppy” might now be considered highly relevant due to their frequent co-occurrence with “fleas”, “bites” and “barks.”
Due to their high relevancies, the words “hound” and “puppy” will be included in the new concept definition, leading to the concept (“dog”, “hound”, “puppy”, “fleas”, “bites”, “barks”).

As the relevancies of the words contained within the concept are normalised and there is an inclusion threshold, over time certain of the initial keywords may be lost. For example, the words “fleas” and “bites” although initially quite relevant, may not be as relevant as the new keywords “hound” and “puppy” (because they appear in other contexts). Thus, these words may be removed from the concept, leading to the concept (“dog”, “hound”, “puppy”, “barks”).

Apart from adding highly relevant words to a concept, Leximancer may also add words that are negatively correlated with the concept (i.e. words that rarely appear in sentence blocks containing the concept and frequently appear elsewhere). This was shown to improve the stability of the convergence dramatically, with very common words which were not relevant contributing against classification.

As stated earlier, the process of learning is iterative, but will converge to a stable state. The learning halts when the number of sentence blocks classified by each concept remains stable.
Section 5: Text Preprocessing and Automatic Concept Identification

Text preprocessing is the first phase of processing that is run from Leximancer’s main menu. This phase converts the raw documents into a useful format for processing. This “preprocessing” involves the following steps:

• **Splitting the information into sentences, paragraphs and documents.** These boundaries are important as they generally mark transitions in meaning. The conceptual map of the documents extracted by Leximancer reflects the co-occurrence of distinct concepts. To prevent concepts from being perceived to be related across changes in context (such as across different documents), the co-occurrence is only measured within (and NOT across) blocks typically containing 3 sentences.

• **Removal of non-lexical and weak semantic information.** Within each sentence, the punctuation is removed along with a collection of frequently occurring words (called the stop-list) that hold weak semantic information (such as the words ‘and’ and ‘of’). Furthermore, for documents extracted from Internet email and news groups, the headers are cleaned up and the non-text attachments are removed.

• **Identifying proper names, including multi-word names.** Often in documents the proper names (such as people, places or company names) depict important entities that should be mapped. For this reason, proper names are extracted as potential concepts. In Leximancer, words are classified as proper names if they start with a capital letter. As every word that starts a sentence falls into this definition, only start-of-sentence words that are not in the predefined stop-list are considered as names.

• **Optional language test of each sentence.** To remove non-textual material from the text, such as menus and forms in web pages, sentences that are unlikely to be part of the specified language are removed. This is achieved heuristically by removing sentences that contain less than 1 (or 2) of the stop-list words. If processing spoken language, this setting should be turned off.

Automatic Concept Identification is the next phase of processing, in which “seed words” are identified as the potential starting points of concepts. As explained earlier, concept seeds represent the starting point for the definition of concepts. They are single words (such as “violence”) that are the potential central keywords of distinct concepts. In this optional phase, seed items are automatically identified.
Configuring Preprocessing (Simple Menu Options)

Prior to running the Text Preprocessing phase, the settings should be configured to suit the problem at hand. In order to modify these settings, you can double-click on the “Preprocess Text” icon from the main interface. The following dialog will appear:

The options are divided into 2 sections, described below:

1. Stopword Removal

**Remove Stop Words (yes|no):** Remove words in the predefined Stop List (such as 'and') from the data. Leximancer currently supports English, Dutch, Greek, French, Italian, Spanish, Portuguese, Malay, and German, with the collection of stop words being found in the file “lib\stoplist.pl” (in the folder in which Leximancer was installed). If you are using an unsupported language, you can update this list (e.g. by translating the contained words into your language). Note that stop words are removed, and cannot be selected as manual seed words.

*Commentary:* Stop words are frequent words in a language that are rather arbitrarily designated as having little semantic meaning. Leaving stop words in the data has an obvious effect on the automatic seed selection. If you leave the stop words in, some will be chosen as automatic concepts. This can be annoying, depending on what you are looking for. The presence of stop words also impacts on the machine learning of thesaurus concepts, since almost everything can be correlated with words such as ‘is’ or ‘and’.

**Edit Stopword List (button):** Once you have run the preprocessing stage, you can check the words that were counted as stop-words by clicking on this button. You can browse through this list and if you find any words that you would rather have left in, you can add them to the “go-list.” If you do change the Go List, you must run the Preparation phase again for the newly selected words to be considered by the following mapping phases.

*Commentary:* It can be important to know what words were removed from the text data as stop words, and you may over-ride this removal with the Go List. Note that this list is specific to the current map. You may also be interested in mapping the usage of personal pronouns, for example, so you would need to move these onto the Go List.
2. Language Options

Make Folder tags (do nothing | make folder tags | make folder and filename tags): This parameter is very important if you are comparing different documents based on their conceptual content. This parameter, if set, causes each part of the folder path to a file, and optionally the filename itself to be inserted as a tag on each sentence in the file. For example, a file called “patient1” inside the folder 'control/' below the data folder would have separate tags [[control]] and [[patient1]] inserted in each sentence (if folder and filename tags are set). These tags will be included as concepts in your map. Thus, inspecting the links formed with the other concepts can allow you to compare the content of the various folders.

Commentary: This is probably the most useful simple idea in Leximancer, and generates the most powerful analyses. This feature lets you code all the sentences in each data document with categorical tags just by placing the files in folders, possibly within other folders etc. These tags then become concepts on the map, along with all the other concepts. It is very useful for performing document clustering in a semantic context, or for making a discriminant analysis between two categories. For example, if you had a set of political speeches from a debate on some issue, you could give each speech file the name of the politician, and place all the speech files from each political party in a folder named as the name of the party. Then, if you had several sets of these speeches from different years, you could place each year’s set of folders in its own folder named with the relevant year. The ‘make folder and filename tags’ setting will then insert the name of each politician as a tag on each sentence, and the name of the containing political party as a separate tag on each sentence, and also the name of each year. When you map this data, you will find a concept for each politician, each party, and each year in the Tag Classes collection. You can now choose which of these dimensions to cluster together on the map, so you could view the issues by year, by party, by politician, by year and party, by politician and party, by year and politician, or all of them co-varying at once if you are adventurous.
Configuring Automatic Concept Identification (Simple Menu Options)

Automatic concept identification allows you to choose the number of concepts that you would like Leximancer to automatically extract to show on the map. In order to modify these settings, you can double-click on the “Automatic Concept Identification” icon from the main interface. The following dialog will appear:

**Automatically Identify Concepts (yes|no):** Automatic identification of concepts can be turned off if you would like only concepts that you define yourself to be shown on the map.

**Total concept number (number):** The number of automatically selected concepts that you want to be included in your map. More diverse content requires more concepts, but less than 100 is recommended.

**Commentary:** The larger the data set, or the more conceptually diverse, the more concepts you should look for. As a rough guide, think of the number of concepts increasing logarithmically with the number of documents. However, some data, such as magazine article collections, can be very diverse. Note that the selected set of concepts starts at the top of a ranked list, so you should always get the most important and general concepts. You need to decide how deep you want to go into the more specific concepts. Be aware that if you force more concepts than are really found in the data, you can start getting junk concepts from among the noise.
**Number of names (number):** Names are identified as being words that start with a capital letter. Of the number of concepts chosen, what is the minimum number of concepts that should be forced to be names (a value of 0 allows a natural mixture)

*Commentary:* The default for this is now 0, which creates a natural mixture of words and names by not forcing any names into the list. If you are not interested in names and some are being selected, I normally set this to 1, which selects only one name, then I delete it from the automatic seeds list. If you are particularly interested in names, then increase this number. You can go up to the Total Concept Number, but be aware that if there are insufficient useful names, you can start getting junk automatic concepts from one-off fragments of text.

**Tutorial Questions**

The following sections of this manual will take you through a simple content analysis example. The data contains speeches made by various politicians in the human cloning and stem-cell research debate conducted in the Australian Federal Parliament from 20 August 2002 to 29 August 2002.

- Start Leximancer, and create a project called “stem cell debate”. If you have a version of Leximancer already open, you can open the project manager window by selecting the option “Create or Load project” from the File pulldown menu.
- Double click on the File Selection icon, and choose the folder `C:\Leximancer 2.xx\Tutorial\stem_cell_data` to process. Click OK to return to the main interface.
- Double click on the “Preprocess Text” icon to open up the settings dialog.
- Use the following settings for the Text Preparation Phase
  - In the “Folder tags” option, select “make folder tags”. This will create a new concept for each subfolder contained within your data folder. In the Stem Cell data, there are FOUR such subfolders that contain the speeches made by members of the Liberal Party, the Labour Party, the Nationals and the Independents.
- Close down the Text Preprocessing dialog by clicking on OK.
- Double click on the “Automatic Concept Identification” icon to open up the settings dialog.
- Choose the total concept number as 50, and leave the number of names at automatic. This will create 50 concepts, with the degree of names extracted reflecting the proportion present in the text.
- Close down the Automatic Concept Identification dialog by clicking on ok.

The Automatic Concept Identification node will currently be orange, indicating that it is selected. Clicking on Start will only run processing up until the end of the selected node, allowing you to configure the setting as you go.

- Click on “Start” to run the preprocessing stage and automatic concept identification.
While processing, you can receive more detailed information as to the progress by opening up the Log.

- Open the Log by selecting “View Log” from the Log pulldown menu. The log will be updated at the end of each phase of processing.

For this phase there are two important sources of information contained within the log:

- **Processed Files.** Towards the top of the log window, there is a list of the files that were processed by Leximancer.
- **Ranked list of items:** Towards the bottom of the log window, there is a list of automatically extracted seed words, ranked by their strength. Items enclosed in brackets are identified as “proper names” due to the amount of instances they are capitalized.

- Wait for the processing to finish (i.e. for both nodes to turn green)
- Write down the names of the TWO files in the processed files list contained within the “indep” folder.

| ____________________________ | ____________________________ |

- Write down the FOUR most important concepts found by Leximancer (the Concepts at the top of the list with the highest strength values)

| ____________________________ |
| ____________________________ |
| ____________________________ |
| ____________________________ |

In the text preparation phase, stop-words (high frequency words like “very” and “of”) will have been removed from the text. However, sometimes some important concepts may be removed from the analysis. You can view what words have been removed and add any important words back into the analysis by adding them to the go-list.

- Close down the Log
- Double Click on “Preprocess Text” once again
- Click on the Edit Stopword List button. The following dialog will appear:
Browse through the “Stopwords removed during preprocessing” list
Add the words “concerning” and “chose” to the “Items not to be removed in future” list
Click on the OK buttons to return to the main interface

Now, single click on the “Automatic Concept Identification” icon to select it, and click “start” to rerun both phases.
Look at the ranked list of items in the log
Did the two high-frequency words that you added to the go list appear as concepts?

Why?
Section 6: Concept Editing and Thesaurus Learning

Starting with the concept seeds automatically extracted by leximancer, the “Concept Editing” phase allows users to add concepts to, or remove them from this list. The following phase (Thesaurus learning) then generates the thesaurus of terms associated with each concept. As stated earlier, concepts are collections of correlated words that encompass a central theme. For example, the concept “client” may contain the terms “customer”, “customers”, “client”, “clients”, “subscriber” and “subscribers.” Once such lists of words have been identified for each concept, the concept map can be generated which displays the relationship between the concepts in the text.

The learning of the thesaurus associated with each concept is an iterative process. Seed words are named as such because they start out as being the central terms of a concept, collecting related keywords over time. Also, through learning, the seed items can be pushed to the periphery if more important terms are discovered.

Concept Editing

Double clicking on the concept editing node opens a dialog that allows you to merge, delete or add concepts. This is important for a number of reasons:

- automatically extracted maps may contain several concepts (such as think and thought) that are quite similar, or other concepts that may not be of interest to you. In the Concept Editor, you can merge similar-looking concepts into a single concept, or delete concepts that you do not wish to see on the map.

- you may wish to create your own concepts (such as violence) that you are interested in exploring, or create categories (such as dog) containing the specific instances found in your text (such as hound and puppy).

In order to modify concepts automatically extracted by Leximancer, Automatic Concept Identification (the node prior to concept editing) needs to have been run. If it has been run successfully in the past (i.e. if you have previously generated a map), the node will be green. Double clicking on the concept editor node once automatic concept identification has been run will bring up the following dialog:
In this dialog there are tabs for editing automatic concepts (concepts identified by Leximancer), and manual concepts (concepts that you wish to define yourself). If you have opted to create folder tags during text preprocessing, then an Auto Tags tab will all be visible. Tags concepts are concepts in which (unless otherwise instructed) no associated terms will be learned by Leximancer. Tag classes are useful if you want a simple keyword search for the terms that you see.

Prior to thesaurus learning, only the central seed term for each concept has been identified. In the window above, left-clicking to select the concept ‘adult’ has revealed the single seed term (identical to the concept name) comprising the initial thesaurus for this concept at this stage.

You can use the options in the above interface to merge or delete selected automatic concepts (note: holding down <ctrl> while clicking allows you to select multiple items).

You can also edit automatically-extracted concepts. For instance, if you wish to add terms to the initial thesaurus definition, select the concept from the list and click on “Edit Selected Concept”. The following dialog will appear:
This interface allows you to add or delete terms related to your concept. For example, if you are interested in finding sections in your text containing violence, create the concept “violence” and add any terms from the Frequent Words or Names list above that you think indicate a violent act. If you want to add a term that does not appear in the frequent words or names lists, you can do so by typing an entry in the text box at the bottom left of the window. You can also identify terms that constitute negative evidence for a concept (evidence that a concept is absent), but this option should be used with care. Leximancer will automatically learn the weightings for these words from the text during the learning phase.

Note: if you wish to create a compound word, such as “undergraduate student”, simply add both terms separately to the current thesaurus. Then select these terms from the “Current Thesaurus” list and click on “Merge into a Compound Word.”

You can also rename automatic concepts, and decide whether to allow Leximancer to rename the concept during subsequent learning.

If you wish to create your own concept(s), return to the main Concept Editor dialogue and click on the User Defined Concepts tab. The “Add New Concept” button is now enabled, and this opens the Concept Creation window. Here you can name the new concept yourself, or choose from lists of frequent words and names. Your new concept will then appear under the Used Defined Concepts tab in the main concept editing dialogue, and can be edited using a similar approach to that described above.

Note: If you edit concepts using the settings described, you will need to save the revised seed list and reload it each time you have rerun the Automatic Concept Identification node, or the usual automatic concepts will be learned and loaded. Delete the auto-concepts, and load your edited seed list under the User-Defined Concepts tab.
Tutorial Questions

► Open the Concept editor by double-clicking on the “Concept Editing” node.
► From the concept list, delete concepts that you do not consider to be relevant to a political debate about stem cell research (note: you can highlight several words by holding down <Ctrl> while left clicking)
► Browse the concept list and merge words like “disease” and “diseases” that you believe to be related to the same concept
► Select a concept that you have merged, and click on “Edit Selected Concept.” This should show you a list of seed words for the concept.

Creating or adapting the seed list for words is important in content analysis applications where you are tagging for concepts that you are interested in and wish to manually define. To practice how this may be done:

► Select a concept you wish to edit and click “Edit Selected Concept”
► From the concept editing page, browse the list of frequent words and names to see if there are any other words that are central the concept that you wish to add.
► Now, add a couple of these to the list of seeds.
► Close the concept editing window, by clicking on “OK”.

38
Thesaurus Learning

Before running the learning phase, you should configure the learning parameters to suit your document set. Double clicking on the Thesaurus Learning node will bring up the following dialogue, and allow configuration of the settings below:

Learn Concept Thesaurus (yes|no). Turning off the thesaurus learning will prevent Leximancer from adding additional items to the concept definitions. This will result in searches for concepts being conducted as a keyword search, rather than a weighted accumulation of evidence. This may be essential for text data sets shorter than a few pages, as thesaurus abstraction is less useful for a very small vocabulary, and can result in too few concepts remaining.

Learning threshold(1-21): This setting allows you to control the generality of each learned concept. This setting is inversely related to the relevancy threshold required for a word to be included in the concept thesaurus. Increasing the level will increase the fuzziness of each concept definition by increasing the number of words that will be included in each concept. After you have run the learning phase, make sure you examine the Thesaurus, by right-clicking on the Thesaurus Learning node. Once you have run the learning phase, you can see in the left hand panel of the Thesaurus how many iterations it took for learning to finish. This number should ideally be between 6 and 11. If the number is more than 10, consider lowering the learning threshold. Conversely, if the number of iterations is less than 6, consider raising this threshold.

Commentary: This setting controls how easy it is for the thesaurus learning to grow a concept to a broader set of words. The easier it is, the more iterations of learning will occur, the more words will be added to the thesaurus, and the higher the risk of the concept growing into a more general concept. This can result in the concept being renamed or being subsumed into another concept which already exists. If you examine the log file after learning you can monitor this behaviour. If the data does not consist of natural language, you should probably disable learning, as described above.
**Sentences per Context Block (1-5):** This option allows you to specify the sentences that appear in each learning block. A learning block is said to contain a concept if there are more instances of the positive concept terms present than the negative. This number of sentences per context block during learning should be three (3) in almost all circumstances.

*Commentary:* This setting is shared by the automatic concept selection system but not by the classification/coding system. The value of this parameter should almost always be three. One notable exception, as mentioned above, was press release data where a value of 2 was used, while not breaking at paragraphs.

**Break at paragraph (ignore paragraph/break at paragraph):** Paragraphs typically mark changes in context that may be relevant. As you generally want concepts to be specific to certain contexts, you should choose to “break” at paragraph boundaries so that the contexts are not combined. Only if the majority of paragraphs in the text are shorter than 3 sentences should you consider ignoring paragraphs during learning. The average number of sentences per paragraph in your document set is given beside this option.

*Commentary:* Note that this setting does not control whether classification, or coding, crosses paragraph boundaries. However, it is the same setting as the one which controls automatic concept selection. Only very rarely should this setting allow text segments to cross paragraph boundaries. One notable exception involved government press releases which had an average number of sentences per paragraph of only slightly more than 1. This was due to a marked lack of discourse development. For that data, automatic seeding and learning performed optimally using 2 sentences per context block while not breaking at paragraph boundaries.

**Learn Tag Classes (yes/no):** In some situations, such as selecting “make folder tags” in the preprocessing phase, Leximancer will automatically create tag classes. Turning on “learn tag classes” will treat Tag classes as normal concepts, learning a thesaurus definition for each. This setting is important if you are conducting concept Profiling (discussed below) where you wish to extract concepts that segregate between different folders or files (such as extracting what topics segregate Liberal from National party speeches).
**Concept Profiling**

These settings allow the learning process to discover new concepts that are associated with user-defined and automatic concepts, as defined in the manual and automatic seeds files. This is extremely useful for profiling concepts or names, for doing discriminant analysis on prior concepts, or for adding a larger layer of more specific concepts which expand upon a top layer of general automatic concepts. This also has the advantage that you can ignore large sections of the text that are not relevant to your particular interests. Note that this feature does not use Tag Classes as starting points. If there is a concept in your Tag Classes file that you want to profile using this feature, switch on “Learn Tag Classes” in the above options.

Once the initial concept definitions have been defined, words that are highly relevant to these concepts can be identified as potential seeds for new concepts. For example, using the initial seed “flowers” will result in a concept definition relating to this word with new concepts being allowed to form based on this concept that could divide this concept into more specific topics, such as “roses”, “daffodils”, “petals”, and “bees”.

This process is quite useful if you are trying to generate concepts that will allow segregation between various document categories. For example, if you are trying to discover differences between good and bad applicants, simply place exemplars of each in two separate folders (one for each type), and set folder tags to be on in the text preprocessing stage. This will then create concept classes for each folder. In the learning stage, only include these folder tags in your automatic concept lists, and use the following settings to extract relevant segregating concepts.

**Number discovered(0-1000):** How many extra concepts should be discovered? More pre-defined concepts normally require more discovered concepts, but less than 100 is recommended. A value of 0 disables this feature.

*Commentary:* I would generally select between 3 and 10 discovered concepts per pre-defined concept, to give a reasonable coverage.

**Themed discovery (Concepts in ALL| Concepts in ANY| Concepts in EACH):** Choose how you want the discovered concepts to be related to the pre-defined concept set: ANY gives the Union (related to concept1 OR concept2 ...), EACH gives the Exclusive Disjunction (XOR: related to concept1 OR concept2 but not both), and ALL gives the INTERSECTION (related to concept1 AND concept2). Choosing the intersection of the concepts will only extract concept seeds that are highly relevant to all or most of the learned concepts. For example, conducting a themed discovery from the concepts “sun”, “surf”, and “sand” may lead to concepts more relevant to beach scenarios than using words relevant to only one of these concepts (i.e. the union of the concepts). XOR is particularly designed for strong discrimination of target classifications (i.e., finding specific concepts that segregate between the predefined concepts).

*Commentary:* The idea of a theme occurs here because if the pre-defined concepts surround some theme, such as say beach life, or environmental issues, you probably want the discovered concepts to follow the theme, so choose the ‘intersection’ option.
Tutorial Questions

▶ From the Main Interface single click on “Thesaurus Learning”
▶ Run the learning phase by clicking on “start” and wait for the processing to finish
▶ Open up the log. About 20 lines from the bottom, there is information as to how many iterations were required to converge. Write down the number of iterations that it took to process the data (i.e., Converged = 1 after ??? iterations)

____________

▶ If this value was less than 6, what would it mean (hint: this was explained in the Learning Threshold Section)?

____________

____________

▶ How could this problem be fixed?

____________

____________

In the log file you will see what concepts were suggested in each of the iterations. In this list you will see a distinction between normal words (enclosed in <> ) and names (enclosed in {}).

▶ Write down TWO word-like concepts found by Leximancer in the last iteration

__________________________    ____________________________

▶ Write down TWO name-like concepts found by Leximancer in the last iteration

__________________________    ____________________________
The Thesaurus

Right clicking on the Thesaurus Learning node once the node has been run reveals a link to View Thesaurus option. This opens the thesaurus document, which specifies the definition of all the concepts that have been used in analysing the documents:

Clicking on one of the “Primary Classes” from the top of the document (left window above), will bring you to Leximancer’s definition of the selected concept (shown in the right windows above). The concept definition shows what words are included in the thesaurus of terms for the concept and their relative weightings. Concepts that are shown in double brackets, such as [[professor_carl_wood]] above, are the words that have been identified as proper nouns. Occasionally Leximancer will also identify possible acronyms, and will display such items in single brackets.

Tutorial Exercises

- Open up the Thesaurus by clicking on the Thesaurus link under the map (close down the two maximized maps from the previous exercise first)
- What are the first 6 thesaurus terms for the concept “ivf”?

______________________  ______________________  ______________________
______________________  ______________________  ______________________
Section 7: Classification and Indexing

Classification

Once the concept definitions have been learned, each block of text is tagged with the names of the concepts that it contains. This process is similar to manual coding used in “content analysis.” However, the benefit of using Leximancer for this task is that it is fast (compared to the time taken for human coders), and is more objective (as opposed to humans, where there is much variability in coding performance).

In Classifying the text document, the following steps are taken:

- The text is broken into context blocks of \( n \) sentences
- For name-like concepts, all the associated terms that are present in the block are noted. The block is said to contain the concept if the word with the highest relevancy to the concept is above a set threshold.
- For word-like concepts, the relevancies of all the associated keywords that are present in the block are summed. The block is said to contain the concept if this sum is greater than a predefined threshold.

Indexing

The next step in processing is to measure the co-occurrence of the identified concepts in the text for the generation of the conceptual map. Furthermore, the positions of the discovered concepts and groups of important co-occurring concepts are noted for use with the associated web browser.

In Leximancer, a distinction is made between entities and properties. Entities are the concepts that are actually shown on the conceptual map, and represent the top-level of classification of the text. Generally all concepts can be used as entities, but there are some cases in which only a subset of the concepts should be used. For example, if generating a conceptual map of things to do when backpacking in Europe, there may be several classes of concepts such as distinct countries, cities, and activities. Including all such information on the one map may become incomprehensible, so it may be best to choose a class or two from which to structure the information.

Properties, in contrast to entities, are concepts that are checked for co-occurrence with the entities, but are not displayed on the cluster map. For example, if choosing to structure the conceptual map of things to do in Europe by location, the properties may be the associated activities for each location (the entities). During browsing the conceptual map, selecting an entity on the map will lead to a display of the related properties in the right-hand panel. It is then possible to browse the document for instances of where the selected entity and property co-occur (e.g., to allow you to search for documents containing a certain activity in a specified region of Europe).
Configuring the Classifying and Indexing phase

The first part of configuring the Classifying and Indexing Phase is to specify which of the defined classes you wish to include as entities and properties. Furthermore, you can define which classes should be considered kill classes and required classes.

Kill and Required Classes

Kill classes are concepts that if found in a classified block of text, cause all other classifications of that block to be suppressed. For example, if trying to suppress the processing of questions given by an interviewer to focus on the responses of the interviewee, tag all comments by the interviewer with his/her name (e.g., just put the name in the form TG_name_TG before the end of each sentence). If your material has dialog markers of the correct form, Leximancer has an advanced setting in the Preprocess Text node called Label Identification which will do this for you. Then you can use this concept as a kill class to suppress the processing of such text. The suppressed info can be accessed by clicking on the kill class concept in the map and browsing “Miscellaneous Locations.”

Required classes by contrast, are classifications that must be found in blocks of text, or else the blocks are ignored. At least one of the required classifications must be found in any context block for it to be indexed into the map.

Choosing Entities and Properties

Double clicking on the “Locate Concept Occurrences” node after thesaurus learning has been conducted will open the following dialog:
The current Entities, Properties, Kill Classes and Required Classes are displayed on separate tabs with identical options. You can simply use the arrow buttons to add or delete concepts to each of the lists.

**Classification Settings**

It is important to note that the choice of the length of the text segment used for each classification n-tuple is critical for relational analysis. Part of this decision also involves whether to let these segments cross paragraph boundaries. Leximancer allows you to select the number of sentences per high resolution classification segment. The longer the text segment, the more concepts tend to be measured to co-occur together. This increases the connectedness of the conceptual graph. If concepts are connected to too many other concepts, it is hard to discriminated significant and strong patterns. For extracting cognitive maps, such as the Leximancer cluster map, this makes the patterns tightly grouped but unstable on repetition. Essentially there is not enough contrast in the measured relationships to get strong patterns. This behaviour is most usually observed with lexical concepts. Names tend to be much more selective with their associates. In a similar way, allowing the segments to cross paragraph or speech boundaries can add noise to the concept graph. This also makes pattern formation harder.
Dealing with Unusual Text Formats:

Choice of the parameters for indexing and classification is not difficult for normal prose where the average paragraph length is greater than two sentences. Simply use the default of 3 sentences per segment and don’t allowing paragraph crossings. Problems arise however with dialogue, plays, verse or other unusual text. Issues include:

- For plays and dialogue, we may want to measure the relationships between characters, and between a character and all the concepts mentioned by the character. However, the speaker is normally marked at the beginning of the speech only, and they usually don't mention directly who they are talking to. So we resort to using longer text segments that not only include the whole of most speeches, but cross paragraph boundaries so that consecutive speakers form relationships. Analysing Shakespeare for example, you may wish to use 6 sentences per segment. Unfortunately, this causes the lexical, or conceptual, concepts from the content of the text to be over-connected to other concepts, and the concept map becomes unstable.

- Verse, such as Blake, and other text that does not obey normal sentence punctuation and organisation is difficult for similar reasons. Sentences are either nonexistent or erratic, so the relational connectedness is noisy. Usually the best that can be done is to take each line as a sentence, or code whole paragraphs or stanzas.

High Resolution Classification

The following parameters refer to the tagging of concepts within the sentence blocks. The conceptual map is generated from the co-occurrence of concepts at this level.

Sentences per Context Block (1-100): Specify how many sentences per tagged text block.

Commentary: This is not the same setting as the block length used during learning. You can be more adventurous with this setting, but if you move away from the default 3 sentences, you need to consider interactions with the classification thresholds (below) and some of the indexing parameters (below). The main danger with making this larger is that everything tends to become related to everything else. This can be alleviated to some degree by slightly raising the classification threshold for word-like concepts. If you make the length of the context block smaller, classification becomes more difficult with fewer words as evidence, and the measured relationships become restricted and trivial - the discourse is not observed.

Break at paragraph (yes|no): Prevent tagged context blocks from crossing paragraph boundaries?

Commentary: The discussion above on ‘Dealing with Unusual Text Content’ is relevant here. Again, this setting is not the same as a similar parameter used for controlling the thesaurus learning system. You can be more flexible with this setting, but paragraphs should break context blocks for classification in most normal prose data. Documents with very short paragraphs such as press releases, dialogue, and plays may require paragraph crossing.
**Word Classification Threshold (0.1-4.9):** This threshold specifies how much cumulative evidence *per sentence* is needed for a classification to be assigned to a context block. Word-like concepts are classified in this way.

*Commentary:* Note that the actual threshold used is this value multiplied by the number of sentences per context block chosen above, not the number of sentences found in any particular segment of text. This means that a fixed threshold is applied for all context blocks. It may seem that the actual threshold should be calculated from the number of sentences found, but this would mean that less evidence would be required to trigger a classification in some places than others. After all, one sentence is less evidence than three. The units of this threshold are the same as the relevancy standard deviation values shown for terms in the thesaurus, so you can get a feeling for how much cumulative evidence you need by looking at the learned thesaurus which can be viewed after the completion of the Learning phase.

**Name Classification Threshold (2.6-5):** This threshold specifies the minimum strength of the *maximally weighted* piece of evidence to trigger classification. This is the normal classification of name-like concepts.

*Commentary:* The idea behind this threshold is that one strong piece of evidence is enough to indicate the presence of a named thing such as company. For example, the name of the CEO or the stock exchange code would be enough to indicate the company is involved. However, lots of weak evidence for the company is not sufficient, as it could be describing the same area of the market but actually a competitor company. This is related to the notion that a named thing is an instance of a class, rather than a class. If you want cumulative tagging of named things based on a similar lexical context, there is a setting below for that. Again, the units of this threshold are relevancy standard deviation units as shown in the thesaurus.

**Low-resolution Classification and Indexing**

The following parameters refer to the indexing of the presence of concepts in the text. That is, the instances of the concepts that will be accessible through Leximancer’s search engine. This indexing can occur at a lower-resolution (i.e. a coarser scale) so that the concept needs to be contained in more consecutive sentences before it is identified.

**Blocks per Bucket (1-100):** During indexing, the concept of a bucket is introduced. A bucket contains one or more consecutive context blocks. If the sum of the evidence of a particular concept within the bucket is below a threshold, the specific concept tag is removed from all the sentences in the bucket. For larger document collections, you will probably need to increase this number to conserve resources.

*Commentary:* This feature is designed to overcome the problem (mentioned above) of over-connectivity when trying to code longer blocks of text. What we want to do is to inspect a longer block of text, say 18 sentences, and tag the
concepts which are important to the total discourse while ignoring passing references to other concepts. However, we still want to retain the tight proximity of relationship, say within 3 sentences. The bucket is a way to do this. Let a bucket consist of 6 consecutive 3-sentence coded context blocks. Add up the total weight of each concept represented in the bucket, and if the total weight of any concept does not reach a threshold, erase it from every context block in the bucket. Now disband the bucket, and index the remaining concepts from each context block with only the other concepts remaining in the same 3-sentence context block. The result is that we have filtered out a background noise of passing references, while still maintaining a close co-occurrence measure of relationship. You will need to use this feature for data sets over roughly 20 Mb to save your memory and disk space during indexing. Of course, this will vary with your available computer resources, and will vary strongly with the number of concepts you are using. Indexing resource usage increases rapidly with increasing numbers of concepts.

Tutorial Questions

► Double click on the Locate Concept Occurrences icon to open up the settings dialog.

In this analysis of the documents we will be concerned with only the opinions expressed by the Labour and Liberal parties (i.e. only a subset of the material contained within the documents). In order for only this data to be analysed:

► In the “Required Classes” list add the Labour and Liberal tag classes (i.e. TG_LABOUR_TG and TG_LIBERAL_TG)

For most cases, the default settings for the other options will suffice. Note however, that if you are using Leximancer for content analysis, you may wish to try various settings to assess the sensitivity of the analysis (that is, conduct the analysis several times with various settings).

► For now, leave the settings at their default values and click OK
► Click the start button to run the “Locate Concept Occurrences” phase

For content analysis applications, you can import the data generated by Leximancer into a statistical analysis package such as SPSS.

► Using Windows Explorer, open and inspect the file “highresdata.txt” that is located in your map folder (My_Documents/LeximancerProjects/stemCell/map).

This file contains the data that is relevant to content analysis applications of Leximancer. This file contains a new line for each text segment processed, containing an evidence score for each concept. You will see that most concepts will have a value of 0, indicating that the sentence block was not tagged as containing the concept (i.e., the threshold was not met). Using this file, you could easily use discriminant analysis (e.g., using SPSS) to evaluate whether or not there are significant differences in the speeches given by the members of the Liberal and Labour parties.
Section 8: Mapping and Statistics

The last phase of processing that can be activated through the main interface is “mapping” in which the conceptual map that displays the relationship between variables is constructed. Within this stage, you can choose to generate a conceptual map, or general statistics:

Concept Mapping

In concept mapping, there is one setting to choose, namely whether to use a linear or gaussian map.

The Gaussian map has a more circular symmetry and emphasises the similarity between the conceptual context in which the words appear. The linear map, by comparison, is more spread out, emphasising the co-occurrence between items. The differences between these two settings will be described in more detail below.

One of the principal aims of Leximancer is to quantify the relationships between concepts (i.e. the co-occurrence of concepts), and to represent this information in a useful manner (the map) that can be used for exploring the content of the documents. The conceptual map can be thought of as a birds eye view of the data, displaying the main features (i.e. concepts) and how they interrelate.

The mapping phase generates a two dimensional projection of the original high dimensional co-occurrence matrix between the Entity concepts. This is a difficult problem, and one which does not necessarily have a unique solution. The resulting cluster map is similar to Multi-Dimensional Scaling, or MDS, but actually uses the relative co-occurrence frequencies as relationship strengths which leads to asymmetric relationships between entities. An asymmetric matrix of relationships cannot be dealt with by standard MDS. It must be emphasised that the process of generating this map is stochastic. What this means practically is that for a strict interpretation of the cluster map, the clustering should be run several times from scratch and the map inspected on each occasion. If the relative positioning of the concepts is similar between runs, then the cluster map is likely to be representative. Note that rotations and reflections are permitted variations. If the map changes in gross structure, then revision of some of the parameters is required. The Java viewing applet supplied with Leximancer allows easy animated re-clustering and map comparison, including rotation and drawing around significant areas.

Note: The most common reason for cluster instability is that the entities are related to too many other entities. In other words, the map is too highly connected, and no
strong pattern can be found. The Linear variant of the clustering algorithm is much more stable for highly connected maps, so switching to this setting will often stabilise the map. However, the most important settings which govern the connectedness of the map are the classification thresholds and the size of the coded context block, which are located in the High-Resolution Classification configuration area. If the coded context block is too large, or the high-resolution classification threshold is too low, then each entity will tend to be related to every other entity. The low-resolution bucket size offers a way of indexing longer text segments without increasing connectedness. If you see some highly connected concepts which are effectively bleached of meaning in your data, removing them from the map will usually stabilise the map. These words, such as ‘sort’, ‘think’, and ‘kind’, can often appear in spoken transcripts and may be used as filler words which are essentially stop words. Remember to inspect the actual text locations to check the way these are being used before removing them.

In summary, the Linear clustering algorithm is more stable than the Gaussian, but will discover fewer indirect relationships. The cluster map should be considered as indicative and should be used for generating hypotheses for confirmation in the text data. It is not a quantitative statement of fact.

**Tutorial Exercises**

- Configure the Mapping phase, selecting “Gaussian” as the map type (Note that in fact Linear is the recommended setting here – we use Gaussian on this map for instructional purposes only)
- Run the Mapping Phase by clicking on START. Once finished, a map will appear
- When you have inspected the Gaussian map, close the map window. Now set the map type to linear and rerun the mapping phase.

Once all stages of processing have been completed a map will appear. If you have loaded a previously run project, you can view the generated map by clicking on START, or right-clicking on Map and selecting “View Map.”
The Concept Map

When the Mapping Phase is run, a window appears displaying the conceptual map on the left, and on the right, a list of instructions on how to interact with the map.

What the Concept Map means:

- The brightness of a concept is related to its frequency (i.e. the brighter the concept, the more often it appears in the text).
- The brightness of links relate to how often the two connected concepts co-occur closely within the text.
- Nearness in the map indicates that two concepts appear in similar conceptual contexts (i.e. they co-occur with similar other concepts).
- Coloured circles with names surround local clusters of concepts. These are called themes, and form around highly connected concepts which are parents of thematic clusters. While not essential, the colour of the theme gives an indication of the connectedness of its parent concept.
Tutorial Exercises

The directions of how to use the map are provided on the initial page, but can be viewed any time by clicking on “View Map Instructions” below the map.

To familiarise yourself with the map, try the following:

- Left-Click on any concept to reveal its links
- Left-Click on any vacant position to hide all the visible links
- Drag on the map to scroll it in any direction
- Right-Click on the map to centre the map again
- <Shift>-Click to zoom in on the map
- <Ctrl>-Click to zoom out again
- Use the upper slider to change the number of concepts visible on the map.
- Use the Theme Size slider to change the general size of the thematic clusters. You can choose either a few very general themes or many very specific themes.

Checking Map Stability

As described earlier, the generation of the map is a stochastic process, potentially leading to different final positions of the concepts each time the map is generated. However, there may be some important stable configurations that exist over several generations of the map.

- Click on the Max button TWICE to open up two new map windows
- Move and resize these windows so that they are side by side. Your screen should then look like the following:
On each map, click on Reset to randomise the position of the concepts
Now, click on Learn on each map and wait until they have finished
See if you can identify clusters of similar looking concepts, and see if the same clusters exist on each map (make sure all concepts are visible using the slider)

This map is rather unstable, partly because the concept *research* is so dominant it is almost redundant. The map should certainly use linear rather than Gaussian mode in the mapping phase to maximise its stability.

You can draw on these maps by dragging with the right mouse button. The drawing can be cleared again by right clicking.

- Highlight similar looking groups on each map
- Identify whether or not the position of the Liberal and Labour parties relative to these other clusters are consistent across your maps.

Once you have inspected the maximised map, you may want to print it or export it into another application for use in a report or presentation.

- Click on Settings on one of the maximised maps to bring up a list of the concepts and links currently visible in the map window. If you do not see Background colour and Font size options below this, increase the size of the window slightly to allow this. Change the map background colour from black to white, and increase the font size of labels on the map from 12 to 14. Your screen should then look like the following:
Use the maximise button at the top, right of the map window to display the popup map full-screen.

Click on Copy at the bottom, right of the map window to copy the map image, then open a document, presentation or spreadsheet application that you might use to create a report. Use the paste function in the target application to paste the map image into the new document.
Browsing Information Using the Concept Map

The Document Summary

Clicking on the “View Document Summary” button below the concept map reveals a set of extracts that characterise the text. The extracts are blocks of sentences that contain concepts that have been found to be central to the text. The document summary allows users to gain a feel for the relationships between central concepts in the text, and may be useful for users who want peruse important sections of a large text or data set.

Relationships between concepts

As stated earlier, the conceptual map can be used to browse for instances of specific concepts and relationships. For example, when TG_LABOUR_TG is clicked on the map, the co-occurring concepts are displayed in a separate frame:

Clicking on the “Navigate Locations” button beside one of the related concepts allows you to browse through the instances of where this concept co-occurs with the main selected concept. For example, navigating the concept “embryos” will display instances in the text where members of the Labour party talk about embryos:
Although, by default, clicking on a concept will show the related entities of the concept and their associated strengths, other options also appear in the main map window (at the bottom of the text window):

- **Related Entities**: this is the same as the default action when you right click on an entity from the map. Clicking on this option gives a ranked list of entities that co-occur with the main selected entity.

- **Entity Vocab**: Clicking on “Entity Vocabulary” will allow you to list the top 30 most frequent words that appear in sentences containing the entity.

- **Related Properties**: this option will give a list of properties (and possibly combinations of properties) that co-occur with the main entity. Clicking on the icon beside a property will allow you to browse through the locations in which the entity and specified properties co-occur.

- **Property and Vocab**: Clicking on “Property Vocab” will give a similar list for the concept, but details what “properties” co-occur with the chosen entity. Clicking on the icon beside the property will give a list of the 30 most frequent words that appear in the sentences containing both the entity and the property.

- **Misc. Locations**: In some cases, there will be some locations in the text in which the entity does not co-occur with any other entities or properties. This option allows you to browse these locations.
Tutorial Exercises

Use the browse button to explore instances of text relating to the concepts “ethical” and “debate” to view some of the main arguments involved in the debate.

The Log Book

When browsing through instances in the text where concepts occur or co-occur, a “Log book” has been implemented that allows you to save instances of the text that exemplify this relationship. When you are browsing the text as in the window shown above, an “Add to Log” option is visible on the left and below the instance navigator box. Clicking on this will save the highlighted text to the log. You can view the contents of the log by clicking on the Log Book tab:

You can add your own comment to the Log in the right-hand window. You can also use the Copy to Clipboard button to paste it into another document for editing or printing. When you close down the map, the Log Book is saved in your project directory (e.g., my documents/LeximancerProjects/stemcell/map) in the file logbook.html.

Tutorial Exercises

Find an interesting thread of arguments relating to stem cell research, and add instances of them to your log book.
Generating Concept Statistics

Apart from using the map, you can use the statistics option for visualising the co-occurrence between concepts.

- Close down the map window.
- Double Click on the Map node. The following dialog will appear:

![Map Settings Dialog](image)

- Select “Concept Statistics” to display the output as a set of statistics rather than a map. This will bring up the following dialog:

![Map Settings Dialog](image)

This dialog allows you to visualise the relationship between particular concepts.
- Add the concepts TG_LABOUR_TG and TG_LIBERAL_TG as category variables.
- Add interesting topics such as “bill”, “people” and “research” as attribute variables.
- Click on the “OK” and then START. This will open up a bar chart showing the co-occurrence between the selected concepts:
For example, in the above graph, we can see that the Liberal party talks less about people than does the Labour party, relative to the total amount of text from each party.

By inspecting your stats output, determine other interesting trends.

In your own data, the above approach can also be used for highlighting temporal trends within your document set, or differences between various conditions. For example, you can place your data into separate folders under a common parent folder for representing distinct sections in time. By turning on “make folder tags” in pre-processing, a new concept will be created for each of these periods, and the co-occurrence with important attribute variables can be calculated.
Section 9: Text Preparation (Advanced Settings)

The following sections outline advanced settings for each of the various phases not previously mentioned.

The options in this section relate to the preprocessing of the text, which allow you to specify how the information is formatted (e.g., what constitutes a paragraph, and what words should be ignored during processing). The options are as follows:

**Using Tabular Data**

Leximancer can effectively analyse data in tabular formats. This data should have been exported as delimited text from a spreadsheet or database, and the first line must contain the column headers. This is particularly useful when some of the data fields contain natural language text. In order to import a table, simply double click on the File Selection node, select “Single Table” as the file format and choose the file you wish to process. Instead of “OK”, the button at the bottom of the Data Selector dialog will change to “NEXT=>” which will allow you to select which columns of the table you wish to process as text, or to count as variables. The variable fields should be discrete rather than continuous, as each different value is treated as a separate concept in leximancer (i.e. they will appear as separate concepts on the map).

**Processing Languages other than English**

From the main interface, selecting Set Locale from the Language pull-down menu will allow you to set the locale name. Use this configuration if the data contains non-English text. Names and availabilities of locales depend on your operating system. For Windows 2000, supported locales are of the form 'french' or 'french-canadian'. The help button next to the box for entering the locale lets you view a list of valid locales for Windows, but note that you may not have some of these installed, and Leximancer doesn’t yet work with all the listed languages. For various un*x systems, several methods for finding locales are described at http://www.perldoc.com/perl5.6/pod/Finding-locales, but they are generally of the form 'fr_FR'.

Commentary: This setting is mainly required for selecting the character sets that are considered as alphabetic and numeric, and also for deciding which are upper case and which are lower case. Perl 5.6, and hence Leximancer, only fully supports 8-bit character sets at present. Unicode will come shortly.
**Text Preprocessing Settings**

From the main menu, you can view extra settings by unselecting “Simple” from the Menu Complexity pulldown menu. Double clicking on the Preprocess Text icon will then reveal the following dialog:

The extra options not previously described are detailed below:

**Stopword Pattern (Regular Expression):** This is an optional setting for ignoring types of words during document preparation. Enter a regular expression to match words that will be ignored. For example, words containing numbers would match the pattern ‘\d’. See Appendix A for a description of the regular expression format.

*Commentary:* You might be having problems with a bunch of numerical codes which are filling up your automatic seeds or your vocabulary lists. The stop pattern provides a way to remove certain patterns of words using the full power of Perl regular expressions. Note that this pattern is not anchored at a start or end of word. To do that, you need to use the `^` and `$` patterns. For example, to remove either the whole word ‘dog’ or the whole word ‘cat’ from all the data, use the following pattern: `^(dog|cat)$`

**Identify Names (yes|no):** Do you want words that seem to be names (i.e., non-stop words, starting with a capital letter) to be stored as potential concepts?

*Commentary:* This setting requires text data which uses upper and lower case, where upper case designates proper names. This doesn’t work in many languages or in some text data where case is missing, but it is very useful much of the time for tagging proper names. Note that it binds compound names such as University of Queensland into one token. Also note that it is by no means perfect at dealing with capitalized words at the start of sentences.

**Language testing (off, weak, normal, strong):** The Language Test feature examines raw text sentences to decide whether they are valid prose from the configured languages. This is achieved by counting the number of stop-words that appear within
each sentence. If this number is high, it is likely to be a sentence from a configured language. This option allows you to specify the number of stop-words that are required for the sentence to be further processed.

Commentary: This feature is good for reports or other prose documents where you don’t want to process tables of numbers or lists of words. It is almost essential for web pages or e-mail messages which usually contain menus or signatures. This sort of repeated data can potentially contaminate your automatic seeds and machine learning. If the data is not prose from a supported language, or if it is composed of transcribed speech or other colloquial matter which does not obey the rules of prose style, then this feature should be weakened or disabled. You should also disable this if you need to analyse absolutely every bit of text in the data.

Label Identification (none|dialogue|headers): This function is designed to utilise speaker labels or headers in the text data. The dialogue setting identifies labels that start with upper-case, end with a colon and space, and are located at the start of a line. Each dialogue label is appended to the end of every subsequent sentence until a new label is found. This can be useful for plays or interview transcripts. The headers setting looks for labels that start with upper-case, end with a colon and space, are located at the start of a line, and are followed by a name. Each label forms a separate tag, in combination with its following name. All label-name pairs are appended to every subsequent sentence. A label's associated name value changes whenever a new instance of that label is found. This can be useful for text e-mail messages or for data obtained from a document retrieval system.

Memory usage: The automatic concept selection process can consume a great deal of memory, depending on the number of concepts you have requested. This setting specifies the memory usage policy. The more memory utilized, the faster it will run, but this can overload your system.

Commentary: On a Windows system, keep the memory usage low, at least for the first go at a map. You can use higher settings on Linux with more confidence.

File Preparation (All Files|New Files Only): Delete any previously prepared text?

Commentary: The normal situation involves mapping a fixed set of documents, and in this case you want any previously prepared text cleared out from the map. In case you want to add some new files from a data folder to an existing map, then use this setting to stop the previously prepared text from being deleted. This will also prevent the importation and preparation of any files in the data folder which have been prepared before.

HTML (normal|simplified): Leximancer creates a copy of the original document files, but with the addition of sentence markers for indexing purposes. These documents (which are displayed when browsing the information through Leximancer) can be as similar as possible to the source documents if you wish. By default, this setting simplifies the surrogate versions (to simplified HTML text elements) to guard against problems which may occur when complex HTML source data is analysed.
For example, if the original text is coloured white on a black background, it may not be displayed appropriately, so choosing “simplified” will remove this colour formatting.

Commentary: The default setting for this these days is to keep it simple, because the uncontrollable formatting and scripting in some HTML data caused the text highlighting to fail.

Sentence Boundaries (Automatic|New Line): Leximancer comes with a sentence boundary detector, based on Western European languages. However, if you want each text line to be considered as a new sentence, use this setting. This can be useful for semi-structured text, or log files.

Commentary: The default sentence boundary detector is a regular expression designed for Western European text. It can in fact be edited if you are comfortable using a text editor to modify the configuration file in a map folder, called LexConfig.pm. Be very careful not to use Word or Wordpad, or any other word processor that will quietly add its own formatting to the file, as this will break the map. Be warned that it is not a simple regular expression. If your text does not suit the default sentence splitter, but you can arrange to have one sentence per line, then turn on this setting. The main uses for this are log files, catalogues, spreadsheet or database tables, or verse.

Start of Document (regular expression): This field is only used for text files that contain multiple documents. Enter a regular expression (if you are familiar with this terminology and format) that will match the starting line of each document. For example, text e-mail files would normally require the pattern `^From\s`. See Appendix A for a description of the regular expression format.

Commentary: The principle reason for having this setting is to allow Leximancer to break up long text documents so that browsing is easier. It can be slow or infeasible for you browser to load an entire 100 Mb file of e-mail messages at once. You can also use this to break up the data into separate units for classification on each separate line of the high-resolution classification output. Note that learning and classification text segments never cross document boundaries.

Start of Paragraph(select from list, or enter regular expression): This field controls the way paragraphs are found in plain text documents. You can choose from the options “line with no letters OR indent of one or more spaces”, “line with no letters OR a label followed by a semicolon and a space”, “line with no letters only”, or you can enter your own regular expression.

Commentary: Paragraphs can be important because people are likely to change topic at a paragraph boundary, or perhaps the speaker changes at a paragraph. Some care should be taken to select or create a pattern suitable for the data. This means taking a glance at some of your data files. Perl regular expressions will let you create very powerful patterns.
Section 10: Concept Learning (Advanced Settings)

Automatic Concept Identification

From the full Automatic Concept Identification menu, the following options are available:

The options not previously described are detailed below:

Concept Specificity (number): Concrete concepts differ from normal concepts in that they are much more specific. That is, they are defined as being words that are strongly related to a small number of concepts (as opposed to normal concepts that are strongly linked with a large number of other concepts). Concrete concepts are suitable for creating book indexes, for example. However, because of their specific nature, you will need many more concrete concepts to cover the content of a document (which can be very demanding on resources).

Commentary: Use this with caution. You need lots of memory and lots of concepts to use this effectively.

Sentences per Context Block (1-5): Specify how many sentences per context block. This setting is shared by the Thesaurus Learning phase, and can also be set there. This should be three (3) in almost all circumstances.
Commentary: This setting is shared by the thesaurus learning system, but not by the classification/coding system. The value of this parameter should almost always be three. One notable exception, as mentioned above, was press release data where a value of 2 was used, while not breaking at paragraphs.

**Break at paragraph (yes|no):** This setting is to prevent context blocks from crossing paragraph boundaries. Only if the majority of paragraphs in the text are shorter than 3 sentences should you consider ignoring paragraphs. This setting is shared by the Thesaurus Learning phase, and can also be set there.

Commentary: Note that this setting does not control whether classification, or coding, crosses paragraph boundaries. However, it is the same setting as the one which controls learning of the thesaurus. Only very rarely should this setting allow text segments to cross paragraph boundaries. One notable exception involved government press releases which had an average number of sentences per paragraph of only slightly more than 1. This was due to a marked lack of discourse development. For that data, automatic seeding and learning performed optimally using 2 sentences per context block while not breaking at paragraph boundaries.

**Boilerplate cutoff (very weak – very strong):** In selecting concepts, some concepts may be overly specific in the sense that they only appear in a particular textual context. For example, certain words may only appear in repeated addresses, disclaimers or copyright notices, and hence, will be identified as specific concept. Setting the Boilerplate cutoff, filters out such overly specific concepts from the automatic list. However, if this value is set too strong, the remaining concepts may become too general.

Commentary: This feature looks for common words which frequently occur in the same context and stops them being selected as automatic concept seeds. It can be very useful for web pages which contain repeated links or menus. Be aware that it can cull concepts you might want if you set it to a high value. Note that this filter does not remove the words from the text, it just stops them being selected as automatic concepts. You can still use these words as manual seeds.

**Bigram sensitivity (number):** A bi-gram is a collection of two words that can be thought of as a specific concept (such as “undergraduate student”). This setting controls the sensitivity of the bi-gram detection system. The system calculates how often words appear to the left and right of each other word, and are included as a bi-gram if the frequency is above the specified threshold. If this setting is too high, many additional very specific bi-grams will be created.

Commentary: This function looks for words which occur frequently on either side of automatic seed words, and turns the word pair into a concept. Note that the optimal value of this setting changes with the size of the data. For very small data sets, you may need to turn this off entirely. For larger data sets, you can be much more aggressive in detecting bi-grams.
**Thesaurus Learning**

From the full Thesaurus Learning menu, the following options are available:

![Concept Learning Settings](image)

The options not previously described are detailed below:

**Learn Concept Thesaurus (yes/no):** This setting allows you to disable all machine learning based on the concept seeds, and generates a simple keyword classifier based on your seed terms. Tag classes behave this way anyway, independent of this setting.

*Commentary:* It is occasionally necessary to disable thesaurus learning. This is mainly for structured data such as log files, catalogues, or data tables which do not use natural language, or for short data sets (less than three pages). In these cases, the automatic concept selection and the later mapping phases can produce very useful maps, but the thesaurus augmentation is not necessary or desirable. If you try using learning in these cases, you may see far too many of the concepts being subsumed into more general concepts, and it may not converge within 11 iterations no matter how low you make the Learning Threshold.

**Learning Type (Automatic|Supervised):** This setting is for learning from supervised classification, such as from hand tags, folder tags, or headers. This will build classifiers that attempt to faithfully match human classification decisions, rather than discover an extended thesaurus from seed words. Note that the names of the learned classifiers will be modified to the form `sc_..._sc`.

**Clear the lexicon (yes/no):** If you have already trained on a data set and are using the same map folder, this option lets you keep or destroy the old learned concepts. If you keep the old concepts, any new concepts will be automatically added to the list.
Commentary: If you retain the old lexicon, you can add new concepts to it by revising your seeds files. This can allow you to grow your lexicon even though you have limited computer RAM memory.

Follow-on learning (yes|no): If you have previously learned a set of documents that has been recently updated, you can allow the system to update the learned concepts saved from previous sessions. This learning will be comparatively rapid as only a few iterations of learning will be required to modify the concept definitions (as opposed from learning them all from scratch).

Commentary: Be aware that if you have turned on automatic concept selection, this Follow-on Learning is disabled anyway. The reason for this is that if you use Follow-on Learning, any new automatic seeds file would otherwise be ignored. Use this setting if you want the concepts to be the same as those in the current lexicon, but you want to rapidly update the words in the lexicon for a slightly modified data set. This can be an interesting way for growing a very rich thesaurus, but classification performance will degrade if you heavily modify the data set.

Sampling (automatic|1-10): Sampling during learning speeds up the learning process by only reading every $n$th block of text. The automatic setting is normally fine, but you can override this if necessary, by choosing $n$.

Commentary: The automatic sampling setting looks at the size of the total text data set to decide an appropriate value. The actual sampling number is increased by 1 for every 15 Mb of text data, so for anything under 15 Mb, sampling of 1 is used, which means every context block is examined for learning. For 15 to 30 Mb of text data, a sampling of 2 will be used, which means that every second context block is examined for learning. Note that classification never uses sampling, and classifies every context block. Tests have show that classification performance only decreases marginally if the automatic schedule is followed.

Phrase separation (off|0-5): Concept seeds of two or more terms in combination can be considered as similar to phrases. This setting specifies the maximum number of words between two terms in the text for them to match a phrase. If you switch this off, the components can be anywhere within the classified context block.
Section 11: Classification and Indexing
(Advanced Settings)

Classification Settings

From the full Classification Settings dialog, the following settings are available:

The options not previously described are detailed below:

**Classification Threshold for Supervised Classifiers:** This threshold specifies how much cumulative evidence per sentence is needed for a supervised classification to be assigned to a context block. Supervised classifiers have names of the form `sc_..._sc`.

**Treat Names as Words (yes|no):** This setting forces name-like classes to be classified using the same system as word-like classes, allowing more intuitive coding.

**Statistics Type (count|weight):** Normally an assigned tag is simply counted. You can change this setting so that tags are assigned a confidence weight. This results in weighted sums rather than count statistics.
**Document Metadata (no output|as vector|as matrix):** Some advanced applications require classification metadata for each document. If enabled, this is written out in both Perl and XML format (as documents.dat and documents.xml). Matrix classification allows more accurate document comparison by utilising concept co-occurrence within each document.

**Indexing Settings**

From the full Indexing Settings dialog, the following options are available:

The options not previously described are detailed below:

**Property Tuple filter (1-5):** Leximancer saves instances of individual properties or collections of properties that co-occur with the main entities. For example, given that “till” and “evening” are concepts, Leximancer can store information as to where this pair of properties has appeared close to an entity. Changing this parameter changes the number of properties occurring together that will be picked up as a group. For larger document collections, or larger numbers of word-like Property classes, the recording of larger combinations of properties can be expensive on resources. This filter specifies how many Properties from the top of each ranked data tuple are actually indexed.
Commentary: Recording tuples of co-occurring property combinations, even in the context of an entity, can be combinatorically dangerous. This setting limits your exposure by specifying the maximum number of concepts taken from any tuple. It uses the classification weights to rank the concepts before truncating the list.

Tuple Indexing (tuples as found | each single property | each property pair | record vocab only): This setting changes the way locations of Property tuples are indexed. Normal behaviour is to record the location of each tuple once, indexed by that tuple. Alternatively, the location of each tuple can be indexed under each class in the tuple, or each pair of classes. The last two options can be very expensive on resources. The final option is just to record the vocabulary for each separate property, with no tuple locations at all.

Commentary: Don’t worry too much about this setting – if you want a non-default setting here, chances are you will understand it. If you don’t want any properties at all, just empty the list of possible properties higher up the configuration screen.

Remote Links (local | remote): Normally, indexed documents for browsing are linked to the document location on the local host. If the data has been collected from a remote web site and stored in a mirrored folder hierarchy, with the domain name of the remote host as the top folder (e.g., as created by the wget robot), this setting will cause browse document links to point back to the remote document. Note that this disables sentence highlighting and in-page hyperlinking. Links will only point to the start of a document.
Section 12: Leximancer Strategies

This document is intended to be a guide for some of the styles of analysis that are possible with the Leximancer system. However, the system is designed to be a general-purpose tool, so the confident user is encouraged to experiment. Firstly, the toolkit of useful techniques will be described – these are commonly used components of the analysis process. The following section then describes how to use these tools in various applications of Leximancer.

Toolkit

Automatic Concept Selection
This step automatically extracts seeds for concepts spanning the entire document collection. These seeds represent the starting point of concepts, with the full definition being extracted during the learning phase. The automatic extraction of concept seeds is performed during the “automatic concept identification” phase. You can select how many concepts to use, and you can force a specified number of the concepts to be based on proper names.

Folder Tag Generation
Use Folder Tags to easily convert categories of text (or variables) into explicit tags in the text, and hence into concepts on the map. This step causes the names of all parent folders of a file, and optionally the name of the file itself, to be embedded as separate tags on each sentence within a file. For example, a file called “patient1” inside the folder “control1” below the data folder would have separate tags [TG_control1_TG] and [TG_patient1_TG] inserted in each sentence (if folder and filename tags are set). These tags are imported into the Automatic Tag Classes list so long as automatic concept selection is enabled. Note that the hierarchical order of the folders is not important, and is not carried into the Leximancer map. If you have a folder called ‘Control’ under several different branches, then this becomes one concept called ‘TG_Control_TG’ on the map, and this can be a powerful feature for freeing up the exploration of category covariances.

Manual Concept Seeding
You can seed your own concepts prior to running the learning phase. This is accessed through the Concept Editing node. If you create new manual seeds, thesaurus definitions of these concepts will be extracted from the text along with any automatic concepts. You can also create your own Manual Tag Classes and these act like lists of keywords, or fixed dictionaries – they are not modified by the learning process.

Concept Profiling
This function is not the same as automatic concept discovery. The aim here is to discover new concepts during learning which are relevant to the concepts defined in advance, either in the manual concepts area or the automatic concepts area. For example, this setting would allow you to extract the main concepts related to stem cell research from a broader set of documents. Note that Tag Classes do not take part in this process unless you have enabled the “Learn Tag Classes” option in Thesaurus Learning. Concept Profiling settings can be found in the Thesaurus Learning icon. If you have some Tag Classes, such as folder tags for example, which you wish to
profile, simply turn on the Learn Tag Classes option. It is important to understand that although these discovered concepts are seeded from words that are relevant to the prior concepts, they are then learned as fully-fledged independent concepts. As a result, the map will contain some peripheral areas of meaning that do not directly contain the prior concepts. Contrast this with the ‘Required Classes’ function described below.

This function has three alternative behaviours: ALL, ANY, and EACH. You can ask for the related concepts to be relevant to many of the prior concepts, and thus follow a theme encompassed by all the prior concepts – this is the ALL option, and resembles set intersection; alternatively, the discovered concepts need only be related to at least one of the prior concepts – this is the ANY option, which is similar to set union. The EACH option discovers equal fractions of profile concepts for each predefined concept, and these concepts show what is peculiar to each predefined concept. The EACH option is very useful for enhanced discrimination of prior concepts.

The concept seeds created by this profiling function are saved in a seeds file called ‘assoc_seeds.dat’. You can come back and load these seeds into the manual concepts area if you wish to refine the set of associate concepts. If you do, you should then switch off Profiling and run the learning phase again.

**Required Classes**

The Required Classes feature is accessed through the Locate Concept Occurrences node. This does not create new concepts. Rather, you choose concepts, which have already been learned, as strict constraints on the mapped text. At least one of the required classes must be found in a text segment for it to be indexed onto the map. This is a much more restrictive way of focusing the map than the Profiling technique described above. Social Network analysts might consider these as structural variables.

**Properties**

Properties are a second level of classification. You can select any concept as a property using the Possible Properties list in the Locate Concept Occurrences node settings. Properties (unlike entities) do not appear on the cluster map, but rather are viewed to be associated with the entities on the map. When you select a concept on the map, you can choose to view its associated properties. Properties that co-occur with the chosen entity will then be revealed in terms of a ranked list, with the text locations at which they co-occur being browsable. For example, this can allow you to navigate a travel guide of things to do in Europe, using the locations in Europe as entities (thus being visible on the map), with the activities being viewed as associated properties.

Properties have two main uses:

- Firstly, you might want to know the characteristic vocabulary used by different people or organisations about a range of concepts. To do this make sure the concepts are in the Entities list, and all the required names are in the properties list.
- Secondly, you might want to know the pairs of individuals or organisations that are related by certain concepts. Again, place the relational concepts in the Entities list, and the names as properties.
Applications of the Toolkit in Leximancer

The following sections describe some of the main uses of the Toolkit within Leximancer.

**Semantic Characterisation of the Entire Document Set.**

This is the default behaviour of Leximancer. The strategy allows the automatic extraction of a collection of concept seeds that reflect what was important to the authors of the text based on their word usage. It should be mentioned that there can be several reasons for words being important to authors. Intended meaning is the most obvious, but emotional state and conventional usage are also important. Normally, a stop-list of frequent and meaning-poor words is used to remove words such as ‘is’, ‘and’, ‘but’, ‘I’, ‘me’, etc. This is important for a conventional semantic map, but you can over-ride the removal of selected stop-words using the Go-List if you wish to analyse some of them. In fact, stop-word removal can be disabled entirely, but this can have consequences. For example, almost all concepts could be related via the word ‘and’.

The default setting for automatic concept selection does not force any proper names into the list. This allows the most natural ranking of concepts. If you need to enrich the map with actor names, then force in some names.

You can choose the total number of automatic concepts that are generated from the top of a ranked list. Don’t make this number too large for short documents, or you will generate very low frequency junk concepts.

You can edit the automatically generated concepts prior to learning if you wish to remove those which are not of interest or merge some together. Use caution when merging concepts – words that appear to have similar meaning may be used in quite different contexts in certain documents. It is advisable to generate a raw map first and inspect the semantic groupings. Adjacent concepts may then be safely merged during map revision.

**Comparison of Names or Categories based on Shared Semantics**

The strategy here is to compare categories based on shared concept usage. The categories are usually Tag Classes, often generated using the folder tags setting. For example, you might have several submissions on an issue from different people and organizations, and you want to know how they focus on the major topics. The ‘folder tags’ setting will cause each file to generate a tag class. This is also very useful for looking at trends over time. Simply put documents into folders by month or by year etc. Use automatic concept selection to generate a set of concepts that characterise the whole text collection, then complete the map, making sure that the tag classes and the automatic concepts are both in the Entities list. Under the mapping configuration, make sure that you are using a linear map. The map should then show the categories distributed around the concepts. Now place two map applet windows side by side, reset and re-cluster one of them, and then compare with the other. If the locations of the categories relative to the concept field show little repeatability, then you should conclude that the categories are difficult to differentiate based on the global concept selection. Essentially, they all address most of the same global concepts to similar
degrees. This is a result in itself, but if you actually wish to discriminate between the categories, see the next section.

**Discrimination of Concepts, Names, or Categories based on Semantics**

The goal here is to generate concepts that are relevant to each element of a set of prior concepts or categories. We wish to show the topics that distinguish them. For example, we might want to know what attributes discriminate failing students and what attributes discriminate successful students. First, ensure that the prior concepts are defined either in the manual seeds area or the automatic seeds area. If the prior concept is in fact a tag class, turn on the **learn tag classes** option. These concepts can then be used by the learning phase to generate related concepts. **Delete any other concepts that are not required for profiling.** The related concepts are generated by using the ‘Concept Profiling’ tool, as described above. Use the XOR setting, and choose to discover at least one concept per prior concept, and preferably more.

After running the learning phase, you have the option of loading the discovered associated concept seeds for revision and relearning, as described above under the description of the ‘Concept Profiling’ tool. If you do this, then switch off concept profiling.

To create the map, make sure the prior concepts and the discovered concepts are in the Entities list. Finally, make sure that you use the linear mapping setting.

The final map should show the prior concepts in proximity to their discriminating discovered concepts. Be aware that the discovered concepts in this case do not necessarily characterise the whole text. In fact they may be quite specific to the prior concepts and not cover the major themes of the whole data set at all.

If you wish to go on to perform statistical discriminant analysis, the file ‘highresdata.txt’, located in the map folder, can be loaded into a spreadsheet or a statistical package. This file is a tab-delimited data file containing a row for each text segment, which lists the file, the document, the sentence, and the weighting found for each concept. The first row of the file contains the column headers.

**Lexical Comparison of Names and Categories (Supervised Training and Classification)**

The goal here is to detect similarities between names or categories based directly on similar lexical usage instances, rather than on similar concept usage. An example of this could be plagiarism detection. Another example would be marking. The categories under inspection should all be tag classes, such as student numbers and/or grades extracted from file names using the folder tags function, from *headers* within the files, or from categorical variables in tabular data. Remove any automatic concepts or tag classes you are not directly interested in – remember that the presence of other concepts on the map will affect the clustering and contribute to indirect semantic grouping. You need to turn on the setting **learn tag classes** so that a thesaurus entry can be learned for each. Also turn on the **supervised learning** setting under Thesaurus Learning. To perform stylistic comparison, such as for plagiarism detection or author identification, set the ‘Sentences per context block setting’ to 1 to match based on style – we want to classify based on intra-sentence syntactic style in
those cases. Next, run the learning phase. Remember to save any seeds you might want again by using the Export function.

When configuring the classification settings, you will need to use the advanced menu mode. Again, set the number of sentences per context block to 1 for a stylistic analysis. You will find two sets of classes to choose from – one set consists of the original tag classes, the other of a supervised classifier (denoted by sc_..._sc) for each original tag class. The supervised classifiers have been trained to attempt to mimic the original tag classifications, but only using the available lexical evidence. In this way, the supervised classifier for one author can match text from another author if there are stylistic similarities, or supervised classifiers learned from marked text can be assigned to unmarked text. This classification can be examined using the overlap of supervised classifiers with exact tag classes on the map or in the generated data files. To increase the sensitivity of matching, you can decrease the classification threshold for supervised concepts. However, the best way to decrease the matching sensitivity and increase precision is to raise the blocks per bucket classification setting.

**Profiling an Issue or Scenario within a Text Collection**

If we want to profile a more complex scenario which is present within a text collection, but there are other concepts and regions of the text which are not relevant, then this strategy is appropriate. Disable automatic concept selection and define manual concept seeds for multiple simple concepts that encompass the scenario. For example, if you are interested in beach recreation, define manual concepts such as ‘sand’, ‘surf’, ‘sun’, ‘water’, ‘beach’, ‘fun’, and ‘towel’. Don’t make the seed words for each too complicated, and don’t try too hard to restrict the seeds of each concept to just the topic you are after. We will be considering the intersection of all these elements.

Enable the ‘Concept Profiling’ function and select the concepts in **ALL** option. Choose to discover several concepts per prior concept. As described in the Tools section above, you have the option of revising the discovered concept seeds after learning. If you are attempting to discover the network of associated names around a name or a scenario, the advanced version of the learning configuration page lets you choose to only discover name-like concepts.

You should try the Gaussian mapping algorithm first for this style of map.

**Extracting a Social Network with Semantic Context**

The goal here is to extract a network of actors, or names, based on their interactions, and to characterize the conceptual nature of their interactions. An example of this could be finding a network of companies, employees, cities, and counties from a set of industry reports. Firstly, configure automatic concept selection to extract only names, as many names as you think are warranted. Do this by forcing all the automatic concepts to be names. Now tidy-up the automatic concepts, removing any unwanted proper nouns, and adding any missing names that you want to watch.

Next, enable the ‘Concept Profiling’ function, select the concepts in **ANY** setting, and choose to discover one concept per prior name. You can increase the number of discovered concepts if you want a richer map. As mentioned in the Tools section...
above, you have the option of revising the discovered concepts after performing the learning phase.

Make sure the names and concepts are included in the Entities list. For a more interesting analysis, put the names in the Properties list, and, in the advanced version of the classification page, set the ‘Property tuple filter’ to 3 and the ‘Property tuple indexing behaviour’ to the value: ‘each property pair’. These settings will let you select a concept entity in the map as a type of predicate, and then view pairs of properties (i.e. names) that are related via the predicate.

The resulting map will show a network of names intertwined with concepts that describe and mediate the relationships.

For a more traditional and constrained social network which uses structural variables, use automatic concept discovery to select the names, as before. Next, manually seed some concepts as structural variables. After completing the learning phase, put the name-like concepts in the Entities list, and place the structural concepts in the list of Required Classes. The result will be that only text segments that contain one of the required classes will be mapped. Consequently, the map will show a network of names based on relationships that involve at least one of the required concepts.

**Data which is not Natural Language**

Data that is not natural language can be analysed using Leximancer so long as it is not HTML. Examples of this might be logs files or catalogues. There are two main considerations: (1) arrange the data as one record, or ‘sentence’, per line in the data files and set the ‘one sentence per line’ switch in the preprocess text configuration window; (2) activate the ‘disable learning’ setting in the thesaurus learning configuration window. The thesaurus learning system relies on the nature of natural language, and can produce undesirable results when applied to other data.

The other Leximancer systems will work fine, but you will need to give some thought to the classification settings – in particular the number of sentences per segment and the possible uses of paragraph delimiters, if any.

**Tables of Data which include Text Fields**

If you have exported a database or spreadsheet table of data that includes some text fields, this can be analysed very effectively by Leximancer, but the data requires some preparation. A wizard in the File Selection dialog has been included in the newer versions of the software that guides you through the preparation.
Appendix A: A Quick Reference for Perl Regular Expressions

Perl Regular expressions are patterns designed to match text strings. Here are some of the most useful rules, which have been highly simplified.

Each character matches itself, unless it is one of: +.?*^$()[{}|\]
Those are called special characters. If you want to actually match a special character literally, then put a \ in front of it.

The special characters are used as follows:

.  This matches any single character.
^  This matches the start of the line.
$  This matches the end of the line.

[abc] This matches any one of the characters a, b, or c. You can use any characters, or some of the special patterns (specifically: \w, \s, \d, or those which look like [:alpha:] ) inside the square brackets. This is called a class. See below for those special patterns.

[0-9] This matches a single character from a range of characters, in this case digits 0 to 9. You can use letters as well.

[^a-z] This class matches any single character which is NOT in the range a to z. The up-arrow negates the match.

(dog|cat|horse) This matches either the word 'dog', 'cat', or 'horse'. You can use patterns inside the parenthesis as well.

Quantifiers.

These specify multiple instances of some pattern in a row. These are placed after the pattern.

w+ The + quantifier says to match one or more times. This matches one or more 'w' characters in a row.
G? The ? quantifier says to match zero or one times. This will match zero or one 'G' characters.
.* The * quantifier says to match zero or more times. This pattern matches zero or more of any character (i.e the '.' pattern) in a row, and will match the maximum possible.
a{2,5} The quantifier {2,5} says to match a minimum of 2 and a maximum of 5. In this case the character 'a' will be matched from 2 to 5 times in a row.
a{3} This matches exactly 3 'a' characters in a row.
a{4,} This matches at least 4 'a' characters in a row.
Some very useful special patterns are:

\w This matches any single letter, digit, or underscore ('_').
\d This matches any single digit.
[:upper:] This matches any single uppercase letter.
[:lower:] This matches any single lowercase letter.
[:alpha:] This matches any single alphabetic character.
\s This matches a single whitespace character, including space and tab.
\t This matches a single tab character.
^\s*$ This matches a single blank line.
Software Credits

Included with the Leximancer distribution are the following utilities:

**Perl** - See the file called 'Copying' in the perl distribution for the terms and conditions of the Gnu General Public License (GPL) which covers Perl, and other software as noted below. This free software license implies that you a) are offered no warranty by the authors of any software covered by that license; b) can always obtain the source code for such software; and c) are free to use and redistribute that software under the terms of the GPL. We are happy to help you obtain the source code for any such software. Generally, for Perl and its associated modules, the best place to get the latest source code is CPAN at http://www.cpan.org.

We distribute a version of Perl compiled for Windows as a courtesy and for the convenience of Leximancer users. We are not offering, or making any pretense to offer, support for Perl or associated modules. Neither are we offering any warranty for such software. Leximancer software itself is NOT distributed under the GPL.

**HTML::SimpleParse** - Copyright 1998 Swarthmore College. Author: Ken Williams <ken@forum.swarthmore.edu> Distributed under same Free Software license as Perl.

**Tie::TextDir** - Copyright 1998-2001 Ken Williams. All rights reserved. Author: Ken Williams <ken@mathforum.org> Distributed under same Free Software license as Perl.

**Antiword 0.33** – Copyright (c) 1998-2002 Adri Van Os. http://www.winfieId.demon.nl/index.html Covered by the GNU General Public License.

**CGI::Application** - Copyright (c) 2000, Jesse Erlbaum <jesse@vm.com> and Vanguard Media (http://www.vm.com/). All rights reserved. This library is free software; you can redistribute it and/or modify it under the same terms as Perl itself.

**AdminMisc extension for Win32 Perl** - Copyright (c) 1996-1999 Dave Roth. All rights reserved. Courtesy of Roth Consulting. http://www.roth.net/consult Covered by the GNU General Public License

**gzip 1.2.4** (18 Aug 93) - Copyright (C) 1992-1993 Jean-loup Gailly
This program is free software; you can redistribute it and/or it under the terms of the GNU General Public License as published by the Free Software Foundation; either version 2, or (at your option) any later version.

This program is distributed in the hope that it will be useful, but WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU General Public License for more details.
You should have received a copy of the GNU General Public License along with this program; if not, write to the Free Software Foundation, Inc., 675 Mass Ave, Cambridge, MA 02139, USA.

**pdf2text** - From Xpdf. The Xpdf software and documentation are copyright 1996-2002 Derek B. Noonburg.  
mailto:ederekn@foolabs.com  
WWW: http://www.foolabs.com/xpdf/  
Covered by the GNU General Public License.

**Globe2** - Copyright 2000 Ben Clifford  
http://www.hawaga.org.uk/c0deZ/globeApplet/  
Actually, a heavily modified applet derived from Globe2 is included.  
Original code generously made available by Ben Clifford.  
All published web pages using this applet must include attribution to Ben Clifford and a link to his site.

Some Tokenizer code from Dan Melamed's kit  
(http://www.cis.upenn.edu/~melamed/)  
has been included in Tokenize.pm, specifically the sanitize subroutine.  
This code is freely distributed under the GNU General Public License.

Code for a mini cgi server called **MWeb**, by Jeff Clement, has been included in a file called cgiserver.pl. Note that this code has been extensively modified. The following copyright notice applies to any original MWeb code:

```
Copyright © by Bluesine Consulting <mailto:jeff@bluesine.com>.  
All rights reserved.
```

Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:  
* Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer.  
* Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution.

```
THIS SOFTWARE IS PROVIDED BY THE AUTHOR AND CONTRIBUTORS "AS IS" AND ANY EXPRESSOR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE AUTHOR OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.
```

The Wordnet ([http://www.cogsci.princeton.edu/~wn/](http://www.cogsci.princeton.edu/~wn/)) stoplist of English stopwords is used as the basis for the stoplist.pl and languagetest.pl files.  
The following copyright notice applies to those two files:

```
WordNet 1.6 Copyright © 1997 by Princeton University. All rights reserved.
```
THIS SOFTWARE AND DATABASE IS PROVIDED "AS IS" AND PRINCETON UNIVERSITY MAKES NO REPRESENTATIONS OR WARRANTIES, EXPRESS OR IMPLIED. BY WAY OF EXAMPLE, BUT NOT LIMITATION, PRINCETON UNIVERSITY MAKES NO REPRESENTATIONS OR WARRANTIES OF MERCHANTABILITY OR FITNESS FOR ANY PARTICULAR PURPOSE OR THAT THE USE OF THE LICENSED SOFTWARE, DATABASE OR DOCUMENTATION WILL NOT INFRINGE ANY THIRD PARTY PATENTS, COPYRIGHTS, TRADEMARKS OR OTHER RIGHTS.

The name of Princeton University or Princeton may not be used in advertising or publicity pertaining to distribution of the software and/or database. Title to copyright in this software, database and any associated documentation shall at all times remain with Princeton University and LICENSEE agrees to preserve same.

*******************************************

All other code, documentation, and data is Copyright 2000,2001,2002 Andrew E. Smith, http://www.leximancer.com/. All rights reserved. All use must be accompanied by a valid license, and all of the intellectual property therein is proprietary and confidential and must be treated as commercial-in-confidence. This is not Open Source code, Free Software, GPL, or Shareware.
Index

Applications of Leximancer ............................................................................................................10
Author style detection ....................................................................................................................75–76
Automatic Concept Selection ........................................................................................................72
Bigrams
  Setting Bigram Sensitivity ............................................................................................................66
Boilerplate Cutoff ...........................................................................................................................66
Browsing Information using Leximancer
  The difference between Entities and Properties........................................................................42
  Using the Navigate Button............................................................................................................9
Buckets in Leximancer
  Configuring Bucket Settings .......................................................................................................46
  The Use of Buckets in Classification ..........................................................................................46
Classification and Indexing
  Configuring Classification and Indexing ....................................................................................43–46, 70–71
  Distinction between Entities and Properties ................................................................................42
Concept Seed Words
  What is a Concept Seed Word? .....................................................................................................25
Concepts in Leximancer
  Automatic Concept Selection ....................................................................................................72
  How are Concepts Learned in Leximancer? ................................................................................26–27
  Manual Concept Seeding ............................................................................................................72
  Specifying the Number of Concepts to Extract ..........................................................................31
  Viewing Leximancer Concept Definitions ................................................................................53
  What is a Concept in Leximancer? .............................................................................................24
  What is a Concept Seed Word? ..................................................................................................25
  What is the Relevancy Metric? ....................................................................................................25
Conceptual Analysis
  Assumed Versus Inferred Categories ...........................................................................................21
  How do You Distinguish Concepts? ............................................................................................21
  Reliability ...................................................................................................................................21–22
  What is Conceptual Analysis? ....................................................................................................20
  What is the Appropriate Level of Analysis? ................................................................................20
Content Analysis
  Issues Involved ...........................................................................................................................20–22
  Types of Content Analysis ..........................................................................................................20
  What is Content Analysis? ...........................................................................................................19
  What Types do Leximancer Perform? ..........................................................................................22
Documents
  Specifying the Start of Documents .............................................................................................64
Entities
  The Difference between Entities and Properties ........................................................................42
File Formats
  Dealing with Unusual File Formats ............................................................................................44–45
  Supported File Types ..................................................................................................................11
Folder Tags .....................................................................................................................................30, 72
Gaussian Maps ..................................................................................................................................48–49
Go-List
  Adding Automatically Removed Stopwords Back into the Analysis .............................................29–30
<table>
<thead>
<tr>
<th>Topic</th>
<th>Page(s)</th>
</tr>
</thead>
<tbody>
<tr>
<td>What is the Go-List?</td>
<td>29-30</td>
</tr>
<tr>
<td>Kill Classes</td>
<td>43</td>
</tr>
<tr>
<td>Language</td>
<td></td>
</tr>
<tr>
<td>Processing Documents in a Foreign Language</td>
<td>77</td>
</tr>
<tr>
<td>Specifying the Language of the Documents Processed</td>
<td>61</td>
</tr>
<tr>
<td>Language Removing Non-Sentences from the Analysis</td>
<td>63</td>
</tr>
<tr>
<td>Language Testing</td>
<td>28</td>
</tr>
<tr>
<td>Removing Non-Sentences from the Analysis</td>
<td></td>
</tr>
<tr>
<td>Learning in Leximancer</td>
<td></td>
</tr>
<tr>
<td>Allowing Follow-On Learning</td>
<td>68</td>
</tr>
<tr>
<td>How are Concepts Learned?</td>
<td>26-27</td>
</tr>
<tr>
<td>Learning Threshold</td>
<td></td>
</tr>
<tr>
<td>Specifying the Learning Threshold</td>
<td>38</td>
</tr>
<tr>
<td>Leximancer</td>
<td></td>
</tr>
<tr>
<td>Applications</td>
<td>10</td>
</tr>
<tr>
<td>Browsing Information using the Map</td>
<td>9</td>
</tr>
<tr>
<td>Configuring and Running Classification and Indexing</td>
<td>43-46, 70-71</td>
</tr>
<tr>
<td>Configuring and Running Concept Learning</td>
<td>35-40</td>
</tr>
<tr>
<td>Configuring and Running Mapping</td>
<td>48-49</td>
</tr>
<tr>
<td>Configuring and Running Text Preprocessing</td>
<td>61-64, 28-30</td>
</tr>
<tr>
<td>What is Leximancer?</td>
<td>4</td>
</tr>
<tr>
<td>Linear Maps</td>
<td>48-49</td>
</tr>
<tr>
<td>Manual Concept Seeding</td>
<td>72</td>
</tr>
<tr>
<td>Map</td>
<td></td>
</tr>
<tr>
<td>Configuring and Running Mapping</td>
<td>48-49</td>
</tr>
<tr>
<td>How is the Map Generated?</td>
<td>7</td>
</tr>
<tr>
<td>Map Controls</td>
<td>50-52</td>
</tr>
<tr>
<td>The Difference between Gaussian and Linear Maps</td>
<td>48-49</td>
</tr>
<tr>
<td>What Information is Shown on the Map</td>
<td>5-8</td>
</tr>
<tr>
<td>Name Binding</td>
<td>62</td>
</tr>
<tr>
<td>Name-Like Concepts</td>
<td>31</td>
</tr>
<tr>
<td>Specifying the Number of Names to Extract</td>
<td>31</td>
</tr>
<tr>
<td>Navigating Information using Leximancer</td>
<td>See Browsing Information</td>
</tr>
<tr>
<td>Optimising Performance</td>
<td></td>
</tr>
<tr>
<td>Preventing Prepared Data Deletion</td>
<td>63-64</td>
</tr>
<tr>
<td>Sampling Information During Learning</td>
<td>68</td>
</tr>
<tr>
<td>Setting Memory Usage Options</td>
<td>63</td>
</tr>
<tr>
<td>Paragraphs</td>
<td></td>
</tr>
<tr>
<td>How to Ignore Paragraph Boundaries in the Analysis</td>
<td>39</td>
</tr>
<tr>
<td>How to Ignore Paragraph Boundaries in the Analysis</td>
<td>66</td>
</tr>
<tr>
<td>Specifying Paragraph Boundaries</td>
<td>64</td>
</tr>
<tr>
<td>Perl Regular Expressions</td>
<td>78-79</td>
</tr>
<tr>
<td>Plagiarism detection</td>
<td>75-76</td>
</tr>
<tr>
<td>Profiling</td>
<td>73, 76</td>
</tr>
<tr>
<td>Proper Nouns</td>
<td>See Name-Like Concepts</td>
</tr>
<tr>
<td>Properties</td>
<td>73</td>
</tr>
<tr>
<td>Browsing Property Locations</td>
<td>56</td>
</tr>
<tr>
<td>The Difference between Entities and Properties</td>
<td>42</td>
</tr>
<tr>
<td>Property Tuples</td>
<td>70</td>
</tr>
<tr>
<td>Regular Expressions</td>
<td>78-79</td>
</tr>
</tbody>
</table>
Relational Analysis ................................................................. 22
Required Classes .................................................................. 43, 73
Running Leximancer ............................................................. 11–12
Semantic Characterisation ...................................................... 74
Semantic Comparison of Concepts ........................................... 74–75
Semantic Discrimination of Concepts ....................................... 75
Sentences
  Specifying Sentence Boundaries ............................................ 64
  Specifying the Number of Sentences per Context Block ............ 38, 45
Social Networks ..................................................................... 76–77
Software Credits .................................................................... 80–82
Stopwords
  Adding Automatically Removed Stopwords Back into the Analysis .... 29–30
  Specifying a Stopword Removal Pattern ................................. 62
Supervised learning .................................................................. 75–76
Supported File Types .............................................................. 11
Text Preprocessing
  Configuring and Running ...................................................... 61–64, 28–30
  What does this Phase do? ......................................................... 28
Thesaurus ................................................................................. See also Concepts in Leximancer
  Viewing the Thesaurus ............................................................ 53
Toolkit .................................................................................. 72–73
Vocabularies
  Viewing the Words used in the Context of Concepts .................. 56
Web Pages
  Viewing Remote Web Pages through the Map ......................... 71