An investigation into the use of linguistic context in cursive script recognition by computer

An Investigation into the Use of Linguistic Context in Cursive Script Recognition by Computer

by

Neil Howard Brammall

This work was carried out in the Department
of Computer Studies at Loughborough University

Abstract

The automatic recognition of hand-written text has been a goal for over thirty five years. The highly ambiguous nature of cursive writing (with high variability between not only different writers, but even between different samples from the same writer), means that systems based only on visual information are prone to errors.

It is suggested that the application of linguistic knowledge to the recognition task may improve recognition accuracy. If a low-level (pattern recognition based) recogniser produces a candidate lattice (i.e. a directed graph giving a number of alternatives at each word position in a sentence), then linguistic knowledge can be used to find the 'best' path through the lattice.

There are many forms of linguistic knowledge that may be used to this end. This thesis looks specifically at the use of collocation as a source of linguistic knowledge. Collocation describes the statistical tendency of certain words to co-occur in a language, within a defined range. It is suggested that this tendency may be exploited to aid automatic text recognition.

The construction and use of a post-processing system incorporating collocational knowledge is described, as are a number of experiments designed to test the effectiveness of collocation as an aid to text recognition. The results of these experiments suggest that collocational statistics may be a useful form of knowledge for this application and that further research may produce a system of real practical use.

Acknowledgements

I would first like to thank my supervisor Dr. John Connolly and my director of research Dr. Chris Hinde for all their help. This thesis would never have been written without them.

I would also like to thank Graham Gerrard and Rob Kirkwood of Computing Services at Loughborough University for copious amounts of filestore and for their tolerance, Dr. Murray Holt for providing me with sample word lattices, and Dr. Lindsay Evett for lunch and words of advice.

A special thank you to Anne for her support and understanding, to Charlie King's Big Fight, The Chasers and Carlton for the music, and to my friends and housemates for all the tea and sympathy.

Last but not least I would like to thank my mum and dad, and all my family, for their love and support and for reminding me to eat.

Contents

List of Figures

Chapter 4

Chapter 5

The gods did not reveal, from the beginning,
All things to us, but in the course of time
Through seeking we may learn and know things better.
But as for certain truth, no man has known it,
Nor shall he know it, neither of the gods
Nor yet of all the things of which I speak.
For even if by chance he were to utter
The final truth, he would himself not know it :
For all is but a woven web of guesses.

Xenophanes, 6th century BC.

Chapter 1

Introduction

Giant strides have been taken in the improvement of the technological capabilities of electronic computers since their introduction in the 1940s.

While this progress continues with seemingly no end in sight, there has in recent years been an increasing concentration on the means by which humans communicate with computers.

While the ever-growing use of windows-based environments offers a far more intuitive interface than the old command-line methods, much communication is still keyboard-based.

A much-discussed alternative to keyboard-based communication is communication by means of speech and/or handwriting. There are of course fundamental differences between the two modes of communication, as discussed later.

This thesis will concentrate on the use of handwriting as a means of communication with a computer.

1.1 Text Recognition

Written communication with a computer can come in two forms.

Firstly, a user can communicate directly with the computer by writing on some form of graphics tablet or, increasingly, straight onto the screen of a hand-held computer. This takes away the need for the user to learn keyboard skills before they can interact effectively with the machine. This method is also ideal for gestural input such as pointing to on-screen objects, crossing out mistakes and so forth.

Secondly, existing documents can be scanned into the computer and then processed as electronic documents, circumventing the need to laboriously type them in.

Clearly a high level of recognition accuracy is absolutely vital for both these methods of communication.

The recognition by computer of text has been a goal for over thirty five years. Up until quite recently, systems attempting recognition have generally based their attempts on purely visual information.

While increasingly sophisticated methods have brought about a marked improvement in accuracy since the early days, there seems to have been a law of diminishing returns at work, i.e. there appears to be a ceiling of accuracy above which methods employing purely visual information cannot go when attempting to recognise handwriting from a wide variety of sources.

It should be noted that recent years have seen considerable advances in the field of online recognition, but the available systems are invariably trained by one particular user, and recognition accuracy is seen to tail off when this training is curtailed, or when an unknown user is entering text.

1.2 The Use of Linguistic Knowledge

The benchmark against which the performance of text recognition systems has traditionally been judged is human performance. Humans have a remarkable (though far from infallible) ability to read even the most visually degraded text. Clearly sources of information above and beyond the merely visual are at work here.

As many studies have shown (see **2.3 – Linguistic Knowledge as an Aid to Text Recognition**), humans use many levels of knowledge to interpret handwriting and indeed any other image.

Pragmatic knowledge, or world knowledge, may place a document in a particular context. For instance, a letter from a bank is likely to concern financial affairs, and the reader will be primed for this and for the style and content of the language associated with it immediately upon discovering the letter's source.

Lower levels of knowledge will come into play on smaller units of language. At the sentence level, semantic and syntactic knowledge may permit the read to hazard a guess at an illegible word by considering the words surrounding it.

One more level down – at the word level – morphological knowledge (relating to the grammatical components of words) and orthographic knowledge (relating to the shapes of letters) may help to distinguish a word where no clues are forthcoming from a wider context.

This complex use of and interaction between different types and levels of linguistic knowledge is of course quite likely to be carried out completely subconsciously by a human reader.

It is clear that incorporating at least some of these knowledge sources into the automatic recognition of text can offer potential improvements in performance.

A component of a text recognition system that exploits linguistic knowledge sources can be viewed as a post-processing 'black box' to a recognition stage acting on visual information.

This 'low-level' recogniser will produce as output not a definitive statement as to the identity of an input image, but a set of hypotheses for each entity in the input image.

The job of the linguistic knowledge-based component is to select the most appropriate hypothesis for each entity according to the knowledge sources at its disposal (see **Fig. 1.1**).

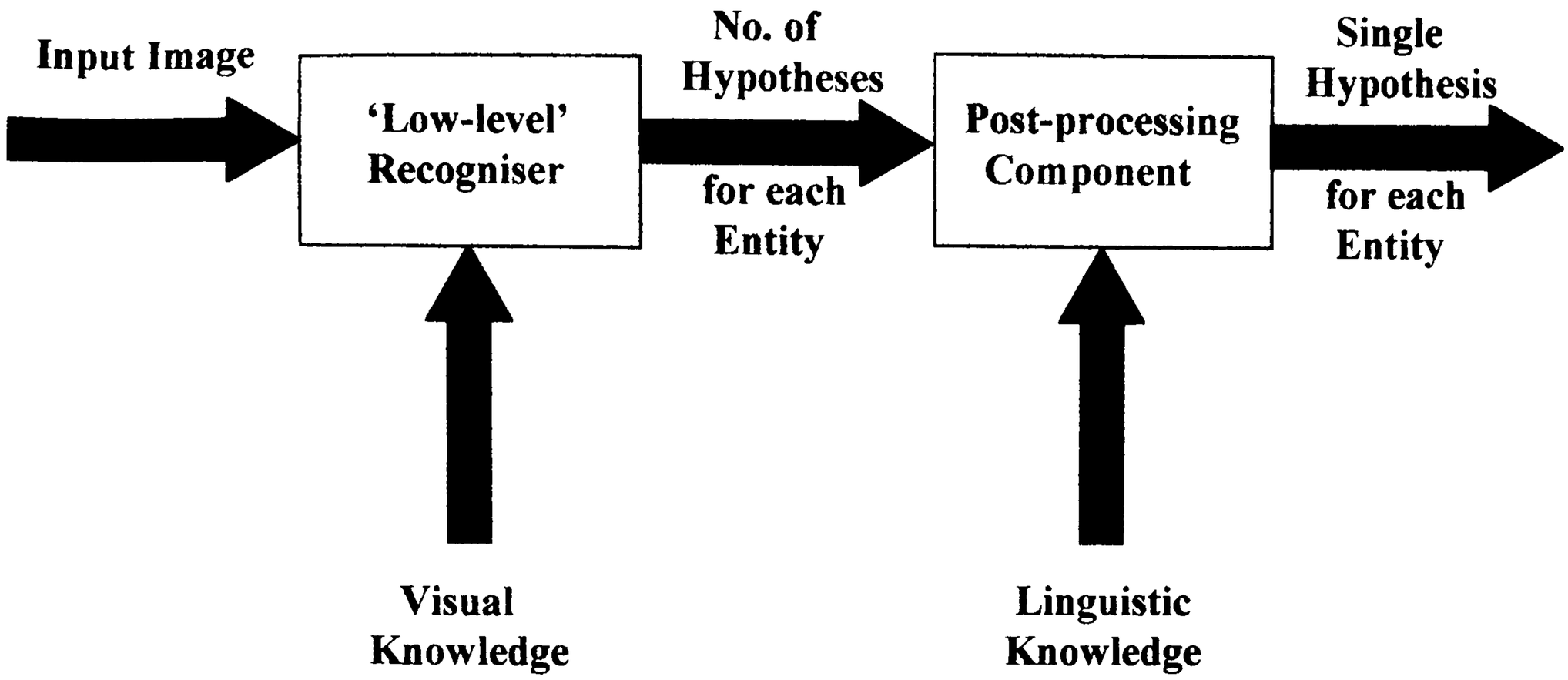

Fig. 1.1 – The role of the post-processing component in a recognition system

Let us assume that the input image is an English sentence, and that each entity in the input image is an English word (although there may also be an intermediate character processing stage).

The output from the low-level recogniser will therefore be a set of hypotheses for each word position in the sentence, often represented as a **word lattice** (see **Fig. 1.2**).

Input sentence :

Come fill the cup

Word lattice :

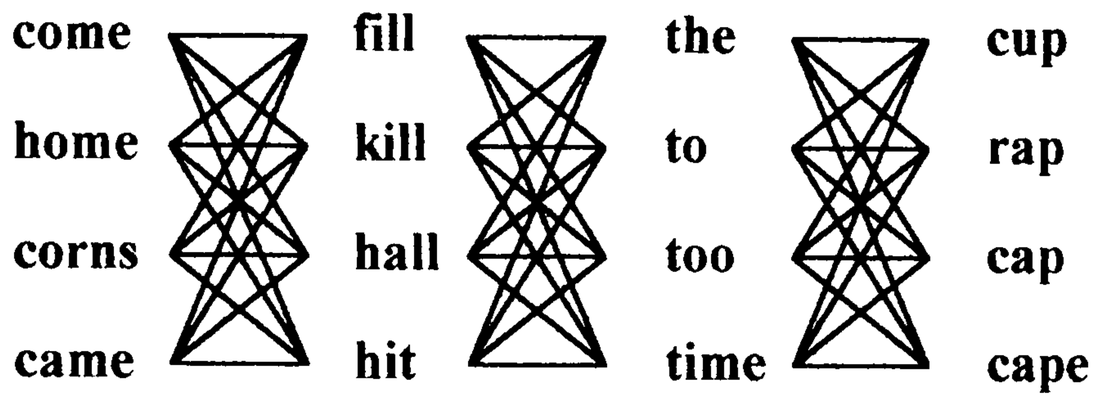

Fig. 1.2 – A sentence represented by a word lattice

The post-processing component now has to choose the most appropriate word at each word position in the sentence from these hypotheses. This is equivalent to choosing the path through the lattice which best matches the input image. For a 4x4 lattice as above the number of possible paths through the lattice is $4^4 = 256$ paths.

1.3 The Use of Collocation

Collocation is the habitual association of a word in a language with other particular words in that language. A collocation may occur between words in adjacent positions or over a wider frame of reference.

Collocation differs from syntax in that each word is considered as an individual lexical item associating with other individual lexical items as opposed to being a member of a class of words associating with words also belonging to classes.

It is my contention that this tendency of particular words to predict their environment is a useful source of knowledge when reading and may therefore be used to improve the performance of an automatic text recognition system.

A number of studies (e.g. Rose & Evett, 1993, and Hull, 1994) have produced results which indicate that this may be the case.

The aim of this thesis is to pursue this argument further by carrying out detailed analysis on a large body of text to create a collocational knowledge source, and then using this knowledge source alone to identify the correct input sentence from a number of hypotheses. It is suggested that this will give a clearer picture than is currently the case as to whether, and to what extent, collocational knowledge is an appropriate knowledge source to exploit in automatic text recognition.

1.4 System Overview

As discussed previously, the exploitation of collocation is carried out as a post-processing step.

The collocational knowledge is represented as a collocation dictionary containing information about the collocational relationships contained in a corpus, or collection of texts. An entry in this collocation dictionary is in the following generic form :

Word A Id
Word B Id (Position) Frequency Strength

This entry represents a collocation between two words (**A** and **B**). The words themselves are represented by **Word Ids**, as this is a more efficient representation than storing the text of the word itself.

Position represents the position of word **B** in relation to word **A**. This field is shown in brackets as positional information will not always be stored.

Frequency represents the number of times that the collocation between **A** and **B** occurs in the text under analysis.

Strength gives a measure of the significance of the collocation between the two words.

A dictionary containing entries as above for all the collocations in a given corpus provides a clear picture of the collocational behaviour of the words in that corpus.

An actual low-level recognition system was not available during this project to provide input for the post-processing component. It was therefore necessary to simulate the results produced by a low-level recogniser.

This was done by analysing some sample output produced by an actual recogniser and generalising this to provide a filter which could be applied to any input text.

The input to this filter is an English sentence and the output is a set of hypotheses for each word position in that sentence in the form of a word lattice (see **Fig. 1.3**).

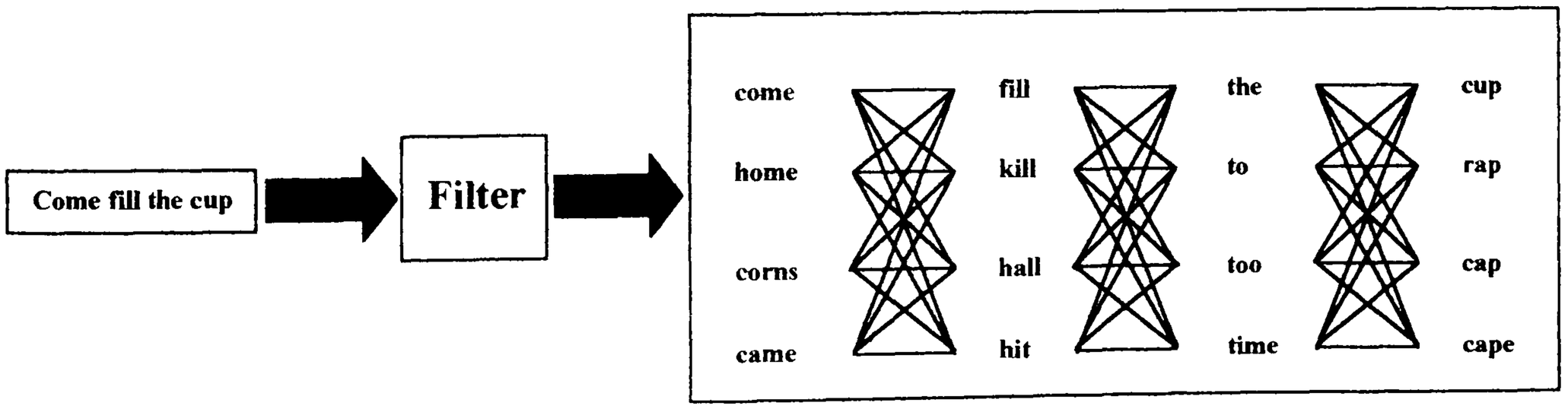

Fig. 1.3 – A filter to produce a word lattice from a sentence

It should be noted that some lexical analysis is also carried out at this stage in order to remove from the lattice any words that are not recognised by the system. The set of words recognised by the system is stored in a lexicon based on the **Collins Electronic Dictionary**. There are **78,055** words in this lexicon.

The post-processing component takes the word lattice produced by the low-level simulator illustrated above as input. The job of the post-processing component is to select the path through the word lattice representing the sentence which most resembles the original input sentence, based on the knowledge contained in the collocation dictionary.

A diagrammatical overview of the system is given in **Fig. 1.4**.

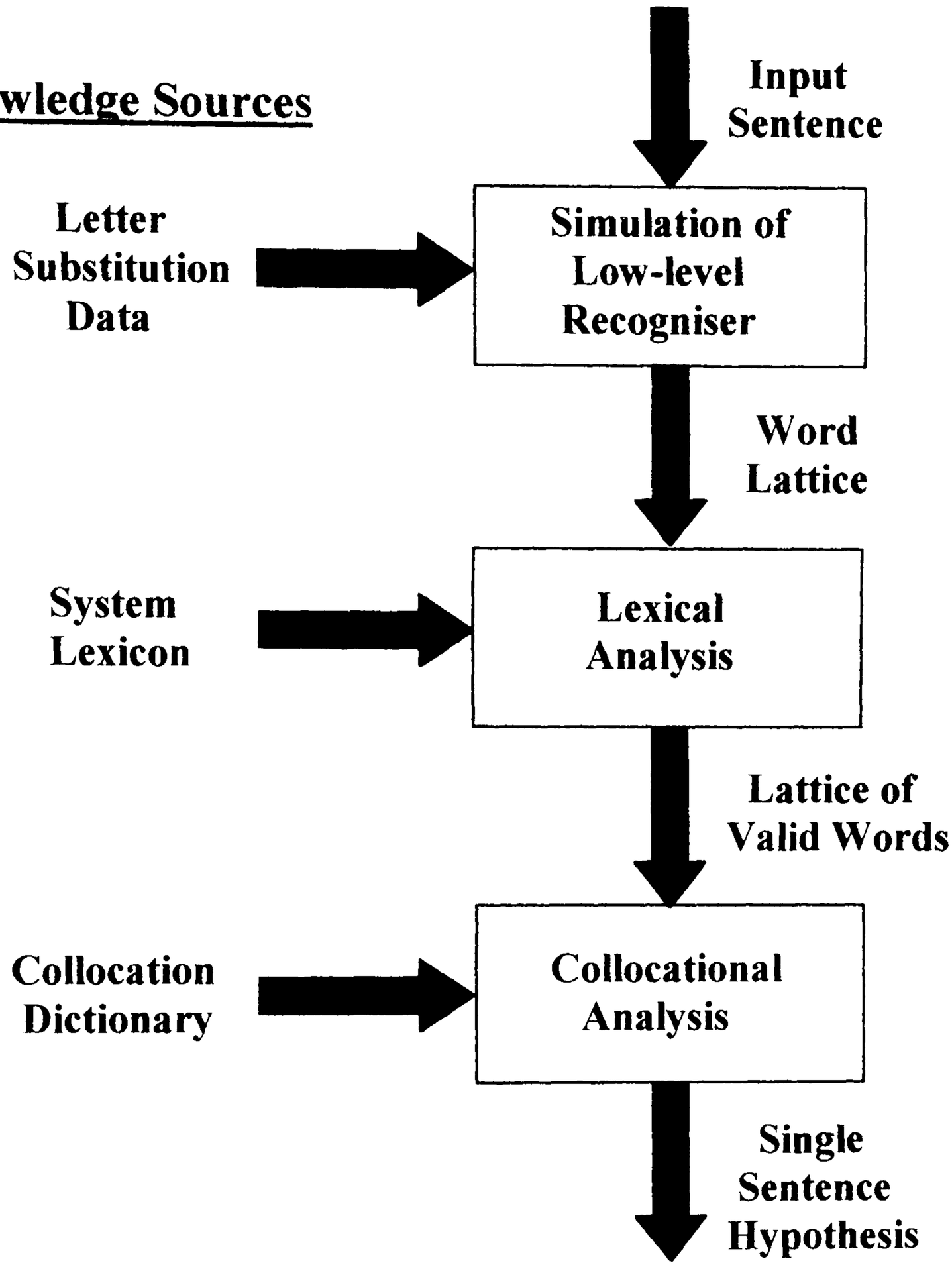

Fig. 1.4 – An overview of the system

The hypothesised sentence produced can then be compared to the original input sentence to produce a measure of success or failure.

A number of experiments were carried out to test the system using various different criteria.

For the purposes of experimentation there were three variables to consider :

- the input text
- the collocation dictionary
- the lexicon

These entities were combined in different ways to give different criteria against which the system could be tested.

Two sources of input text were used to test the system. One of these texts was the same as that used to compile the collocation statistics during the creation of the system, while the other text was drawn from a completely unrelated source.

Using these two sources of input was intended to contrast how the system performs when working with data it has been trained on with how it performs when working with generic data of which it has no specific knowledge.

The system performed better when operating on the data that it had been trained with, but performed creditably well when presented with data it had never seen before.

The second variable to be considered was the collocation dictionary.

During the construction of the system, two collocation dictionaries were compiled, each representing the collocational information about the text in a different way.

One method of representation was more compact than the other but less detailed and therefore allowed shorter processing times. The other representation gave detailed information about all collocations in the text which made for more accurate, but slower processing.

As would be expected, the full representation gave better results than the compact dictionary. The decision of which dictionary to use would depend on the environment in which the system was being used, and the relative importance of accuracy to speed of processing.

The final variable in the experiments was the lexicon used. The collocation dictionary and the lexicon are inextricably linked in the system, so to change the lexicon is to change the collocation knowledge base.

A number of experiments were carried out using a lexicon tailored for a specific input text (i.e. the lexicon consisted of a list of every word in the text). A collocation dictionary was then derived from the text using this lexicon. Essentially this gives a system tailored for a specific input, and experiments on this system give an idea of performance in an environment in which there was a high level of prior knowledge about the input.

This tailored system did indeed compare favourably with the totally generic system.

1.5 Summary

In summary then, I propose that accurate text recognition by computer would be a highly desirable feature. It has been found that recognition based on purely visual information does not provide the level of accuracy necessary to achieve this aim.

Observation of human performance in the reading of visually degraded text has shown that various sources of linguistic knowledge are called upon to facilitate the task of recognition.

It is suggested that the incorporation of linguistic knowledge in automatic text recognition may therefore improve recognition accuracy.

In particular, it is proposed that collocation is a suitable knowledge source for an automatic text recognition system to exploit.

Previous studies have indicated that this may be the case, and this thesis aims to investigate this contention further by constructing a large collocational knowledge base and using this as the sole criterion upon which a hypothesis as to the identity of an input image is based.

A number of experiments were carried out to test the effectiveness of this collocational knowledge base. The results of these experiments suggest that the exploitation of collocational knowledge can indeed improve the accuracy of a handwriting recognition system.

Chapter 2

Literature Survey

This survey will summarise four main areas of research.

First I will briefly discuss the literature which studies the motivation behind using natural language as a means of communication with a computer.

The next section will review the efforts put into the visual recognition of text by computer since the early 1960s.

Section three will discuss the use of linguistic knowledge as an aid to natural language recognition.

Finally, I will review the study of collocation both generally and as an aid to handwriting recognition.

2 . 1 - The Use of Natural Language in the Human-Computer Interface

Communication with a computer using natural language has been the dream of science fiction writers for decades.

It is clear that an effective natural language interface would offer intuitive communication with a computer without the need to learn keyboard skills.

While this thesis concentrates on off-line communication, where the natural language is pre-prepared and presented *en masse* to the computer, this section of the literature survey will look at general natural language communication, both on-line and off-line.

First of all, it is important to distinguish between the use of speech and the use of handwriting as a means of communicating with a computer. Many studies have analysed the differences between the two means of communication.

An early study described in Blankenship, (1962), concluded that there are no inherent syntactic differences between speech and writing and that any noticeable differences between the two forms were due to individual style. However numerous subsequent studies refute this contention, and find many differences both in the form and the type of information communicated by speech and by writing.

O'Donnell, (1974), in particular suggested that limitations in Blankenship's system of analysis may have accounted for her not finding any clear-cut grammatical differences between speech and writing. It should also be noted that Blankenship's study is based on a very small sample of four people, each of whom provided one written text, and one spoken passage. O'Donnell studied units of language called 'T-units', consisting of one independent clause and any dependent clauses syntactically related to it. So for instance the sentence :

> *"It is obvious that anyone who presides over an organisation of more than two million people is going to be both admired and hated."*

consists of one T-unit as defined by O'Donnell, whereas the sentence :

> *"He saw it; he liked it; he bought it."*

consists of three T-units.

The main advantage given for using T-units is that, unlike sentences, they can be objectively identified in both speech and writing.

Limited experiments showed handwriting to be more elaborate and structurally complex than speech, in that writing tended to contain a greater number of T-units than speech and these T-units tended to have a greater average length than those found in speech.

This hypothesis was backed up in Poole & Field, (1976). They compared speech and writing along the following dimensions :

- structural complexity
- language elaboration
- verb complexity
- personal reference

Their results suggested that written systems are more complex in structure, showed more adjectival elaboration, had a more complex verb structure and contained fewer indices of personal reference than speech,

Chafe, (1982), also noted this detached quality of writing, owing to the general lack of direct interaction between the writer and his or her audience. Chafe also concentrated on the differences in speed, estimating that writing progresses at one tenth of the pace of speech. This difference in speed was attributed mainly to the greater concentration on the organisation of language during writing. Chafe also backs up the earlier findings of O'Donnell. While not explicitly referring to T-units, he maintains that writing generally features many more sub-clauses than speech.

Biber, (1986), offers a review of the previous work in this field, and an attempt to tie up all the findings to date. The conclusion was that written language was more general and detached, more elaborate in its structure and more explicit, in the sense that the vocabulary was more precise than that used in speech.

The above findings suggest that communicating with a computer using speech may be suitable for issuing brief, specific commands, whereas if a larger amount of structured information were needed to be communicated, written communication would be preferable.

Conclusions relating directly to the use of linguistic statistics such as collocation in the recognition of natural language can also be drawn from these findings. When compiling statistics relating to the way in which words in a language tend to co-occur, one must decide upon what constitutes co-occurrence. I.e. how close together in a passage of language must two words be before we can say that they have co-occurred in that passage?

The findings discussed above suggest that, due to the greater complexity of writing than speech, we must allow a greater span of co-occurrence when dealing with handwriting than we would with speech, as words a considerable distance apart in a written passage (i.e. with many other words in between them) are more likely to be part of the same "idea unit" (from Chafe, 1982), than they would be in a spoken passage.

A more detailed discussion of this concept of collocational span will take place in section **2.4**.

A number of studies have looked at the use of written communication with a computer in comparison to other forms of input.

Mahach, (1989), compared input with a light pen with input by mouse, cursor keys, and alphabetic keys on a keyboard.

The pen was found to be superior for gestural input (pointing, striking through words etc.) but very poor for entering text in terms of speed and accuracy. However, this was attributed to the failure of the recognition software rather than any inherent flaw in the interface. This is a recurring theme in many studies.

Wolf, Rhyne and Ellozy, (1989) and Wolf, (1990), reached similar conclusions. Users were very positive about handwriting as a means of communicating with a computer, but were repeatedly frustrated by recognition errors.

Carr, (1991) and Wolf, Glasser and Fujisaki, (1991), found that systems which allowed user training of the system produced more encouraging results.

Briggs *et al.*, (1992 and 1993), again found that, importantly, users were in favour of the *concept* of handwritten input, but disliked the limitations of the software.

All these studies suggest that a human-computer interface based on handwriting would be highly desirable if it were fast enough and had high enough recognition accuracy.

It should be noted that the technology of on-line handwriting recognition has moved on greatly since these studies were carried out. Hand-held machines such as the Apple Newton™ (see Yaeger, 1997), after dubious beginnings in the early 1990s, now offer high recognition rates after training. Recognition engines such as Graffiti have also arrived on the scene, exploiting the gestural input which is such an advantage of pen-based technology.

The irony is that as on-line handwriting recognition systems are becoming a realistic proposition the need for them, certainly in the workplace, appears to be diminishing as increasing numbers of people develop keyboard and mouse skills (see for instance Wheelwright, 1996).

It would seem however that there is still a real need for proficient, accurate off-line recognition of cursive handwriting.

Many repositories of data are still paper-based, and as the desirability of electronic access to this data grows ever greater, the problem of transferring the data from paper to a digital format must be addressed.

An off-line handwriting recognition system offering high levels of accuracy would appear to be a solution. This thesis contends that for such high levels of recognition to be achievable, an element of linguistic knowledge should be inherent in any such system.

2 . 2 - The Visual Recognition of Text by Computer

This thesis does not deal directly with the visual recognition of text, so this section of the literature survey is intended to offer merely a brief exploration of the work undertaken in this field.

There are many ways to categorise the different approaches taken over the years to the recognition of text by computer.

There are character recognition and whole word recognition, on-line and off-line recognition and many other variations.

In the early days of handwriting recognition, two distinct approaches stand out :

- analysis by synthesis
- analysis on a letter by letter basis

The first method is typified by the work described in Eden & Halle, (1961), Eden, (1962), and Matthews, (1961).

In analysis by synthesis, a model of human handwriting is created, and recognition is performed by matching the input to this model.

The model presented by Eden & Halle consisted of a number of primitive shapes. They suggested that each letter of the alphabet could be made up of a combination of these primitives. Such a hypothesis will of course struggle when presented with the high variability inherent in handwritten text.

Matthews suggested achieving recognition by generating a set of sentences, the number of which is limited using various criteria, then matching the input to one of the generated sentences. The enormous number of sentences generated, even after the restrictions were applied, made this approach practically infeasible, particularly in 1961 when the filestore available would have been severely limited. Even with today's machines however the vast number of potential sentences that would be generated would in all likelihood prove restrictive.

The second approach - letter by letter analysis - is typified by the work put forward by Frishkopf & Harmon, (1961) and Harmon, (1962).

The big problem encountered by these studies was attaining the correct segmentation of a word into individual letters. This word segmentation problem will be a recurring theme throughout this section. These problems resulted in low recognition rates (around 30% word recognition) for Frishkopf and Harmon's system. However, Harmon in his study of 1962 saw a way forward by introducing simple contextual constraints to the recognition process to perform error correction. These constraints operated at the letter level in the form of letter bigram frequencies, and resulted in considerable improvements in recognition accuracy.

One way around the letter segmentation problem is to deal with the input on a word by word basis, as the segmentation of a sentence into words is generally a far easier task than the segmentation of a word into letters. This approach was taken by Earnest, (1962).

There would inevitably have been problems of storage with this approach at the time of this study. Whereas there are 26 letters in the English alphabet, there are obviously thousands of words. This would not cause a problem today in terms of available filestore, but efficient look-up algorithms would still obviously be necessary. Earnest claimed to achieve around 60% word recognition accuracy with his system.

Another avenue explored in these early days of handwriting recognition was the use of temporal information - see especially Brown, (1964) and Mermelstein & Eden, (1964).

Here, words were entered by a light pen on a tablet, and information relating to the movements of the pen during word formation was used to try and determine the characters being written.

Reviews of these early attempts at handwriting recognition can be found in Lindgren, (1965), and in Harmon, (1972).

An excellent overview is also provided by Sayre, (1973). He argued against the use of temporal information as an aid to recognition, as this information is not generally available to humans when reading handwriting, and that our aim should be to simulate human performance as closely as possible.

Sayre proposed a system of letter segmentation to provide a number of alternatives at each letter position in a word, and then the use of bigram statistics to choose one of these alternatives, (see **2 . 3 - Linguistic Knowledge as an Aid to Text Recognition** for a much more detailed description of the use of n-gram statistics).

This "on the fly" method of segmentation, whereby no final letter segmentation is fixed upon until the last minute marked a great breakthrough. This is the basis for segmentation for many systems today.

Sayre's paper marked something of an end to what can be seen as the first wave of work directed at automatic handwriting recognition. This was followed by a hiatus until the late 1970s and early 1980s, as many of the problems encountered during these early studies were considered virtually intractable.

Farag, (1979), marks the beginning of a fresh effort in this area. Farag avoids the problem of word segmentation by using temporal information to identify whole words. This word-level recognition system achieved remarkable recognition rates (98 – 100%) for a very small number of key words, making it suitable for limited vocabulary domains.

Another way of avoiding segmentation problems was suggested by Tappert, (1982). Rather than explicitly segmenting words, segmentation and recognition were combined into one operation by evaluating recognitionfor each possible segmentation of a word. This was a development of Sayre's idea of avoiding rigid (and possibly incorrect) segmentation andwould be taken up in many future studies.

Letter segmentation is still a big thorn in the side for systems based on purely visual information. A number of studies deal specifically with this problem.

Balestri, (1988), proposed a method whereby the 'goodness' of each potential segmentation of a word was evaluated using stochastic methods.

A number of studies - Badie & Shimura, (1982), Holt, Beglou & Datta, (1992) and Houle, (1994), put forward methods for tracing the contour of a written word, and identifying likely segmentation points based on this trace.

Dunn & Wang, (1992), reviewed the literature in this problematic area, and identified two main categories ofapproach : explicit segmentation, and segmentation-recognition, as described earlier in Tappert (1982). They concluded that segmentation techniques used in isolation were not suitable for cursive script recognition owing to the high level of ambiguity and variability. They recommended the technique of hypothesise then test.

This method is pursued in many studies. Among them are Bozinovic & Srihari, (1989), Ouladj *et al*., (1989), Edelman, Flash & Ullman, (1990) and Fujisaki *et al*., (1991).

The effort to avoid the word segmentation problem by using whole word matching also continued, for instance in Hull, Khoubyari & Ho, (1992), in Caesar *et al*., (1994) and in Gorsky, (1994).

Cheriet, (1994), proposed a system which combined whole word and character recognition, by attempting to identify global features, then extracting key letters based on this information.

In the meantime, the recognition of isolated or printed characters could virtually be considered a solved problem.

Mandler, Oed & Doster, (1985), described a method with a high success rate in the recognition of isolated handwritten characters, especially when temporal information was exploited. Higher error rates were observed for off-line recognition however.

Kahan, Pavlidis & Baird, (1987), described a system for the recognition of machine printed characters which could deal with variations in font and size, and with slight variations in the orientation of the page.

An area that saw much interest in the late 1980s and early 1990s to the present was the use of neural networks in handwriting recognition. A full study of this field would constitute a literature survey in itself, but I shall discuss a handful of representative studies.

The general approach is to represent some linguistic entity as a node in a neural network. Weightings are then assigned to give some sort of score for each node, representing its similarity to the input.

Morasso *et al.*, (1990), and Morasso & Pagliano, (1991), proposed allographs as these basic units (i.e. alternative graphical structures representing the same symbol). Experiments using this method yielded recognition rates of around 75 – 80%.

Pittman, (1991), presented three neural nets. One worked on purely visual input, another made use of temporal stroke information, and a third combined the outputs of the other two.

Hoffman, Skrzypek & Vidal, (1993), proposed a network made up of a limited number of primitive hand motions, used to recognise isolated characters.

Senior, (1994), proposed a neural network to carry out graphical pre-processing steps on an input image, including shearing, rotation and scaling, to make subsequent processing more straightforward.

In the commercial sphere, the Apple Newton uses neural network technology in the training of the recognition engine for a particular user (see Yaeger, 1997).

Another interesting strand of research acknowledged that many successful recognition systems exploited temporal information captured at the time of writing, and looked at ways to extract this kind of information from an off-line image (i.e. an image presented to the system some time after being written).

Govindaraju, Wang & Srihari, (1992), Doermann & Rosenfeld, (1992) and Boccignone *et al.*, (1993), all proposed methods for doing this.

Also of interest is the work carried out to analyse text at the document level. These studies isolate particular regions of interest in a document, such as an address block, a post code or a signature, before carrying out recognition on this area, often using a limited lexicon.

Srihari, (1992), concentrated on address blocks on items of mail, while Downton *et al.*, (1992), looked specifically at post codes.

Tang & Suen, (1992), proposed a more general method of converting a compound pattern (one of many parts) into an integral one ready for recognition.

Much work in recent years has gone into combining the results of many different (and different types of) classifiers to improve recognition.

A number of hierarchical systems exploiting different levels of linguistic knowledge to this end are described in the next section.

However, there are many studies which describe attempts to combine the results of a number of low-level systems.

Hull *et al.*, (1992), combined character and word level recognition systems (similar, in concept at least, to Cheriet, (1994), discussed earlier).

Ho, Hull & Srihari, (1992), concentrated on the methods used to actually combine the results of different recognisers in the most effective way. Good results were recorded, with the most effective combination method giving the correct word as one of the tenhighest ranking alternatives around 95% of the time.

Further such combination methods were investigated by Xu, Krzyzak & Suen, (1992).

There have been many useful surveys over the years giving the state of the art in handwriting recognition.

Mantas, (1986), concentrates on the recognition of isolated characters, both printed and handwritten.

Tappert, Suen & Wakahara, (1990), is a review of recognition techniques which exploit temporal information.

Dimauro, Impedovo & Pirlo, (1992), is a general survey of character and word recognition techniques. They suggest that the way forward is to attain a better understanding of human performance in this area.

The same conclusion, amongst others, is drawn in Suen*et al.*, (1993), a review of the literature which concentrates on off-line systems, as does Srihari, (1996).

Lecolinet & Baret, (1994) give a detailed general review of cursive script recognition.

I should like to end this whistle-stop tour of low-level handwriting recognition with reference to the study of Simon, (1994).

He comments on the robustness of human recognition of even visually degraded images, and suggests that automatic text recognition systems should ultimately aim to emulate human performance.

This leads onto the next session, which is a review of the literature relating to the use of linguistic knowledge as an aid to handwriting recognition.

2 . 3 - Linguistic Knowledge as an Aid to Text Recognition

2 . 3 . 1 - The Human Use of Context in Reading

Many studies going back many years have considered the effect of context on human perception of text.

Cattell, (1885), presented human subjects with a series of numbers, letters, words and sentences for short intervals of time. He found that the subjects could grasp around four numbers, three-four letters, two words or a sentence composed of four words.

Letters were slightly more difficult to grasp than numbers, as every combination of digits gives a 'meaningful' number.

Not as many words as letters were grasped at one time, but three times as many letters were grasped when they made words (i.e. when they were put into a meaningful context) than when they had no connection.

Similarly, twice as many words were grasped when they made a sentence (i.e. when they were put into a meaningful context), than when they had no connection.

The subjects considered a sentence as a whole. If a sentence was not grasped as a whole, then scarcely any of its constituent words were. If the sentence as a whole was grasped, then the constituent words appeared very distinct.

These results establish at a very early stage that letters and words are more easily recognisable when placed in a meaningful context. This obviously has immense implications when designing an automatic recognition system intended to emulate human performance. Importantly, the results also demonstrate context operating at two levels - the word level and the sentence level.

Interest in human performance in reading became an area of great interest when the possibility of machine reading of cursive script began to surface around the late 50's.

Human-like performance was an obvious goal for a reading machine, so this performance had to be evaluated in some way.

Neisser & Weene, (1960), carried out experiments in which subjects were asked to identify isolated hand-printed characters. Recognition accuracy was found to be far from perfect (around 95% correct *character* recognition).

The implication drawn from this is that humans struggle with character recognition when the characters are viewed outside a meaningful context.

This contention was backed up by a study described in Miller & Isard, (1963). They found that human subjects found it far easier to read grammatically well-formed language.

Morton (1969) presents a model of human word recognition (updated in Morton, 1979) - the logogen model. A logogen (from the Greek, *logos*, meaning 'word' and *genus*, meaning 'birth'), is a device which accepts information relevant to a particular word response. Each word recognised has a logogen associated with it (with each sense of the same word having an individual logogen).

When more than a threshold amount of information has accrued in any logogen, the response associated with that logogen becomes available for output.

Experimental evidence suggested that the recognition of a word is greatly facilitated by the prior presentation of a context (context lowers the threshold amount needed to be breached for the word to be recognised).

Morton's modified model (Morton, 1979) can be diagrammatically represented as in **Fig. 2.1**.

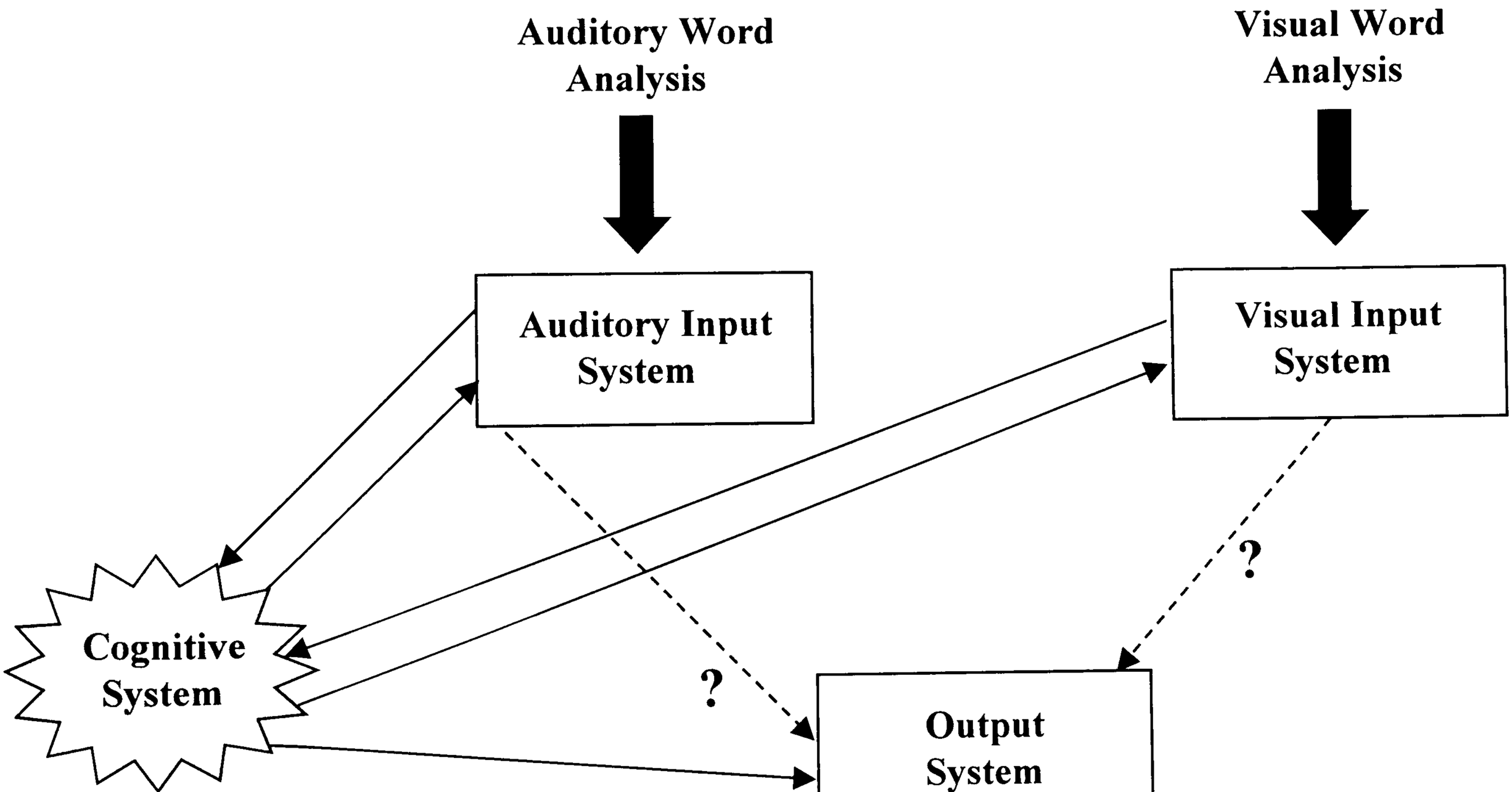

Fig. 2.1 - Morton's Logogen Model

The cognitive system produces 'semantic' information (i.e. knowledge of the world) which interacts positively in the logogen system with the sensory information derived from the auditory and visual stimuli.

It is not certain whether the input systems are directly connected to the output system, or whether output arrives via the cognitive system, but this isn't especially important.

Meyer & Schvaneveldt, (1971), carried out a number of experiments which produced interesting results regarding the effect of semantic association on word recognition. Subjects were required to respond whether two sets of strings were actual words.

Response times were far more rapid when the two words were semantically related (e.g. **bread** and **butter**). This seems to confirm that collocation does play a role in human word perception (see**2 . 4 - Lexis and Collocation**).

Later experiments, (Meyer & Schvaneveldt, 1976), suggest that long-term memory is organised along the lines of a thesaurus, wherein words with related meanings are filed in closer proximity than those with unrelated meanings.

This hypothesis implies that the speed of mental processes involving written and spoken words depends on how closely related the meanings of the words are. Experimental results back this hypothesis.

This 'semantic network' view of the organisationof memory is related to Morton's logogen model. When one logogen 'fires', it is suggested that other logogens in close proximity (i.e. those representing semantically related words), have their own information thresholds lowered, thereby facilitating the subsequent processing of semantically related words.

These results imply that some form of co-occurrence knowledge is used by humans in the recognition of visually degraded text, i.e. if word**A** can be positively identified, then the next word, being difficult to recognise using purely visual information is in all likelihood word **B**.

Schuberth & Eimas, (1977), carried out experiments where subjects were required to identify whether a string of letters was a word or a non-word, with response times being measured.

The strings were presented either in isolation, with a congruous prior semantic context (in the form of an incomplete sentence, e.g. **the dog chewed the ....**), or with an incongruous prior semantic context (four spelt-out digits, e.g. **SIX ONE EIGHT FOUR ....**).

Words presented with the congruous context were recognised more rapidly than when presented in isolation. Words presented with the incongruous context took longer to recognise than those presented in isolation. These results are consistent with Morton's logogen model.

A congruous context facilitates recognition of the correct word, while an incongruous context may actually hamper recognition (the effect of context is so strong, that it can lead us to expect a completely different word to the one presented).

Morton, (1969), and Meyer & Schvaneveldt, (1976), discovered that fewer stimulus features were needed for the recognition of a 'contextually primed' word than for an unprimed word.

Estes, (1977), and Ehrlich & Rayner, (1981), carried out experiments on eye fixations during reading. They postulated that high levels of contextual constraint may influence readers to skip the fixation of highly constrained words, and influence readers to be less sensitive to visual features encountered in central vision.

Fixation times on highly constrained words were indeed found to be shorter than average.

To try and prove the second hypothesis, misspelt words were placed in the highly constraining sentences (with a substitution at one letter position). In many cases the misspelling wasn't noticed. This supports the hypothesis that high levels of contextual constraint cause readers to be less sensitive to purely visual stimuli.

Similar studies described in Carpenter & Just, (1983), and Just & Carpenter, (1987), backed up this hypothesis. They found that syntactic knowledge also provided strong prompting for subjects during reading.

Again we are presented with evidence that context plays an important part in reading.

Haber & Haber, (1981), carried out experiments regarding word shapes. It was found that word shape information, when combined with contextual information (the syntactic and semantic structure of the passage containing the word), was enough to specify a unique word, or at least a very limited set of possibilities in 95% of cases. Revealingly, word shapes in isolation only specified unique words in 25% of cases.

Koriat et al, (1991), address the missing-letter effect, whereby letters are often missed in processing highly common words like **the** or **and**.

The traditional view of this phenomenon is based on the 'unitisation model', (Healy, 1976). This model postulates a hierarchy of processing levels, and assumes that a reader processes text at these levels in parallel.

If a particular unit is highly familiar at a given level, then its processing is facilitated by allowing access to representations at higher levels. So highly familiar words are encoded more easily than rare words because they are processed at the whole-word level, rather than letter-by-letter.

It is also postulated that once a unit is identified at a given level, subjects proceed to the next segment of text without completing processing at lower levels (e.g. the letter level). This implies that itis harder to detect a target letter in a familiar word than in a rarer word, as the familiar word would cause faster access to the whole-word representation.

So according to the unitisation model, word frequency effects are the sole cause of the missing-letter effect.

Koriat et al. question this. Their experiments do support the idea that word frequency is indeed a factor in the missing-letter effect, but also that word function plays a part.

The conclusion is that syntactic knowledge (the function and not merely the frequency of words) is a major cue in reading. This is a convincing argument, particularly as Healy's work did not take account of factors other than word frequency.

A number of recent studies have evaluated machine performance in text recognition in relation to human performance.

Dimauro et al., (1991), investigate the limits of automatic recognition systems which have a training phase based on human knowledge, (one such system is reported in Nadal & Suen, 1993).

An experiment was carried out to investigate the behaviour of human subjects in recognising confusing hand-written numerals.

It was found that there was very little generality in the way that the subjects went about classifying the numerals. Each subject interpreted the patterns according to his or her own experience.

It was stated that human subjects are not particularly suitable as a source of knowledge in supervised machine learning as a result of this personal bias. However, it is debatable whether this unsuitability extends to recognition of items presented in a meaningful context as opposed to in isolation.

Suen et al., (1992), compared machine performance in individual character recognition with human performance.

The characters were split into parts, and it was found that the machine algorithm produced more precise, complete and efficient methods than humans, with high recognition rates, when presented with only part of a character.

However, human-like use of context is a goal to be strived for in automatic text recognition, when whole stretches of textare required to be recognised.

Bellaby & Evett, (1994), and Rose, Evett & Lee, (1994), also report experiments where human performance is compared to machine performance, but this time in the recognition of whole words.

Even when the machine recognisers use limited contextual constraints, they are unable to match human performance in recognising words given in isolation (word-level context, i.e. knowledge about how letters combine, is being exploited by the human subjects).

It is concluded in these studies that human readers make far more comprehensive use of context than a machine, and that automatic recognisers must try to emulate this.

A review of human use of context in reading can be found in Henderson, (1982), going in particular detail into Morton's view of word perception.

Assuming that the aim of automatic reading systems is to achieve human-like performance, it seems clear that the exploitation of context at various levels of processing is a necessary step to achieving this performance.

The next section considers varying views of how various forms of linguistic knowledge interact, or otherwise, in humans.

2 . 3 . 2 - The Interaction of Human Knowledge Sources

It has been repeatedly shown that humans far outperform machines in reading texts due to the use of knowledge other than that provided purely by visual stimuli (i.e. by using linguistic knowledge).

If machines are to match human performance, then it seems they must also utilise linguistic knowledge. How do humans organise and use linguistic knowledge sources?

Marslen-Wilson, (1975), proposes an 'interactive parallel model' of sentence perception. Up until this point, studies had suggested a 'staggered serial' model, wherein the input to any higher level of analysis consists of the outcome of analysis conducted at a level immediately below.

Marslen-Wilson suggests four levels of processing (in ascending order of complexity) :

- phonetic (or presumably orthographic in reading)
- lexical
- syntactic
- semantic

Marslen-Wilson's experiments suggested that the listener (or reader) not only analysed each word phonetically (morphologically) and lexically as he or she heard (read) it, but also simultaneously extracted its syntactic and semantic implications (i.e. processing is conducted at several levels simultaneously, with levels interacting directly with one another).

In particular, it is suggested that the syntactic and semantic representation constructed by the user, constrains and guides the phonetic and lexical analysis of subsequent items.

These propositions are based on experiments wherein the response time of a phonetically-based decision (identifying rhyming words) was roughly the same as that of a semantically-based decision (deciding whether words belong to a similar semantic class).

This would appear to refute the hypothesis of serial processing, wherein semantic processing is invoked some time after phonetic processing.

Forster, (1979), disputes the parallel, interactive model of Marslen-Wilson, presenting a model consisting of a lexical processor, a syntactic processor and a semantic (or 'message') processor all operating autonomously.

Each processor accepts input only from the next lowest processor, and from no other source. No processor has any information at all about the operation of any higher level processor.

Rayner et al., (1983), review this debate about the organisation and interaction of linguistic knowledge.

They look at the problem in computational terms, and suggest that there are potential computational savings to be made through early integration across levels of structure. For instance, they suggest that the use of semantic information may reduce the amount of syntactic processing required.

The actual act of integration may well have a computational cost, but savings would be made in many cases.

This suggests that an interactive model (along the lines of Marslen-Wilson's) would be preferable for automatic text recognition.

However, it is noted that in some cases, integration would be of no help at all, and may add extra computational cost.

For example, it is pointless for a morphological processor to inform the syntactic processor that a word begins with a particular letter, as this has no bearing on the syntactic structure of a text.

Cases such as this suggest that a serial, non-interactive model (along the lines of Forster's) would be preferable.

Rayner and his associates suggest that some hypotheses falling somewhere in between a completely autonomous and a completely interactive model should be investigated.

Experiments investigating an 'in between' model were carried out, with subjects being presented with ambiguous sentences. The amount and type of processing carried out was measured by choosing sentences with particular disambiguation points, at which any previous ambiguity became resolved.

Subjects tended to be 'garden-pathed' - i.e. they read the incorrect meaning until they reached a disambiguation point. The structurally (i.e. syntactically) preferred analysis seemed to be adopted initially, even if this analysis was less plausible on semantic or pragmatic grounds than some alternative analysis.

The results support a hypothesis that there are two largely independent processors operating during sentence comprehension.

One processor is responsible for structural parsing preferences - a syntactic processor responsible for initially computing the structurally preferred analysis of a sentence.

The other processor is responsible for lexical, semantic and pragmatic preferences - a 'thematic' processor that examines the alternative thematic structures of a word, and selects the semantically and pragmatically most plausible one in the context of the sentence.

This hypothesis seems to me to be a plausible compromise between the two extreme views propounded earlier.

The problem with implementing this model in an automatic recognition system is if these processors should interact in any way. For instance, would information garnered from the semantic processor affect the operation of the syntactic processor if it itself fails to affect complete recognition? Intuitively this would be the case in human recognition.

At any event these considerations of process interaction are beyond the scope of this thesis as it only deals with one form of linguistic processing – collocation – and does not feature a syntactic component.

However I believe that the combination of the collocation processor with a syntactic processor, and research into which ways this combination could be configured would be a fruitful line of future research.

2 . 3 . 3 - The Use of Context in Automatic Text Recognition

In referring to 'context' in the title of this section, I exclude the linguistic level of pragmatics, i.e. knowledge pertaining to 'real world' situations and relationships.

In my opinion, 'real world' knowledge has never been satisfactorily incorporated into machine-based applications.

When a text recognition system claims to use context as an aid to recognition, the context generally operates at one of two levels :

- orthographic (i.e. the way in which letters combine to form words is exploited)
- syntactic, and possibly semantic (i.e. the ways in which words combine to form phrases and / or sentences is exploited)

Toussaint, (1978), presents a survey of techniques for using contextual information in pattern recognition. To quote directly :

> '...the effect of context is that some entity Z can have certain properties when Z is viewed in isolation, which change when Z is viewed in some context. Alternatively, Z is seen as one thing in context A and as another in context B.'

He observes that some problems cannot be solved through an ever-increasing depth of analysis, but must be solved by widening the context in which the problem is viewed.

This suggestion is backed up by the findings of Bellaby & Evett, (1994). They carried out an experiment which showed that the ability of a low-level recognition system to identify cursive handwriting can actually decrease as it is trained on new samples of handwriting. As the training set increases, so does the ambiguity present in it. This is due to the inherent ambiguity of cursive handwriting.

Indeed, the performance of recognition systems based on strictly visual techniques (henceforward, low-level systems), seems to be approaching some sort of ceiling, and more and more effort is being aimed at applying higher-level knowledge to the task of recognition.

Toussaint, and also Nagy, (1992), categorise contextual techniques into three groups :

- dictionary look-up methods (top-down approaches)

- probability distribution approximation techniques (bottom-up approaches)

- hybrid methods (combining the above methods)

2 . 3 . 3 . 1 - Dictionary Look-up Methods (the top-down approach)

One of the earliest recognition systems to use this approach was proposed by Bledsoe & Browning, (1959).

Recognition was based on whole words. A vocabulary, partitioned according to word length, was defined. The word pattern in the vocabulary with the closest resemblance to the input pattern was chosen as the designated word.

This basic method was to be used many times in future systems. The main drawback with techniques like this at the time was the storage requirements of the dictionary. This is no longer a serious problem with modern machines with their huge filestore capacities. Efficient look-up techniques will still of course be required for large dictionaries in real-time operation.

One such look-up technique is put forward in Bozinovic & Srihari, (1984), propose a system which produces a set of potential words at each word position. A lexicon lookup procedure is used to 'weed out' invalid letter sequences.

The lexicon is organised as a trie (a three-way tree), which is found to be an efficient representation. This technique is described in detail in Wells et al., (1990).

Each node in the trie represents a letter. If a letter represents the end of a word in the lexicon, then this is flagged.

A lexicon search is successful if a candidate string is found in the trie (by traversal of the nodes), and the last letter in the string is flagged as an end-of-word.

Hull, (1987), describes a two stage recognition process. First, a gross visual description of the word is used to suggest a set of hypotheses about its identity. This hypothesis set is called a 'neighbourhood', which is derived from a dictionary. The words in the neighbourhood have the same feature description as the input word.

A lexicon lookup routine ensures that the hypothesis set contains only valid words. The second stage of the process narrows the neighbourhood down using syntactic (and in theory at least, semantic) knowledge.

Keenan & Evett, (1989), describe the development of techniques for the efficient use of large lexicons in a recognition application.

A structure is created which represents the vocabulary of the system, and also contains various types of information about thewords which can be used to facilitate recognition.

An existing machine-readable dictionary (MRD) was used as the basic database for syntactic and semantic information.

It was suggested that a morphologically-based access system was required, to keep down storage costs, and to reflect the morphological properties of user input.

A list of non-inflected words was obtained from the MRD, each word being assigned a unique, numerical root-morpheme index.

Inflexions of these basic words were generated, with the derivations inheriting the root index (e.g. **funny**, **funnier** and **funniest**, all shared the same index).

When candidate letter strings were checked against the lexicon (stored again as a trie), the indices for those strings which constituted acceptable words were obtained. The entry for every such word could now easily be accessed by its index, and brought into memory.

Hunnicutt, (1989), uses a lexicon as part of a word prediction aid, wherein a likely candidate word is predicted from the initial letter(s) of a word. The candidate words are chosen either from a variable-size, frequency-ranked lexicon, or from a list of words previously used, thereby ensuring that both word frequency and recency effects are taken into account.

The lexicon is partitioned into subject areas (e.g. 'The World in which we live'). A semantic overview of a text can be gained by counting the number of words it contains in each category.

A number of constraints can be applied to a recognition system. Paquet & Lecourtier, (1993) suggest the size of the system's lexicon as one of them.

Constraining the size of the lexicon may allow other constraints (e.g. on writing style) to be eased or removed completely.

Paquet and Lecourtier's application deals with handwritten amounts on French bank cheques. The lexicon of words required to be recognised is restricted to around forty. This means that fewer features are required to uniquely specify a word. A highly restricted lexicon is coupled in this particular application to a highly restricted syntax, which can also be used to constrain recognition.

Dimov, (1994), defines the output verification task using a lexicon as basically the task of approximate string matching between strings belonging to two sets (i.e. the output from the low-level recogniser and the lexicon).

It is suggested that these sets can be enriched with a structure of a probabilistic nature.

The approach proposed by Dimov was developed for the purposes of automatic correction of the most probable mistyping errors that occur during natural language production. The most probable character substitutions in a word are assumed to be independent events, each with a particular probability.

The lexicon contains **lexemes**. These are derivations extracted from root words by a process of symbol deletion. Words input to the lexicon lookup process are considered to be trivial lexemes, which are generated from the current input word.

These lexemes are checked for a match in the dictionary. Each 'hit' represents an error occurring somewhere in the word, which can now be corrected.

Breuel, (1994), describes a dictionary-based method devised for a specific application. This application involves the recognition of handwritten entries on a census form.

It was found that particular phrases recur frequently. This fact was exploited by using a phrase dictionary with each entry having an occurrence probability associated with it.

A word dictionary (also with probabilities) was used to pick up what was missed by the phrase-based model.

The problem with dictionary look-up techniques is the potentially prohibitive time taken to look up a word in a large lexicon, even with the high-performance machines of the 1990s.

The combination of probabilistic methods with lexicon look-up techniques may provide a solution to this problem by cutting down to a manageable size the part of the lexicon needing to be searched.

2 . 3 . 3 . 2 - Probability Distribution Approximation Methods (the bottom-up approach)

These techniques exploit some of the probabilistic properties of language. Certain letter (and indeed, word) combinations are more likely to occur in a given language than others.

These probabilities can either be represented as n-gram probabilities (bigrams for two-letter combinations, trigrams for three-letter combinations etc.), or the transition probabilities between letters (or words) can be represented as a Markovian process.

- n-gram statistics

An early system based solely on the statistical properties of language is proposed by Casey & Nagy, (1968).

Specifically, letter-pair (i.e. bigram) frequencies were exploited. Input characters were partitioned into groups of similar patterns using some similarity measure, with each class being assigned a label.

Next, a matrix representing the bigram frequencies of these labels was compared to a frequency matrix obtained from a large sample of English text, and a specific letter assigned to each label based upon this comparison.

The main drawback of methods like this, and indeed any method relying solely on language usage probabilities is that rare but valid letter pairs will often be overlooked.

Riseman & Ehrich, (1971), introduced the concept of binary digrams. A low-level recogniser outputs a set of alternative letters (without confidences) that contains the correct letter a high percentage of the time.

A simple dictionary of words that are to be recognised by the system is required. Binary digrams, in the form of matrices are defined, giving the 'syntax' of the dictionary.

If there is a non-zero probability that a letter $\mathbf{a_k}$ occurs in the ***i***th position of a word, and that a letter, $\mathbf{a_l}$ occurs in the ***j***th position of the same word, then this information can be recorded by placing a **1** in the **(*k*,*l*)**th position of a 26 x 26 matrix called $\mathbf{d_{ij}}$.

In other words, the probability of occurrence of each letter has been quantised into a **0** or a **1**. Information is lost in this method - a letter pair is represented as being either valid or invalid, with no probability of occurrence being given other than this. However, the size of the lexicon and therefore the time taken to search it is greatly reduced..

Another advantage of this method is that the position of a letter pair within a word is taken into account, and non-contiguous letter-pairs can easily be analysed.

The more digrams that are defined, the more strings can be rejected without resorting to dictionary lookup.

This binary digram method is further developed in Ehrich & Koehler, (1975), and binary trigrams are exploited in Hanson *et al*.. (1976).

Whitrow & Higgins, (1987), describe a post-processing system which utilises **n**-grams. The system accepts as input a directed graph (or candidate lattice) where each node is a possible letter, and the links between them represent the arcs joining the letters in the cursive script (see**Chapter 3** for a more detailed description of such directed graphs, or lattices). Different pathways through the graph represent different potential words.

A list of allowed **n**-grams must be stored. This list is used to reduce the graph by removing arcs which span impermissible **n**-grams.

The choice of the value of **n** depends on a number of criteria. If **n** is small, then a large number of paths are allowable, and search times are large. If **n** is large, then longer paths have to be searched, with the consequent storage overhead. A workable compromise was found in the use of quadgrams.

Koh et al, (1994), propose a hierarchy which uses letter **n**-gram statistics followed by word **n**-gram frequencies. Even modest bigrams for words become very wasteful of memory. Instead, word class (or part of speech) **n**-grams are used.

This is in effect a Markovian process. The exploitation of Markov models in text recognition is discussed below.

- Markov Models in Text Recognition

A Markov Model is a collection of states connected by transitions. Each transition has associated with it a probability specifying how likely the transition is to take place, and an output probability density function, which defines the probability of emitting a symbol from some finite set, given that that transition has taken place.

If we can't tell from the output which state the model is currently in, then the model is called a Hidden Markov Model (HMM).

The states can represent letters or words. If the states represent letters, then the model gives the likelihood of particular letter sequences occurring.

If the states represent words, then the model gives the likelihood of particular word sequences occurring.

The Viterbi algorithm, (Forney Jr., 1973), is a recursive, optimal solution to the problem of estimating the state sequence of a finite-state Markov process.

Forney Jr. states that if English (or any other language) is treated as a discretely-timed Markov process, then the algorithm can be used in text recognition. See **Fig. 2.2** for a diagrammatic representation of how the Viterbi algorithm would be used in a text recognition system.

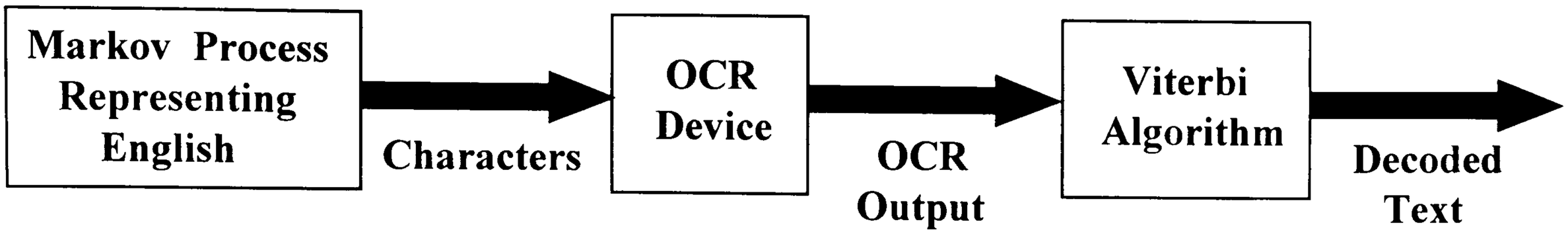

Fig. 2.2 - The Viterbi Algorithm

Kuhn, (1988), modifies the Markovian approach to incorporate word recency into a speech recognition system. Higher probabilities are assigned to recently used words.

In Kuhn's system, a low-level recogniser produces a number of candidate words. Each candidate has two probabilities associated with it :

- a probability based on its resemblance to the input

- a probability based on the linguistic plausibility of that word occurring immediately after previously recognised words

Multiplying these two values together gives an overall probability for each candidate word.

The model is based on trigram probabilities, and incorporates a 'cache' component used to track short-term fluctuations in word frequency.

Results obtained when this modified model was compared to a traditional Markov model seem to confirm the hypothesis that recently used words have a higher probability of occurrence than the pure trigram model would predict.

Markov models have been used to represent the syntax of a language (Hull, 1992; Hanlon & Boyle, 1992; Freedman, 1993).

The states of the model represent word classes (or parts of speech) in these cases.

Hull, (1992), presents a system which accepts a number ofalternative words as input. The syntactic tags for each alternative are input to a modified Viterbi algorithm, which determines a number of sequences of syntactic classes that include each word.

An alternative is output (possibly to further levels of processing) only if its syntactic class features in at least one of these sequences.

The performance of the system was measured by comparing the average 'neighbourhood size' per word (i.e. the size of the set of alternatives), before and after application of the model. Up to 80% reduction of the neighbourhood size was achieved in tests.

Hanlon & Boyle, (1992), use a similar approach. A Hidden Markov Model (HMM) is used to model English grammar. A first order model is used - i.e. a word's syntactic class depends only on its predecessor.

Gilloux, (1994), provides a comprehensive survey of the uses of HMMs in handwriting recognition, with a number of specific applications being discussed.

Evett et al, (1992), exploit the probabilistic properties of syntax to weed out grammatically incorrect word sequences from a candidate lattice. A word trigram transition matrix, based on the frequencies of grammatical categories, is used.

- Combining n-grams and the Viterbi algorithm

Hull & Srihari, (1982), present a system which combines the two approaches discussed above.

One component of the system is based on the concept of binary **n**-grams, (**n** = 2, 3 or 4), proposed initially by Riseman & Ehrich, (1971).

A set of binary arrays of 26 elements each is used to represent an abstraction of a dictionary of all allowable words.

The other component uses the Viterbi algorithm to compute the word in the dictionary that most probably corresponds to the observed word. This probability is based on the probabilities of confusion between letters, and the probabilities of co-occurring **n**-grams.

2 . 3 . 3 . 3 - Hybrid Approaches

Dictionary lookup methods have proved very effective in reducing the number of candidates produced by a low-level recogniser.

However, they suffer from potentially slow access times in domains of any practical size.

N-gram methods have proved less effective, introducing undesired complexity for only modest reductions in error rates. However, they incur much less computational cost in general.

A number of studies have attempted to combine these methods, and retain the effectiveness of dictionary look-up methods without the computational costs associated with them.

Vossler & Branston, (1964), used both methods for correcting mistakes in garbled English text.

Each entry in the dictionary is a probability that a letter substitution will occur, i.e. $\mathbf{P_c(L_i \mid L_j)}$ is the probability that the low-level recogniser outputs letter $\mathbf{L_i}$ when the actual letter in the text is $\mathbf{L_j}$. (e.g. $\mathbf{P_c(H \mid A)}$ will be relatively high, due to the similarity between **A** and **H**). Parallel to this is a normal word dictionary, partitioned by word length.

Now, if the recogniser attempts to identify a three letter word, and produces string **XCQ**, it is required to identify a three letter word in the dictionary which produced this output, using the confusion probabilities.

The other component of the system uses letter-pair transition probabilities, obtained by examining a large text.

Results were found to be improved when the two methods were combined, compared with when they were used in isolation.

Shingal & Toussaint, (1979), propose a 'Predictor-Corrector Algorithm', which is a true compromise between the top-down and the bottom-up approaches.

Given an input word **X**, a modified Viterbi algorithm is used to predict word **Z**. A dictionary is now checked for **Z**. If **Z** is in the dictionary, then it is assumed to be the correct word.

Each word in the dictionary has a value associated with it. If **Z** is not in the dictionary, then the 'scores' of the words in some user-defined neighbourhood of **Z**'s 'mate' (the 'closest fit' of **Z**) are calculated. The word with the highest score is assumed to be the designated word.

Tests on this system suggested that the algorithm could achieve the same low error-rate as the dictionary lookup algorithm, at half the computational cost, as the complexity is reduced.

Srihari et al, (1983), propose an algorithm that integrates three knowledge sources :

- channel characteristics, in the form of probabilities that observed letters are corruptions of other letters. A letter confusion probability table is used.

- bottom-up context. In fact, two types of bottom-up information are used :
  - letter shapes which are stored as vectors
  - the probability of a letter occurring when the previous letters are known, in the form of a transition probability table.

- top-down context, in the form of a lexicon of legal words, which is represented as a trie.

Results of tests on this system showed a significant increase in letter correction rate over previous systems that didn't exploit lexical information, while no increase in computational complexity was observed.

This algorithm is proposed as a word hypothesisation component in a system focusing on the use of global contextual knowledge in the text recognition problem.

Lettera et al., (1986), explore the use of a dictionary in conjunction with a statistical handwritten character recogniser. It is suggested that the possibility of obtaining successful results from a text recogniseris enhanced by a priori knowledge about the vocabulary used by the writer. The statistical component of the system produces a hypothesis matrix, which is narrowed down by dictionary lookup.

It is proposed that the problem of optimum search in a hypothesis matrix is an instance of search in a state space, (Nilson, 1971). The A* algorithm is used to give the optimum search.

It is clear from the work studied in this section that systems based purely on lexicon look-up techniques or purely on statistical methods will give problems.

Lexicon look-up techniques require the storage and searching of large collections of words. While the storage of such a collection is no longer a problem, the searching of the whole lexicon can prove time-consuming.

Purely probabilistic methods risk mis-recognising valid but rarely occurring words.

It is suggested that a system exploiting both lexical information and statistical information (in the form of collocations)provides a compromise that may avoid the worst of both worlds.

2 . 3 . 4 - The Use of Linguistic Knowledge in Automatic Text Recognition

Hull, (1994), gives a review of language-level constraints as applied to the task of text recognition. He discusses five types of constraint :

- Graphical Constraints, dependent upon the consistency in writing style of a particular person.

- Vocabulary Constraints, exploiting the commonality between words used by an author on a particular topic, allowing the dictionary of valid words to be reduced.

- Statistical Constraints, based on the predictive ability of words or other grammatical characteristics, the most common method being the use of collocation data.

- Structural-syntactic Constraints, based on the information provided by a full parse of a sentence.

- Structural-Semantic Constraints, based on the commonality of theme in a document - the 'glue' that binds words together.

St.John & McClelland, (1990), propose a model of the sentence comprehension process centred on viewing the process as a form of constraint satisfaction. The surface features of a sentence (its words, their order and their morphology), provide a set of constraints on the sentence's meaning.

The constraints (syntactic and semantic), define which roles are given to the sentence constituents.

Different constraints compete or cooperate to produce an interpretation of a sentence. If real-valued strengths are assigned to constraints, then a parallel distributed model can be used to carry out this competition of values.

A common way of obtaining syntactic and semantic information about language, to be applied to the recognition process, is to analyse existing electronic resources, as described in the next section.

2 . 3 . 4 . 1 - The Use of Existing Electronic Resources

Existing electronic resources are invaluable sources of knowledge about natural language.

Electronic corpora are of vital importance when looking for information about the actual use of a language, (Sinclair, 1982; Garside, Leech & Sampson, 1987; Aijmer & Altenberg, 1991).

The British National Corpus and the Susanne Corpus will be described in greater detail later in this thesis.

Machine-Readable Dictionaries (henceforward, MRDs), are also valuable resources, providing information about the semantic domains of words, (Alshawi, 1988), and helping to disambiguate the different word senses of homographs, (Amsler, 1981).

Rose & Evett, (1992), used the technique of 'definitional overlap', applied to the definitions of an MRD, as an aid to text recognition.

It was postulated that the dictionary definitions of semantically related words will have words in common, i.e. there will be definitional overlap.

Two types of overlap were defined : strong overlap, where one or both of the words appears in the definition of the other, and weak overlap, where there are other words common to both definitions.

The definitions of pairs of content words in a sentence were compared, with each pair being assigned a score according to the number of strong and weak overlaps occurring (strong overlaps being worth considerably more than weak overlaps).

The correct word in each word position was assigned the highest score in around 70% of cases using this technique.

Existing resources can also be used to define a restricted domain for some discourse, thereby restricting the possibilities at various word positions. Grishman & Kittredge, (1986), contains a number of studies relating to sublanguages and restricted domains.

Also, Arnold, (1990), discusses sublanguage analysis in the field of machine translation.

2.3.4.2 - The Application of Syntactic Knowledge to the Recognition Task

It has already been noted that syntactic knowledge has been found to play an important role in the human perception of language.

Keenan & Evett, (1994), use both a generative (rule-based) parsing system and the probabilistic techniques described in Evett et al, (1992).

Both approaches have advantages and disadvantages, but the generative approach was found in this case to be computationally too complex to be practically feasible.

In conclusion to this section on the use of linguistic knowledge in automatic text recognition, many studies over many years show that humans exploit far more than purely visual information when reading.

It is suggested that the human-like exploitation of higher levels of linguistic knowledge may improve the accuracy of automatic recognition systems.

Systems exploiting such knowledge in various ways have demonstrated higher recognition accuracy than comparable systems based on visual information.

A matter of contention however is which knowledge sources should be used, and how they should be combined. It is suggested that lexical knowledge in the form of a dictionary, and knowledge of linguistic *usage* in the form of collocation statistics could prove effective.

It is also suggested that the future addition of a syntactic component, and research into how this may interact with the other knowledge sources may prove interesting.

2 . 4 - Lexis and Collocation

2 . 4 . 1 - Lexis as a Linguistic Level

J.R.Firth is to all intents and purposes the father of modern lexical studies. In an early paper, (Firth, 1935), he discusses the concept of the central, or seminal, meaning of a word from which all subsequent meanings are derived, and stresses the importance of lexical studies in linguistics.

In a later paper, focusing on the 'meaning' of words, (Firth, 1951), he proposes the concept of meaning by 'collocation' (using the word collocation as a technical term for the first time). He postulates that one of the meanings of a word is its habitual collocation with other words, and in a well known and oft-quoted phrase, states that :

> 'One of the meanings of "night" is its collocability with "dark", and of "dark", of course, collocation with "night".'

In other words, 'You shall know a word by the company it keeps', (Firth, 1957b).

Collocational meaning is contrasted with contextual meaning, (the 'conceptual' or 'idea' approach to the meaning of words), which relates a sentence to some context of situation in the context of culture. The contrast is further expounded later (Firth, 1957b) :

> 'Collocations of a given word are statements of the habitual or customary places of that word in collocational order, but not in any other contextual order, *and emphatically not in any grammatical order*.' [My italics].

Thus, Firth proposes collocation as a separate linguistic level (the 'collocational level'), distinct from syntax in particular.

This idea of a separate, lexical linguistic level is expanded upon by proponents of systemic linguistics, (see, e.g. Halliday, 1961). The 'collocational level' put forward by Firth is renamed **lexis**, having collocations as its patterns.

This idea of lexis as a level of linguistic knowledge forms the theoretical basis of this thesis.

The idea put forward by Firth, that lexical patterns are different in kind from grammatical patterns, has often been questioned. Berry, (1977), summarises the three main viewpoints :

1) Lexis is merely what is left over from grammar.

> Language should be described to as great an extent as possible in terms of grammar, with lexis being resorted to only when absolutely necessary.
> Proponents of this view expect lexis to be eventually subsumed under grammar.
> It is necessary at present to treat lexis and grammar as separate levels, but this is due to the inexperience of linguists rather than the nature of language.

2) Lexis and grammar are by their very nature different.
Lexis will never be subsumed under grammar - the two levels are distinct, and always will be.

3) Between the extremes expressed in 1) and 2) is the view that lexis and grammar differ only in degree - there is no sharp division between the two levels, they are merely at opposite ends of a scale of delicacy.

Halliday, (1966) adheres to the third view. He states that the major preoccupation of grammatical theory is the extension of grammatical description to a greater degree of delicacy, i.e. reducing the very large classes of formal items into very small sub-classes, ideally one-member classes.

This is not always possible using grammar alone, and this is where lexis comes in. Halliday suggests a scale on which items can be ranged, from 'most grammatical' to 'most lexical'. (See also Hasan, 1987).

The most grammatical item is one which is optimally specifiable grammatically, i.e. is reducible to a one-member class by the minimum number of steps in delicacy.

Such an item is not necessarily the 'least lexical' - there may be a collocational environment in which its probability of occurrence is significantly higher than its unconditional probability, (see **2.4.2 – Collocation**, below).

I find this a very convincing argument. Some words are best dealt with using syntax. These words (I shall call them 'grammar words') do not strongly predict their environment in terms of individual words, but rather in terms of grammatical classes. To give an example, the word **the** will in many cases be followed in a sentence by a noun. This is relatively easy to predict. Specific identity of this noun is virtually impossible to predict (e.g.**cat** and **dog** will have a very similar probability of occurring after **the** in a language).

Other words (I shall call them 'lexical words') strongly predict their environment in terms of specific lexical items.E.g. **cat** has a high probability of being preceded by the word **the**.

The ability of lexical words to predict their specific environment is the key to the recognition system described in this thesis. The ways in which grammar words predict their environment will also prove to be of interest however.

These phenomena will be studied in greater detail later in the thesis.

2 . 4 . 2 - Collocation

Much of the groundwork for the practical study of collocations was laid by J.McH.Sinclair (Sinclair, 1966). At the time of this paper however, the resources were not available to gather meaningful information about collocations.

Sinclair suggests that by studying the tendencies of items to collocate with each other, we can discover facts about language that cannot be discovered by grammatical analysis.

Such tendencies cannot be expressed in terms of small sets of choices (as grammatical patterns can).

A particular lexical item is not chosen *rather than* another - they do not contrast in the same sense that grammatical classes contrast.

It is easy to give examples of lexical patterns, but very difficult to prove the assertions made about them. Many statements about collocations are intuitively correct to a native speaker of a language, but very hard to generalise.

Berry, (1977), explains that grammatical items obviously share certain properties, (e.g. **Neil**, **they** and **she** can all be subjects of a verb), so a general label can be applied to them all.

On the other hand, an item of a collocation is a particular, unique 'thing'. Statements made about collocations are less general than those about structures.

According to Sinclair, (1966), the major problem in making definitive statements about lexis is the circularity in the definition of its basic unit of description - the lexical item :

> 'a formal item (at least one morpheme long) whose pattern of occurrence can be described in terms of a uniquely ordered series of other lexical items occurring in its environment.'

I.e. a lexical item is a unit of language representing a particular area of meaning, which has a unique pattern of co-occurrence with other lexical items. Orthographic words are the most convenient form to study, but a lexical item can be a morpheme, a homograph (one particular meaning of a word), or a pair or group of words.

Simple collocation is the main structural criterion of lexis. One item is said to collocate with another item if the probability of it occurring in that item's environment is greater than its individual probability of occurrence would suggest.

Sinclair (1966, and Jones & Sinclair, 1974), defines a number of terms required in the study of collocation :

node - the lexical item currently under examination (normally an orthographic word).

collocate - any item which appears with the node within a specified environment

span - specifies this environment. This is the amount of text within which collocation of items is said to occur. Span positions of collocates are numbered according to their distance from the node (**N**).

E.g. in the sentence :

the cat sat on the mat

if **sat** is the node, then **cat** is a collocate of **sat** at span position **N-1**, and **mat** is a collocate of **sat** at span position **N+3**.

When studying a text, Sinclair suggests that we measure the way in which an item predicts the occurrence of others, and also the way in whichit is predicted by others.

The second of these measurements is chosen as the statement of the lexical meaning of an item - its **cluster**.

The formal meaning of an item **A** is that it has a strong tendency to occur near items **B**, **C**, **D**; less strong with items **E**, **F**; slight with items **G**,**H**, **I**, and none at all with any other item. This information is tabulated in the cluster.

At what point can a collocation be said to be significant? Ultimately, this decision is arbitrary, but we can improve on purely intuitive grounds.

We can calculate the probability that an item will occur at a particular place in a text :

$$\frac{\textbf{Total number of occurrences of a particular item (= f)}}{\textbf{Total number of items occurring in the text (= p)}}$$

If a particular node occurs **n** times, and the span setting multiplied by two (for item places on each side of the node), is **s**, then the probability of our item collocating with this node is :

$$\frac{\mathbf{nsf}}{\mathbf{p}}$$

Statistical tests can assess the significance of any discrepancy between the predicted and the actual figures, giving either a positive correlation (i.e. the collocate attracts the node to itself), a negative correlation (i.e. the collocate repels the node), or an absence of collocation (i.e. the items are neutral in the particular text under examination). The level of significance must be set by the observer.

One way of setting the level of significance which has been adopted in a number of studies, (Berry-Rogghe, 1973; Lancashire, 1987), is the use of the **z-score**.

In statistics the z-score is a way of ascertaining how many standard deviations from the mean a score lies. In terms of word co-occurrence, the mean can be defined as the number of times two words would be expected to co-occur in a text within a particular word span given the size of the text and the frequencies of the two words within that text.

If the two words have a strong tendency to co-occur then the z-score representing the probability of this co-occurrence will be high, i.e. the probability of co-occurrence will be a number of standard deviations above the expected probability of co-occurrence.

To calculate the z-score for the co-occurrence of two words within a text, given the following (from Berry-Rogghe, 1973) :

Z = total number of words in the text,

A = a particular node, occurring FN times,

B = a collocate of A, occurring FC times,

K = number of co-occurrences of B and A,

S = span size,

the following formulae can be defined :

$$\mathbf{P} = \frac{\mathbf{FC}}{\mathbf{(Z - FN)}}$$ - the probability of **B** occurring at any place where **A** does not occur

$$\mathbf{E} = \mathbf{P * FN * S}$$ - the expected number of co-occurrences

$$\mathbf{z\text{-}score} = \frac{\mathbf{(K\text{-}E)}}{\mathbf{sqrt\ (E * (1 - p))}}$$ - the Standard Deviation

Lancashire, (1987), found that collocates with a positivez-score above 1.499 appeared to be semantically attracted to the node. Collocates with a negative z-score appeared to be actively repelled.

Berry-Rogghe, (1973), suggested a significance limit of a z-score greater than 2.576. This gives a significance level of 5%, i.e. the 5% of collocations deviating most from the mean are deemed significant.

Choosing a significance level is an arbitrary choice based on the intuition of the chooser (i.e. which z-score gives intuitively the best results).

2 . 4 . 3 - A Study of Collocation

A major study of collocation was carried out in the 1970's, (Jones & Sinclair, 1974), in which the collocational behaviour of certain orthographic words was studied.

A text of 135,000 spoken words was the chief source of information used, along with a 12,000 word text of written English.

In this study, a span of **4** was found to give optimal results (i.e. four word positions on either side of the node were considered).

It is worth noting here that a study of collocations in speech would probably require the study of a smaller word span due to the reduced structural complexity found in speech in relation to writing (see section **2.1**). This would mean that word co-occurrence would tend to occur over shorter stretches of text.

Indeed, the language models used by the main speech recognition systems commercially available today are based on trigrams (spans of three words).

Sinclair's study makes a distinction between 'meaning' (lexical) words, and 'function' (grammatical) words.

In short texts, grammatical patterns appear to predominate, as they are marked by the occurrence of highly frequent words and morphemes. A very large text is required to make meaningful statements about lexical patterns, as a word with a full lexical meaning will not be required as often as one with a primarily grammatical function.

Halliday, (1961), suggested that items which are mainly grammatical will be collocationally neutral. I.e. they :

> 'are unlikely to occur in any collocational environment
> with a probability significantly different from their
> overall unconditional probabilities...'

An examination was made in Jones & Sinclair's study of the most frequently used words in the two texts :

the, a, and, of, in, to, I, you, it

These are all 'grammatical' words, but they were found not to be collocationally neutral. They attracted a large number of significant collocates in both texts.

However, their collocational patterns showed a distinctly grammatical influence. It is relatively easy to guess which word classes will occur around them (e.g. **the** attracts nouns and adjectives at the **N+1** position, and prepositions at the **N-1** position), but it is not easy to guess which particular words will co-occur.

A particular study was made of the way the word **the** accumulates significant collocates in the written text.

There were found to be important differences in the way that collocates achieve significance. A small number of words accumulated co-occurrences with the node over a long stretch of text, their frequency increasing with that of the node (these were mainly grammatical words, that were more evenly distributed over the text than the purely lexical items).

The remainder of the significant collocates of **the**, concentrated all their occurrences into a narrow band of text. These are text-dependent items, (e.g. in a section of text on shipping, the word **ship** will have a significant collocation with **the**).

However most of these lexical items remained significant collocates of **the** even when the entire text was taken into account.

The degree of prediction exercised by each word is very different. Given the word **the**, the likelihood that it will be followed by, say, **ship** is very small, but the likelihood of **ship** being preceded by **the** is much higher.

Sinclair later (in Sinclair, 1987 and 1991), goes on to define *upward* and *downward* collocation. In the example above, when **the** is the node and **ship** is the collocate, this is defined as *downward* collocation (i.e. the collocation of a frequent word with a less frequent word). The reverse situation is defined as *upward* collocation.

The difference between the two types of collocates (those that continue to accumulate co-occurrences throughout the text, and those that are concentrated in a small area) can be illustrated by the probability measure :

$$\frac{\textbf{Number of intercollocations}}{\textbf{Number of node occurrences}}$$

The first type's probability of co-occurring with the node is fairly stable throughout the text (e.g. **of** with **the**).

For the second type, while it will remain a significant collocate over the whole text (e.g. **ship**), the probability that it will actually co-occur with the node will diminish steadily as the node frequency increases.

The conclusion drawn is that **the** has a limited ability to predict its own environment. In a large text, most of the words which collocate significantly with it will be predicted with a low probability. Those that are predicted with a high degree of stability will generally also be 'grammar' words.

The power of high frequency grammatical words to predict their own environment is limited to the ability to attract particular word classes at particular span positions.

The collocational behaviour of twenty selected lexical words was studied next. A number of aspects were considered :

Collocation with grammatical items - the appearance of grammatical items with lexical items was generally associated with their own word class, and that of the node. However, the choice of particular words within the word class was not always explicable in terms of grammar. This may be due to some lexical influence, or it may be purely text-dependent.

Collocation between lexical items - nouns, verbs and adjectives, mainly. Association between lexical items was still subject to grammatical influence, particularly in the juxtaposition of different word classes (e.g. adjectives were consistently preceded by adverbs and followed by nouns as significant collocates).
However, within the grammatical organisation, considerable lexical selection was found to be taking place. A number of 'lexical sets' started to emerge. For example, it was noticed that words connected with the concept of time had a tendency to occur in the same environment.

Position-Dependent and Position-Free Collocations - in general, significant collocation was found to show a considerable amount of position dependence. Significant collocation was found to be most frequent in span positions adjacent to the node, and very little occurred at the two extremes of the chosen span.

Some of the collocates were significant at a particular span position, but not overall. These are called position-dependent collocates.

A smaller number of the collocates were significant overall, but not at any particular span position. These are called position-free collocates.

> Freedom from a fixed position is said to be a characteristic of lexical rather than grammatical associations, e.g. (from Berry, 1977), 'fair play, please John' and 'play fair, please John', use different grammatical structures but are lexically the same.

A number of interesting results regarding the different ways that 'lexical' words and 'grammar' words collocate will be reported in the experimental section of this thesis.

2 . 4 . 4 - Some Other Practical Studies of Collocation

A number of other studies provide interesting background information for any research into collocation.

Roos, (1976), looks at collocation of terms of learning a second language. He states:

> 'Particular difficulties arise if the learner is confronted with a choice of 'synonymous' lexical items which vary in *usage*' [My italics]

This is clearly a problem for the learner of a second language. Roos suggests that for collocations to be regarded as significant, they should be grammatically permissible as well as statistically likely to occur.

Mackin, (1978), describes attempts to compile a dictionary of idiomatic English. This study confirms the intuitive feasibility of the idea that particular words and phrases 'belong together' for reasons other than grammaticalness.

A list of incomplete phrases were given to subjects. The missing word or words (not just word class) were easily recognisable in the vast majority of cases, once more confirming the use of word-level context by humans during reading.

Plate, (1988), reports attempts to extract co-occurrence statistics from the Longmans Dictionary of Current English.

He suggests that the frequency of co-occurrence of a pair of words provides a reasonable measure of the strength of the semantic relationship between them.

The text units considered were sense definitions in the dictionary, with the co-occurrence span covering the whole definition.

To validate the co-occurrence statistics of the words which were considered to be significant collocations, humans were asked to rate the relatedness of the pairs of words. The correlation between the mean of the human judgements and the conditional probability of co-occurrence was found to be very high, implying that the conditional probability of co-occurrence is strongly related to human judgements of semantic relatedness.

Plate suggests that the data can be used to form lexical sets, i.e. sets of words that are semantically related to a particular word.

Kjellmer, (1991), borrows a phrase from Shakespeare – 'a mint of phrases' - to describe the collection of set phrases at a native speaker's disposal. A large part of our mental lexicon consists of combinations of words that customarily co-occur.

A typology for this 'mint of phrases' is attempted, including the following categories :

Fossilised Phrases - where the occurrence of one word almost always predicts the occurrence of another. This category is further broken down :

right-and-left predictive - both words predict each other equally, e.g. 'Cocker Spaniel'.

right-predictive - the first word strongly predicts the second, but not vice versa, e.g. 'brussels sprouts'.

left-predictive - the second word strongly predicts the first, but not vice versa, e.g. 'arms akimbo'.

Semi-Fossilised Phrases - where one word predicts a very limited number of words. Possible variants are lexically selected. Idioms generally belong to this class. This is also broken down further :

right-predictive - e.g. 'Achilles heel' / 'Achilles tendon'.

left-predictive - e.g. 'inferiority complex' / 'Oedipus complex'.

Variable Phrases - the most common phrases. Sequences of words which co-occur more often than their individual frequencies would lead us to expect, e.g. 'glass of water', 'classical music'.

Reading a passage of text, one is left with the impression that it moves from one of these set expressions to another, with intervening elements being non-collocational and freely available. However, this is just a first impression. Even the words occurring between set combinations constitute groups whose form and order are likely to be conditioned in varying degrees by patterns of collocability.

Kjellmer suggests that we should think in terms of a 'collocational continuum'. At one end are established collocations, at the other are sequences of dubious cohesion.

In discourse, it is suggested that we largely make use of chunks of well-established, pre-fabricated material, that allow us to move through the discourse swiftly.

Renouf & Sinclair, (1991), look at the frameworks in which collocations occur, and at the tendency of these frameworks to 'enclose' characteristic groupings of words.

Frameworks are defined as consisting of a discontinuous sequence of two words, positioned at one word remove from each other. They are not grammatically self-standing, and their well-formedness depends on what intervenes.

The frameworks chosen for study were :

a + ? + of

be + ? + to

for + ? + of

an + ? + of

too + ? + to

had + ? + of

many + ? + of

The different frameworks tended to attract particular word classes, for example, **a + ? + of** and **an + ? + of** tended to attract nouns.

The selection of the word enclosed by the framework seems to be governed by the combined influence of the framework pair.

2 . 4 . 5 - The Use of Collocation in Text Recognition

Some work has been carried out in recent years on the use of collocations as an aid to text recognition (Evettet al, 1992; Rose & Evett, 1992; Rose & Evett, 1994).

Co-occurrence statistics derived from a corpus of text were used to narrow down the number of candidate words for each word position in a sentence, produced as output by a low-level recogniser.

For each word in the lexicon, a list of collocates was given. E.g :

mortgage
[own, own, own, year, stock, stock,
property, property, pay, new, money,
local, lend, lend, lend, issue, increase,
authority, advance, advance]

The degree of repetition of any one collocate corresponded to the strength of association between the node and that collocate.

Pairs of words in a potential sentence were considered (one node and one collocate). The collocate is assigned some score depending on whether it and the node were in each other's collocate list (indicating a strong overlap), or if they had other words in common (a weak overlap). The correct word was assigned the highest score in more than 70% of cases.

Hull, (1994), also uses collocational statistics to reduce the word candidate list at each word position.

Word candidates that had stronger word collocation with their neighbour words were selected as matching the input.

The results attained in these studies suggest that more detailed research into the use of collocation statistics in automatic text recognition may prove fruitful.

Chapter 3

The Provision of Input

3 . 1 - Word Lattices

The output from a low-level text recognition system will propose a number of alternative hypotheses for each word position within a section of text (in this case, an orthographic sentence), and, at a lower level, for each character position within a word.

This can conceptually be thought of as a lattice. E.g. at the character level the word **cat** may produce the lattice similar to that shown in **Fig. 3.1** (a letter lattice).

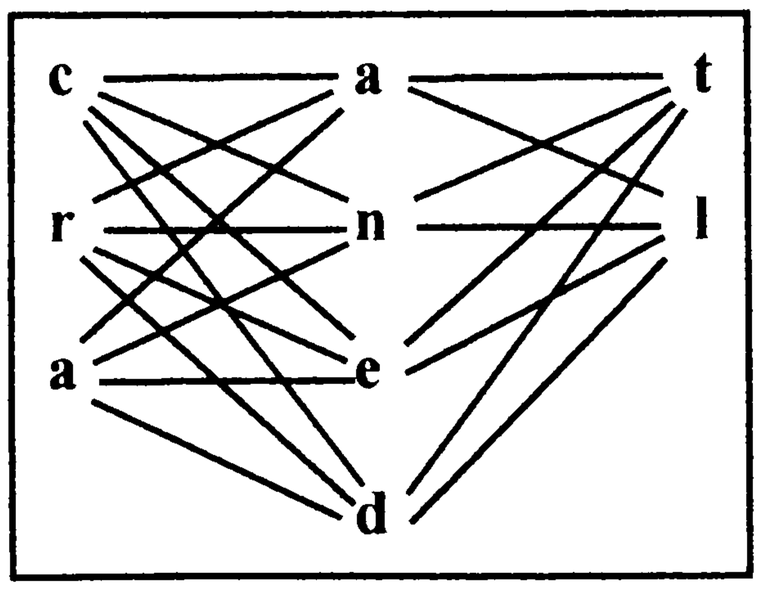

Fig. 3.1 – A letter lattice

At the word level, the sentence **the cat sat on the mat** may produce a word lattice as shown in **Fig. 3.2**.

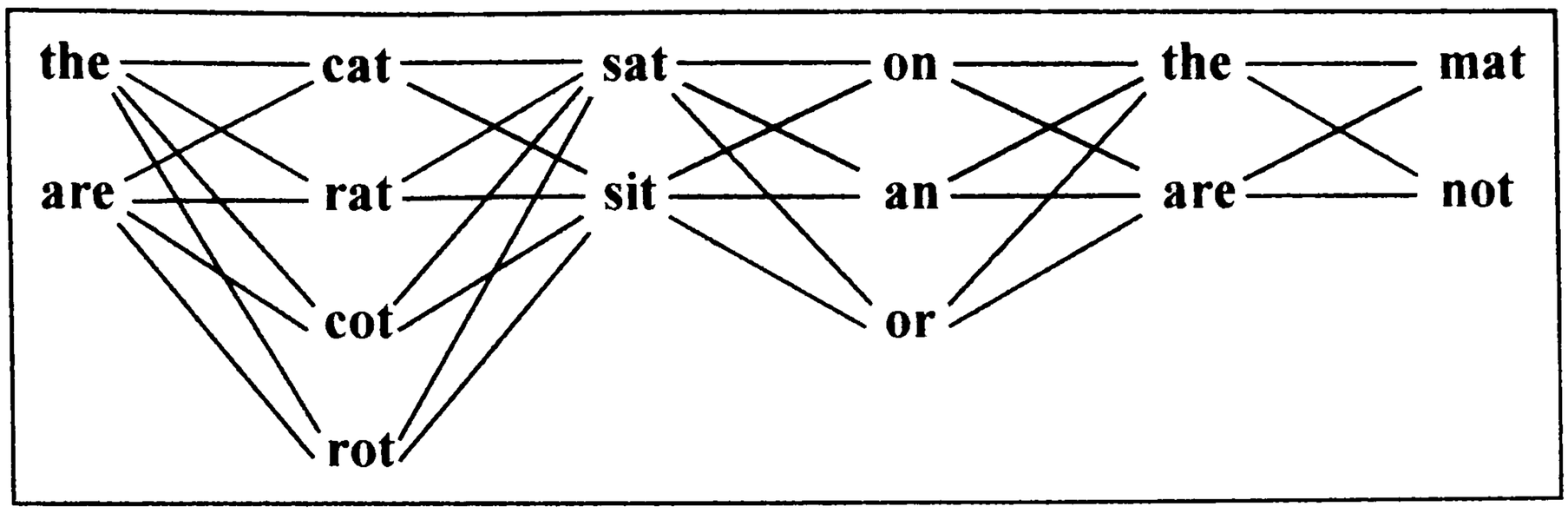

Fig. 3.2 – A word lattice

These lattices are very simplistic and undoubtedly unrealistic examples, merely used to illustrate a point. It is highly unlikely for instance that each candidate word in a particular position would contain exactly the same number of letters. The letter **d** for example is quite likely in a low-level recogniser to be confused with the letter pair **cl**, and *vice versa*.

Many other word segmentation difficulties are likely to surface in a genuine word lattice.

The ultimate aim of a post-processing component incorporating linguistic knowledge is to remove the uncertainty inherent in this type of input, and identify the correct word at each word position in a sentence.

In practice, the linguistic knowledge-based component of a system will assign some sort of score to each candidate word at each word position, giving a measure of the likelihood that that is the correct word. These individual word scores can then be combined to give an optimal path through the lattice, which will represent the system's hypothesis as to the identity of the input sentence.

3 . 2 - Simulation of Low-level Output

Ideally of course, any post-processing system would be tested by passing hand-written data to a low-level (visual) recognition system, and passing on the output from this to the linguistic processor.

However, easy and consistent access to a low-level recogniser was unfortunately not possible, so a method was needed to simulate the output of such a recogniser.

A number of sample word lattices from an existing low-level recogniser were available (see **Appendix A** for a sample), and these were used as the basis for a letter substitution database which could be used in the generation of simulated word lattices.

The sample word lattices offered a large number of letter substitutions from which to construct a database. For each correct word in the lattices, an average of 22 candidate words were offered. Given a (rounded) average word length of 5 letters, this means that an average of 110 substitutions were offered for each letter by each lattice. Given that a total of 310 sample word lattices were used this means that the letter substitution database is based on a total of **34,100** sample letter substitutions.

The actual construction of the letter substitution database is explained in the following section.

3 . 2 . 1 - The Letter Substitution Database

The correct word (henceforward referred to simply as the word), at each word position in these sample lattices was first identified, and separated from the other, incorrect words for that position (henceforward referred to as the candidates).

Given a set of words and a set of corresponding candidates, it is possible to ascertain the mistakes that the low-level recogniser has made in assigning letters to an input pattern. These can be generalised and stored in a database for use by a simulator - the letter substitution database.

The database has an entry for each letter of the alphabet. Each entry is of the form :

letter

substitution 1

substitution 2

.....

substitution n

where **letter** is a letter of the alphabet, and the alternatives represent substitutions made by the low-level recogniser in its output. An alternative will often be a single letter, but may be a letter pair, or a special character representing a null character (see below for an explanation of the use of this null character) .

Taking one word and one candidate, there are three cases to be considered when compiling this database :

- the word has fewer letters than the candidate
- the word has more letters than the candidate
- the word and the candidate have the same number of letters

In the first case, one or more single letters in the word have been substituted for a letter string to give the candidate. e.g. the letter **d** in the word is represented as the pair **cl** in the candidate. I shall call this letter expansion.

In this case, each letter in the word is assumed to have been replaced by the single letter in the equivalent position in the candidate, *and* by the letter pair consisting of the letter in the equivalent position in the candidate and its immediate neighbour to the right.

Take for example the word **dog** and the candidate word **clog**. Clearly in this example the letter **d** has been replaced by the letter pair **cl**.

When analysing the letter substitutions, the process will go through the word **dog** letter by letter. As the candidate word **clog** is longer than the input word **dog**, each letter in **dog** is assumed to have been replaced by the equivalent letter in the candidate word, *and* by the letter pair consisting of the equivalent letter in the candidate word and its neighbour to the right. I.e. the **d** in **dog** is assumed to have been replaced by the letter **c** and by the letter pair **cl** in **clog**.

Likewise, the letter **o** in **dog** is assumed to have been replaced by the letter **l** and by the letter pair **lo** in **clog** and so on.

This method clearly has its limitations. A number of spurious and intuitively inappropriate letter substitutions are generated (as in **o** being replaced by **lo** above for instance). Also, this method produces unsatisfactory substitutions when the candidate word is considerably longer than the input word.

However, in the sample lattices upon which these substitutions are based there are very few candidates that are longer than the head word, and fewer still that are more than one letter longer.

As for the spurious substitutions, the important thing is that the intuitively correct substitutions are picked up (the substitution of **cl** for **d** for instance in the example above). It doesn't really matter that a number of extra, and spurious, substitutions are generated, as long as the correct substitution is included.

In the second case, (the input word has more letters than the candidate word), one or more letter strings in the word have been converted to shorter strings to give the candidate, e.g. the pair **ln** is represented as the letter **h** in the candidate word. I shall call this letter conflation.

In this case, each letter in the word is assumed to have been replaced by the letter in the equivalent position in the candidate, and also by a null character (represented by a special character in the candidate database).

This of course is not an accurate representation of what is happening, but consider the practical operation of the lattice simulator.

Given an input word containing the letter pair **ln**, one of the permutations according to the candidate database is that the letter**l** will be replaced by the letter **h**, and that the letter **n** will be replaced by a null character. So in effect, the letter pair **ln** will have been replaced by the letter **h**, which is an accurate reflection of the low-level recogniser's output.

In the third case, (the input word and the candidate word are of the same length), it is assumed that each letter in the word has been directly replaced by the letter in the candidate in the corresponding letter position.

This assumption is not necessarily correct. Consider a case where the word contains the letter **d** and the letter pair **ln**, and contains six letters in all. In the candidate, the letter **d** may be represented by the pair **cl**, and the pair **ln** by the letter **h**, while all the other letters in the word are represented by single letters in the candidate. Clearly in this case, the candidate will also contain six letters, and the expansion and conflation will not be picked up.

However, most of the common (and many uncommon) letter expansions and conflations will be represented in the database through the cases where the word length and candidate length are different, so for the sake of computational simplicity I feel the assumption is acceptable.

After this process of identifying the letter substitutions contained in the sample lattices, we are left with an unstructured file containing lines of the form :

letter substitution letter

This file must be processed to give a file of the form :

letter 1	**substitution 1.1**	**frequency 1.1**
letter 1	**substitution 1.2**	**frequency 1.2**
.	.	.
.	.	.
.	.	.
letter 1	**substitution 1.*n***	**frequency 1.*n***
letter 2	**substitution 2.1**	**frequency 2.1**
.	.	.
.	.	.
.	.	.
letter 2	**substitution 2.*n***	**frequency 2.*n***
letter *n*	**substitution *n*.1**	**frequency *n*.1**
.	.	.
.	.	.
.	.	.
letter *n*	**substitution *n*.*n***	**frequency *n*.*n***

The **frequency** field gives the number of times **letter** was replaced by **substitution** in the sample lattices.

One further step is required - that of trimming the file. It was decided that for each letter of the alphabet, only the four most frequently occurring substitutions would be stored in the letter substitution database.

At first glance this may seem a small number, but consider a six letter input word. For each letter position, given the original, correct, letter and four letter substitutions, this letter lattice will produce$\mathbf{5^6}$ (**15,625**) candidate words.

The decision to retain just the top*four* was arbitrary, but the decision to limit the number *per se* was not. It is common in a low-level recognition system to assign each letter hypothesis in a word some sort of score denoting the likelihood of it being correct, and to limit the number of hypotheses presented according to these scores.

As no such scores were available to me, it seemed sensible to limit the number of permissible substitutions according to how often they occurred in the sample lattices.

So we now have our letter substitution database, with each entry of the form :

letter

substitution 1

substitution 2

substitution 3

substitution 4

For a listing of the actual letter substitution database derived from the sample word lattices, see **Appendix A**.

3 . 2 . 2 - Producing Simulated Output

With the letter substitution database in place, it is now straightforward to simulate the word-lattice output of a low-level recogniser.

Any test sentence passed as input to the collocation processor is broken down into individual words. Each of these words is broken down into individual letters.

Now a letter lattice can be constructed for each word in the input sentence using the information stored in the letter substitution database, and from this letter lattice a list of candidate words for that word position in the sentence can be built.

The system lexicon is searched with reference to the first two letters in a word, so words beginning with invalid letter pairs can be immediately removed from the list of candidates. For a full description of how the lexicon is searched see **4 . 1 – The Lexicon**.

To further ease the not inconsiderable task of traversing the letter lattice, a matrix of 'illegal' letter pairs is also consulted.

This matrix was created by analysing the lexicon (in practical terms a simplified word list), to see which letter pairs never occur in the permitted list of words. Any letter pairs that never occur are marked with a**0** entry in a 26 * 26 matrix.

When the word lattice generator is traversing the letter lattice for a word, it can therefore consult this matrix for each letter pair it comes across, and immediately discount those containing a **0** entry, thereby saving processing time.

The candidate words that are eventually produced by this traversal of the letter lattice must be looked up individually in the lexicon to weed out the words that are not recognised by the system.

Chapter 5 gives a full description of the lexical filtering carried out during the construction of a word lattice.

Ultimately for each word position in the input sentence there will be a list of valid candidate words. These can be combined to form a word lattice which is then used as input to the collocation processor.

Chapter 4

The System Components

A word lattice presented to the system is processed with reference to two data sources :

- the lexicon
- the collocation dictionary

Although I have referred to these in the singular, there are in fact two lexicons and two collocation dictionaries, used to carry out different experiments in order to evaluate different methods of representing collocational knowledge.

The two versions of the lexicon are the same in terms of format but differ slightly in content, while the two collocation dictionaries actually contain different types of information.

The differences will be explained fully in the sections that follow. (As the two versions of the lexicon differ only in detail, I shall refer to a single lexicon in subsequent sections for reasons of readability).

The lexicon and its corresponding collocation dictionary are very closely linked, and it is virtually meaningless to consider one without the other. The two interact closely, and in different ways at different stages of the processing :

- In the creation of an initial list of collocations, words are stored in the collocation list in terms of their position in the lexicon, and word frequency information is written to the lexicon (see **Fig. 4.1**).

- In the creation of the final collocation dictionary (or dictionaries), word numbers are written from the lexicon to the collocation dictionary, and file position data are written from the collocation dictionary to the lexicon (see **Fig. 4.2**).

  (The two stages of processing mentioned above are described in detail in **4 . 2 - The Collocation Dictionary**).

- In the processing of word lattices, the lexicon is used for word look-up, and to provide position information about the collocation dictionary (see **Fig. 4.3**).

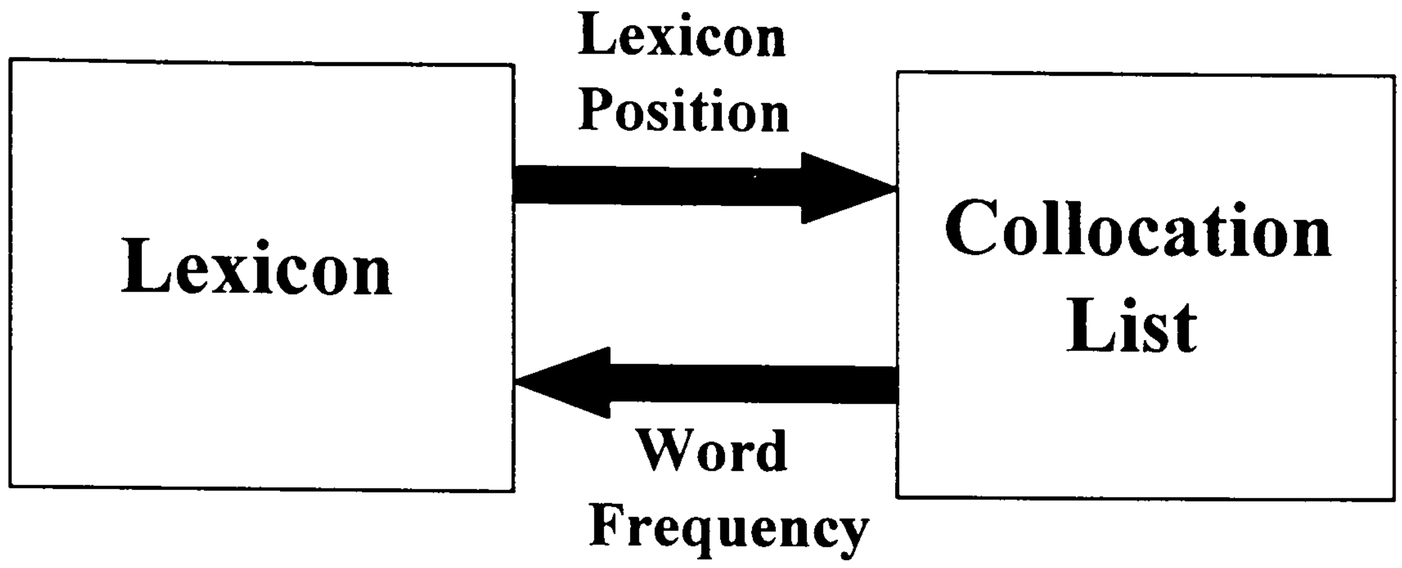

Fig. 4.1 – The interaction between the lexicon and the collocation data during the creation of an initial collocation list.

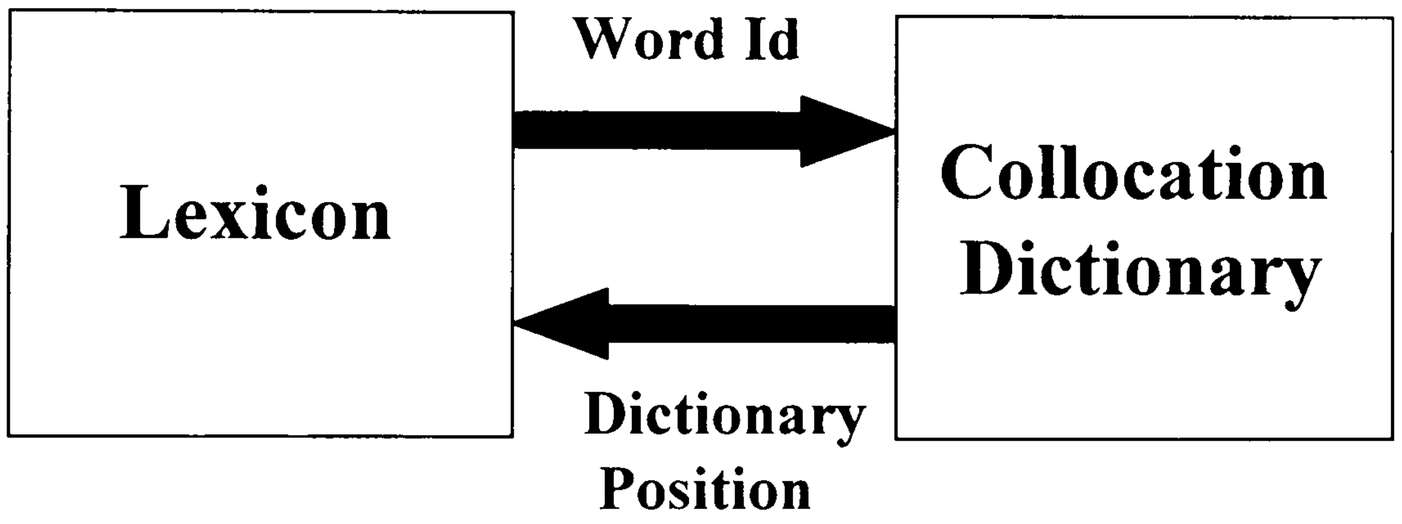

Fig. 4.2 – The interaction between the lexicon and the collocation dictionary during the creation of the final collocation dictionary.

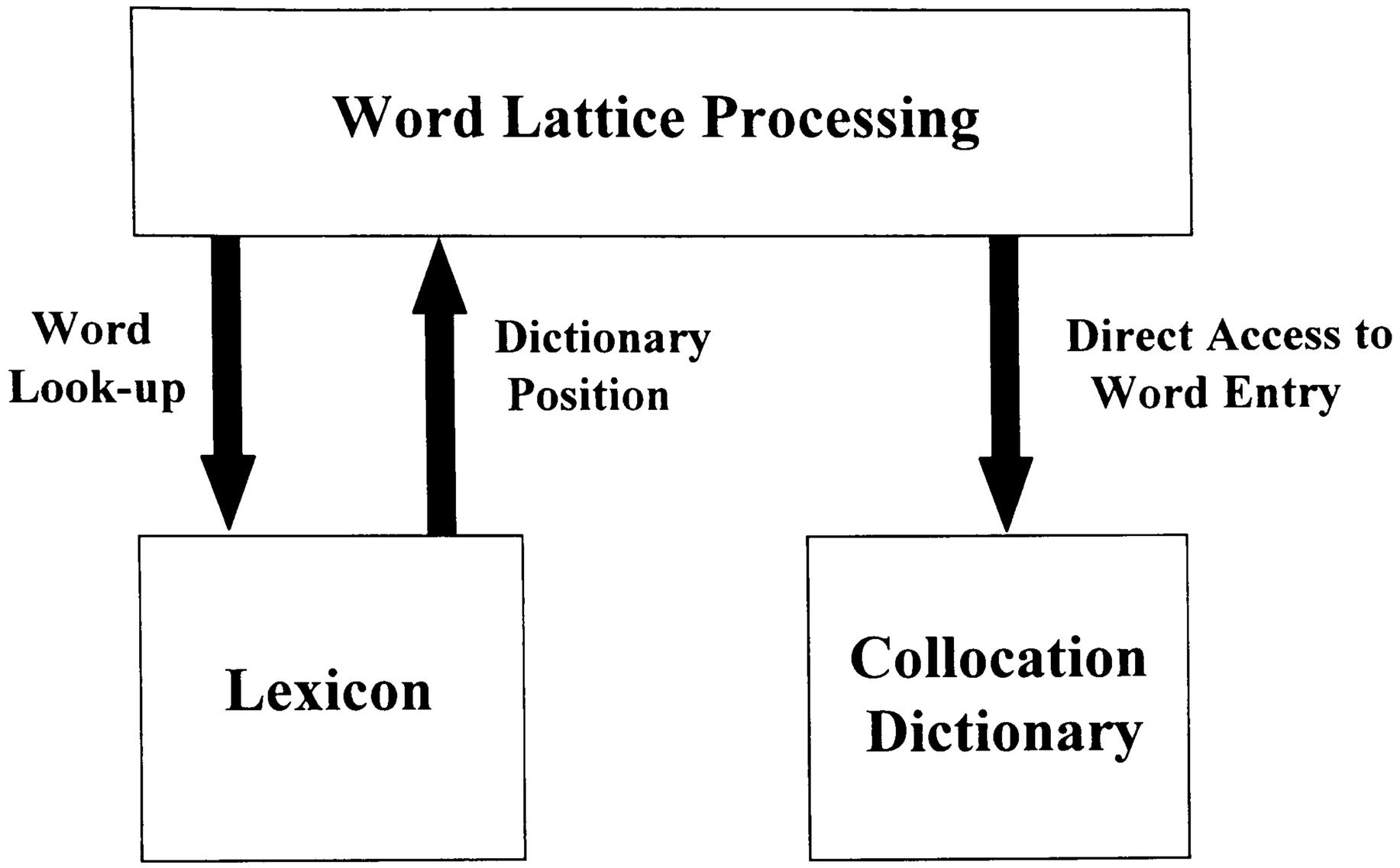

Fig. 4.3 – The interaction between the lexicon and the collocation dictionary during the processing of word lattices.

4 . 1 - The Lexicon

4 . 1 . 1 - Lexicon Structure

The main function of the lexicon is to denote which words are recognised by the collocation processor. However, the lexicon for this system is more than just a simple list of words.

The words in the lexicon (78,055 in all) are taken from the **Collins Electronic Dictionary**. Being a full dictionary, this also contains part of speech and morphological information and the first step in the creation of the system lexicon consisted of stripping this information away to give, initially, a bare list of words. More information was added to the lexicon during subsequent processing.

The format of the lexicon for each word entry is as follows :

word name
word number
word frequency
pointer to collocation dictionary

Each word in the lexicon is assigned a word number (from **1** to **78055**). The word number is used to refer to the word in the collocation dictionary. This is more space efficient than referring to the word by name.

The next field - the word frequency - gives the number of occurrences of the word in the input text used to create the collocation dictionary.

This information is used in the creation of both versions of the collocation dictionary. The number of occurrences of a word is an important factor in ascertaining the significance of a collocation between that word and another word.

The word frequency measure is also used in the actual experiments carried out with the collocation processor.

The final field contains a pointer to the collocation dictionary. This is where the two versions of the lexicon differ.

Obviously if there are two versions of the collocation dictionary, then pointers to positions within these two versions will be different. It was therefore decided to maintain two versions of the lexicon, one for each version of the collocation dictionary.

The pointer stored in each word entry in the lexicon gives the position in the collocation dictionary where the collocation entry for that word begins.

This allows a direct jump during the processing of a word lattice to the relevant entry in the collocation dictionary, followed by a sequential search of that specific entry for the particular collocation required.

Extracts from the Collins Dictionary and the system lexicon can be found in Appendix B.

4 . 1 . 2 - Looking Up a Word in the Lexicon

Word look-up in the lexicon is based on the first two letters of the word in question. This method offers a marked improvement in efficiency compared with a search based on just the first letter of a word.

Assuming that there is an equal number of words beginning with each letter of the alphabet, a search based on the first letter alone of a word could take up to 3002 accesses of the lexicon (i.e. an average of around 1500 accesses). Again assuming an even distribution of words beginning with every possible letter pair, a search based on the first two letters could take up to 115 accesses of the lexicon (i.e. an average of around 57 accesses). Given the number of word look-ups required during the various stages of processing, this gives a dramatic improvement in performance.

A search based on the first three letters of each word would offer an even greater improvement in efficiency, but would incur an added overhead in finding the starting point for a sequential search due to the need to access a three-dimensional matrix.

It was found that a search based on the first two letters of a word offered the best trade-off between this overhead and the number of accesses of the lexicon required.

A 26*26 matrix (the **jump** matrix) contains the pointers to positions in the lexicon. So, for instance, the matrix entry representing the letter pair**ab** contains the file position (in bytes) in the lexicon of the first entry alphabetically for a word beginning with the letters **ab** (see **Fig. 4.4**).

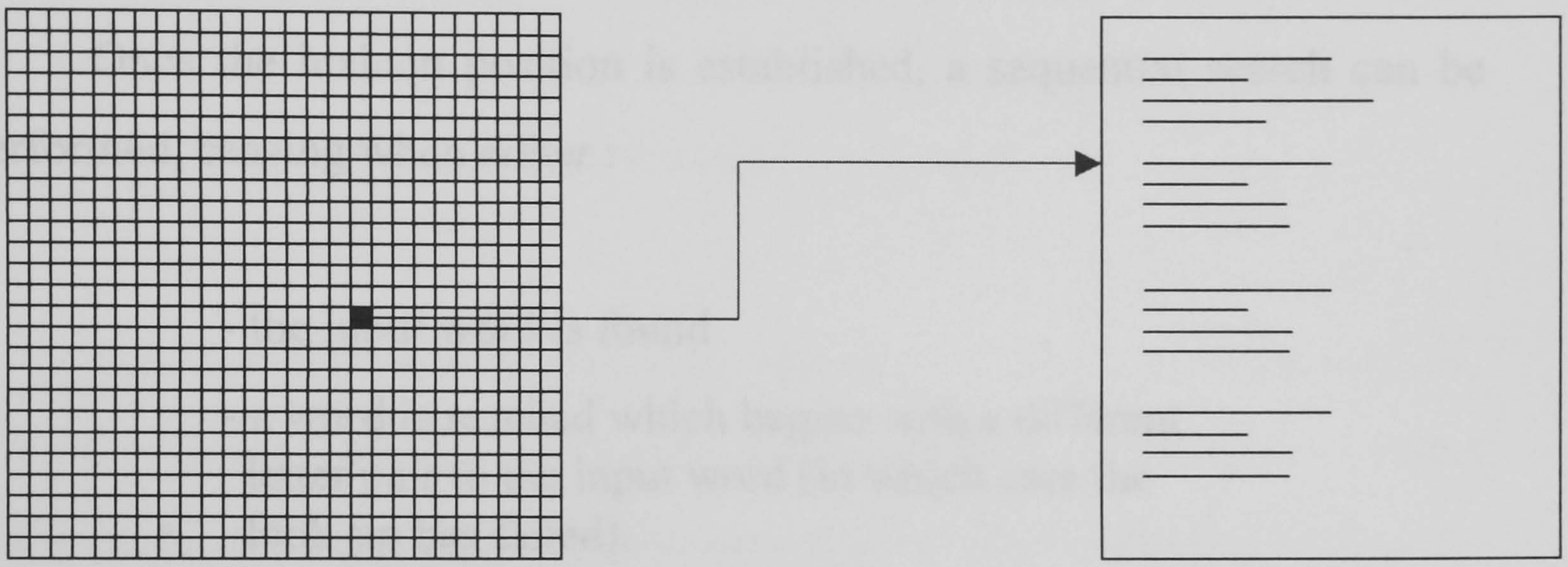

Fig. 4.4 – Using the Jump Matrix to pinpoint the starting point for a sequential search of the lexicon

In the case where there is no word in the lexicon beginning with a particular letter pair a **-1** is stored in the matrix entry corresponding to that letter pair.

The case of single letter words is covered by the matrix entries for letter pairs where the second letter is **a**. These point to the word entry in the lexicon for the first letter in the pair on its own (if that individual letter exists as a distinct lexicon entry). For example, the lexicon position of the word**a** is stored in the jump matrix position corresponding to the letter pair **aa**.

The use of the jump matrix means that words beginning with invalid letter pairs (or at least invalid with reference to this particular lexicon) can be rejected very rapidly, without wasting time and resources on a fruitless search.

Once the lexicon position is established, a sequential search can be performed, ceasing when either :

- the input word is found
- a word is reached which begins with a different letter pair to the input word (in which case the look-up has failed).

It should be noted at this point that the lexicon generally stores only the root form of a word and none of its derivations. For this reason there is an element of morphological processing in the word look-up process, to deal with different derivations of the same root word. There are two cases to consider :

- irregular derivations
- derivations with regular endings

If a word look-up fails initially, then two further steps of processing are carried out to check for these two possibilities.

First, a file of irregular derivations is checked. This file contains information about derivations of the root forms of words for which no generic rules can be applied. The file is based on a similar file used by the Natural Language Processing System (NLPS) in the Department of Computer Studies at Loughborough University. Each entry is of the form :

irregular derivation
root word

Take for instance the word **was.** This may not be found in the lexicon, but its root word, **be,** will be present. It would clearly be an error to reject the word **was** as invalid, so the system would search the irregular derivations file for the string **was**. This file is of no great size, so a straight sequential search was deemed to be acceptable.

If the word being looked up is found amongst the list of irregular derivations, then it is assigned the same word number as its root word. Collocations of all derivations of the same root word are considered to be collocations of that root word for the purposes of this project. See Halliday, (1966) for discussion of this issue.

Should the word not occur in the irregular derivations file, then the second case (that it is a derivation with a regular ending) is considered.

A file of regular endings, also based on a similar file used by the NLPS is consulted. Each entry is of the form :

derivation ending
potential root ending

So the entry :

ed
e

would deal with the derivation **dined**, for instance. The process would strip off the ending **ed** and replace it with the ending **e** to give the root word **dine**.

So when a word look-up has failed, and the search of the irregular derivations file has also proved fruitless, the file of regular endings is worked through sequentially. When a regular ending is found which matches the ending of the word being looked up, then the ending of the word is stripped off and replaced by the potential root ending specified in the regular endings entry. The new word thus created is searched for in the lexicon. If it is found, then the original word being looked up is assigned the word number of the root word as stored in the lexicon.

In some entries in the regular endings file, the potential root ending is given as the character *. This denotes that the ending is stripped off and replaced by nothing. E.g. the entry :

s
*****

would deal with the derivative **dines**. The **s** ending is stripped off and replaced by nothing to again give the root word **dine**.

I have called the second part of each entry in the file the*potential* root ending as there maybe more than one entry for each derivative ending, e.g. the entries :

ed
e

and

ed
*****

As shown above, the first entry would deal with derivatives such as **dined**, but given, say, the word **jumped** the first entry would produce **jumpe**. This is clearly incorrect.

The second entry would strip of the **ed** and replace it with nothing, giving **jump** - the correct root.

It should be noted that many derivatives are listed explicitly in the lexicon as well as their root words, so this morphological processing is often not necessary, and a straight look-up of the word will suffice.

This method of word look-up has proved itself to be quite efficient which is of vital importance considering the enormous number of word look-ups required during all stages of processing.

The next section will discuss the creation and structure of the collocation dictionaries used by the system.

4 . 2 - The Collocation Dictionary

The collocational relationships between a group of words in a text can be viewed as a network. Diagrammatically, a group of four words can be shown as in **Fig. 4.5**.

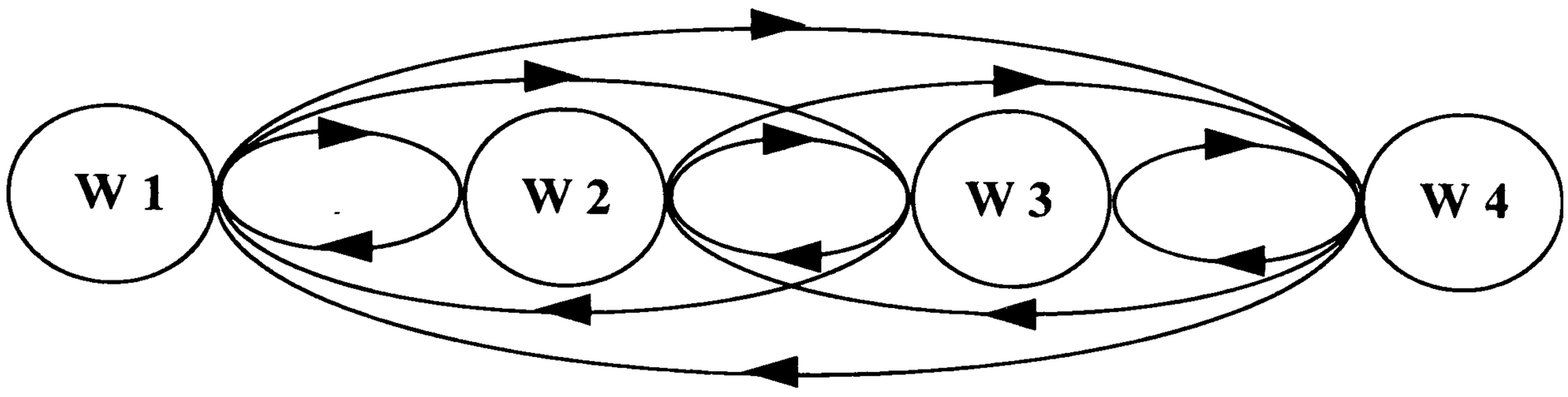

Fig. 4.5 – Representation of the collocational relationships between four contiguous words. W1 - 4 represent the four words in the body of text, and the arrowed lines represent the collocational relationships between these words. Note that each collocational relationship is represented by two arrows, one pointing in each direction.

Consider two words **A** and **B**. The collocational relationship between these two words is different when **A** is the node and **B** is the collocate from when **B** is the node and **A** is the collocate. The two arrows per relationship as shown in **Fig. 4.5** are needed to express this difference (See Sinclair, 1991 for a discussion of 'upward' and 'downward' collocation, summarised in **2 . 4 . 3 - A Study of Collocation**).

In practical terms, it would be inefficient of storage space and processing time to explicitly represent this two-way relationship.

The proposed structure of the collocation dictionary is therefore a cascade based on alphabetical order. I.e. given two collocationally linked words, the first word alphabetically is considered as an 'anchor', to which the other word's position is relative. **Fig. 4.6** gives a diagrammatical representation of this structure.

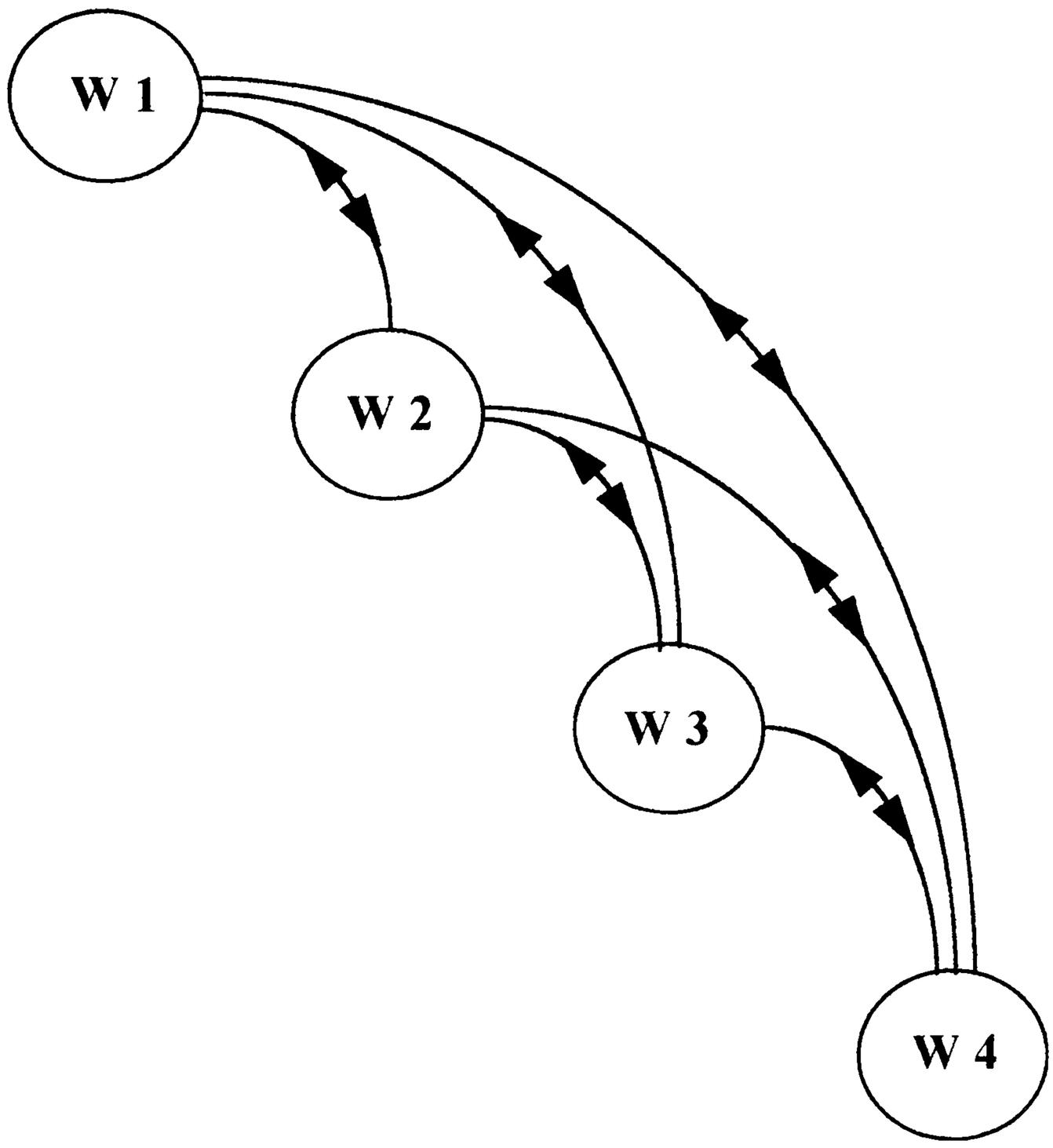

Fig. 4.6 –The collocational relationships between four contiguous words represented as a cascade based on alphabetical order. Each relationship is represented by one link, with the alphabetically earliest word acting as an anchor. In this example words W1 – W4 are in ascending alphabetical order.

As can be seen from **Fig. 4.6**, each relationship between words is represented by a single link, but the two-way nature of the relationship can be encapsulated in this one link.

As well as saving storage space, this representation of collocation relationships makes the processing of the collocation dictionary faster and easier.

4 . 2 . 1 - The Format of the Collocation Dictionary

As has been mentioned previously, there are in fact two collocation dictionaries used by the system for different experiments, but the format of both versions can be generalised as follows :

Entry Number *1*

Word Number *1.1* (Position *1.1*) Frequency *1.1* Strength *1.1*

Word Number *1.2* (Position *1.2*) Frequency *1.2* Strength *1.2*

. . . .

. . . .

Word Number *1.n* (Position *1.n*) Frequency *1.n* Strength *1.n*

Entry Number *2*

Word Number *2.1* (Position *2.1*) Frequency *2.1* Strength *2.1*

. . . .

. . . .

Word Number *2.n* (Position *2.n*) Frequency *2.n* Strength *2.n*

. . . .

. . . .

Entry Number *n*

Word Number *n.1* (Position *n.1*) Frequency *n.1* Strength *n.1*

. . . .

. . . .

Word Number *n.n* (Position *n.n*) Frequency *n.n* Strength *n.n*

where :

Entry Number is the number, stored in the lexicon, used to represent a word recognised by the system which has occurred in the input text used for the construction of the collocation dictionary (see **4 . 2 . 2 . 1 - The British National Corpus**, later in this chapter. Any subsequent reference to the 'input text' in this chapter will be to this text, and not to any text used in the testing of the system at a later stage).

Word Number also represents a word which has a collocational relationship in the input text with the word represented by **Entry Number**.
Due to the 'cascade' structure of the dictionary, **Word Number** will always be either equal to (in the case of a word collocating with itself) or greater than **Entry Number**.

The two versions of the collocation dictionary differ in the content of the remaining fields in each entry. These differences are documented in the following sections, but in general terms, the other fields give the following information :

(Position) gives the position of the word represented by **Word Number** in relation to the word represented by **Entry Number**. I.e. if the word represented by **Word Number** directly preceded the word represented by **Entry Number** in the input text, then **Position** would be **-1**. This field is shown in brackets as it is only present in one version of the collocation dictionary.

Frequency gives the number of times that the two words represented by **Entry Number** and **Word Number** collocate in the input text.

Strength gives the strength of the collocational attraction between the two words. This field actually contains two pieces of information, representing the two-way nature of collocation discussed earlier.
The fundamental difference between the two versions of the collocation dictionary lies in how the strength of the attraction between words is represented.

Below I will describe in detail the differences between the two versions of the collocation dictionary - the **Percentage Score Dictionary** and the **Z-score Dictionary**.

4 . 2 . 1 . 1 - The Percentage Score Dictionary

The format of the Percentage Score Dictionary is as follows :

Entry Number *1*

Word Number *1.1* Position *1.1* Frequency *1.1* N%*1.1* C%*1.1*

Word Number *1.2* Position *1.2* Frequency *1.2* N%*1.2* C%*1.2*

. . . .

. . . .

Word Number *1.n* Position *1.n* Frequency *1.n* N%*1.n* C%*1.n*

Entry Number *2*

Word Number *2.1* Position *2.1* Frequency *2.1* N%*2.1* C%*2.1*

. . . .

. . . .

Word Number *2.n* Position *2.n* Frequency *2.n* N%*2.n* C%*2.n*

. . . .

. . . .

Entry Number *n*

Word Number *n.1* Position *n.1* Frequency *n.1* N%*n.1* C%*n.1*

. . . .

. . . .

Word Number *n.n* Position *n.n* Frequency *n.n* N%*n.n* C%*n.n*

The **Entry Number** and **Word Number** fields need no explanation further than that given in the previous section.

The other fields need a little further discussion, as these differ in the two versions of the dictionary.

The two-way nature of collocational relationships as discussed previously are encapsulated in the **Position**, **N%** and **C%** fields.

The **Position** field gives the position in the input text of the word represented by **Word Number** in relation to the word represented by **Entry Number**. However, it is clear that simply by reversing the polarity of the **Position** field, we can ascertain the position of the word represented by **Entry Number** in relation to the word represented by **Word Number**.

Consider the two words **the cat**. The position of **the** in relation to **cat** is **-1**. Reversing the polarity gives **1**, i.e. the position of **cat** in relation to **the**.

The **Frequency** field is ostensibly the same in both versions of the dictionary, in that it records the frequency of the collocation. However, in the Percentage Score Dictionary, the frequency gives the number of times that the words co-occur with reference to the *specific position* given in the **Position** field.

The **N%** and **C%** fields record a measure of the strength of the collocation between the two words. There are two cases to consider :

- the word represented by **Entry Number** is the node, and the word represented by **Word Number** is the collocate

- the word represented by **Word Number** is the node, and the word represented by **Entry Number** is the collocate

The two fields **N%** and **C%** take these cases into account. So for the **N%** field, the word represented by **Entry Number** is considered to be the node, while for the **C%** field, this word is considered to be the collocate. So once again, the two-way nature relationship is implied in a single entry.

The actual measure of collocational strength used in this dictionary is a percentage score calculated with the formula :

$$\frac{\textbf{No. of collocations between words in given relative positions}}{\textbf{Total no. of occurrences of node in the text}} \textbf{*100}$$

Expressed in English, the measure of the strength of the collocation is the number of collocations between a node word and its collocate in a text in given relative positions, as a percentage of the total number of times the node word occurs in that text.

This measure gives the significance of a collocation in terms of one of the words involved in the collocation. Of course, this measure is represented in the dictionary for both words involved in the collocation, so these can be combined to give an idea of the overall significance of the collocation.

One of the interesting features of this measure of collocational significance is that it is *position-sensitive*. So, for instance, if two words collocate frequently in adjacent positions, but very rarely with another word in between them, the percentage score measure will reflect that.

The Z-score Dictionary differs in this respect, in that it takes into account only a span of words, and not specific positions within that scan. This does give a more efficient representation, as it is not necessary to store position information.

Another source of efficiency in the Z-score representation is that only significant collocations are stored in the Z-score Dictionary, whereas all the collocations in the input text are stored in the Percentage Score Dictionary.

The Z-score Dictionary is discussed in detail in the next section.

4 . 2 . 1 . 2 - The Z-score Dictionary

The format of the Z-score Dictionary is as follows :

Entry Number *1*

Word Number *1.1* Frequency *1.1* Nzscore*1.1* Czscore*1.1*

Word Number *1.2* Frequency *1.2* Nzscore*1.2* Czscore*1.2*

. . . .

. . . .

Word Number *1.n* Frequency *1.n* Nzscore*1.n* Czscore*1.n*

Entry Number *2*

Word Number *2.1* Frequency *2.1* Nzscore*2.1* Czscore*2.1*

. . . .

. . . .

Word Number *2.n* Frequency *2.n* Nzscore*2.n* Czscore*2.n*

. . . .

. . . .

Entry Number *n*

Word Number *n.1* Frequency *n.1* Nzscore*n.1* Czscore*n.1*

. . . .

Word Number *n.n* Frequency *n.n* Nzscore*n.n* Czscore*n.n*

The first obvious departure from the Percentage Score Dictionary is the absence of the **Position** field. The Z-score Dictionary represents the collocational relationships between words occurring within a given span without consideration of their relative positions within that span.

The **Frequency** field gives the number of times that the words occur together in the input text within the given span.

The **Nzscore** and **Czscore** fields represent z-score values (see **2 . 4 . 2 - Collocation** for an outline of how z-scores are calculated, and Berry-Rogghe, 1973 for a fuller explanation). Z-scores are another way of representing the strength of the collocational relationship between two words.

The **Nzscore** field gives the z-score when the word represented by **Entry Number** is the node in the collocation, and the **Czscore** field gives the z-score when this word is the collocate.

Only entries where either the **Nzscore** field *or* the **Czscore** field exceeds the significance level of **2.576** (suggested in Berry-Rogghe, 1973) are included in the Z-Score Dictionary.

This means that only words exhibiting a significant attraction to one another (or at least in one direction) are stored in the dictionary.

4 . 2 . 2 - The Creation of the Collocation Dictionary

Although there are two versions of the collocation dictionary, the processes involved in the creation of both versions were generically the same, and only need describing once.

The input to the creation process was in the form of a body of text, divided into sentences. This body of text was analysed on a sentence by sentence basis, and information about the collocational relationships within each sentence was extracted.

The origin of the input text will be explained later in this chapter in section **4 . 2 . 2 . 1 - The British National Corpus**.

For filestore reasons, it was impossible to use the British National Corpus in its entirety (some 100 million words). The first section of the corpus amounting to over 13 million words was used as input to my system. For information about the texts excerpted to make up this section of the corpus, see Burnard, (1995), pp.151-179.

The raw input text first had to be processed to transform it into a suitable form for processing. The pre-processing required to be carried out on the input text is described in section **4 . 2 . 2 . 2 - Pre-processing the BNC.**

Before the final collocation dictionaries were created, an intermediate collocation list for the input text was produced. This is described in section **4 . 2 . 2 . 3 - The Creation of a Collocation List**.
This section will also describe a number of important initial decisions made about the make-up of the collocation dictionaries.

Section **4 . 2 . 2 . 4 - The Creation of a Collocation Dictionary** describes the transformations carried out on the intermediate collocation list to create a usable collocation dictionary.

Fig. 4.7 gives a diagrammatical overview of the steps required to create the collocation dictionary.

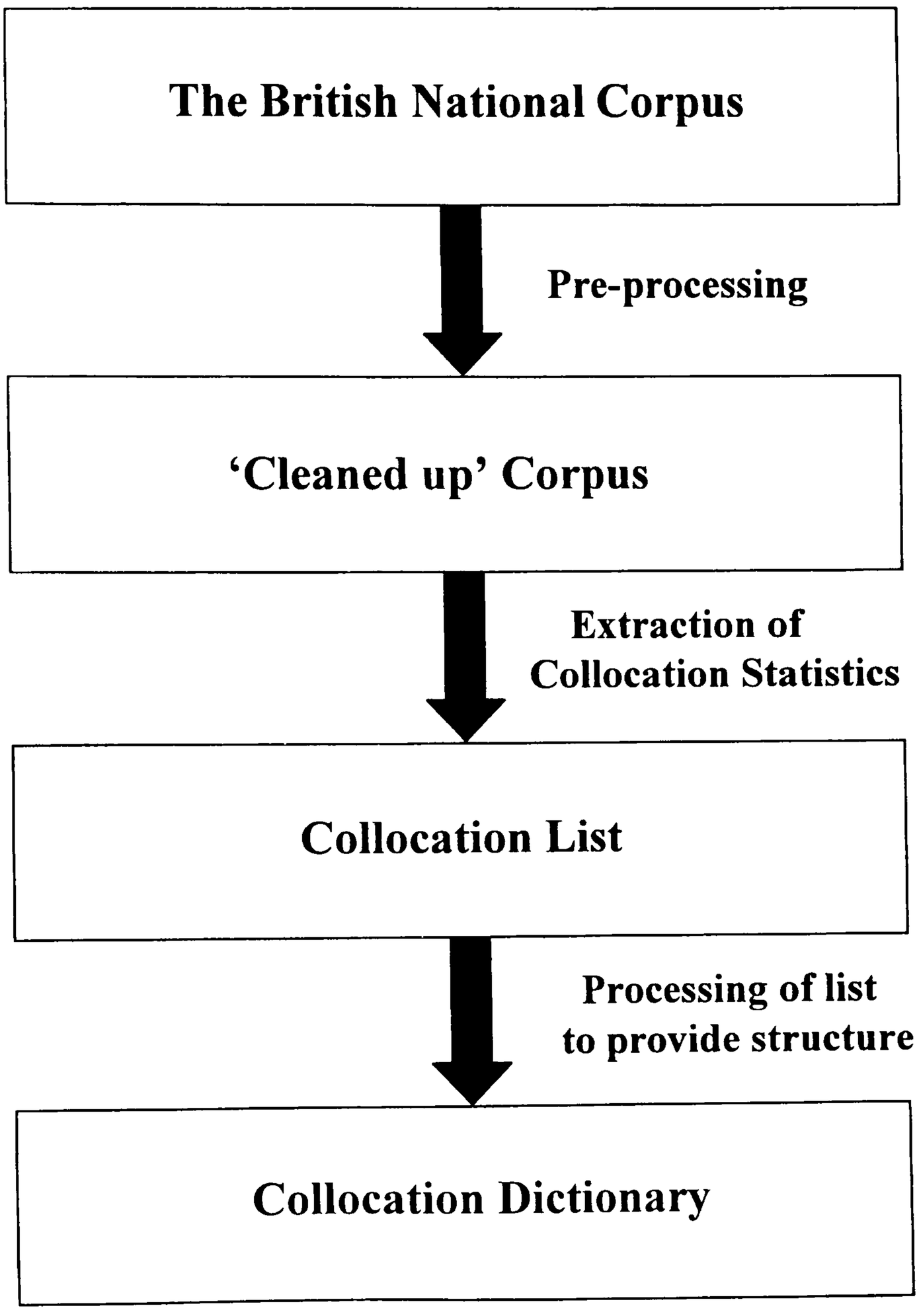

Fig. 4.7 - Steps in the creation of the collocation dictionary

4.2.2.1 - The British National Corpus

The British National Corpus (or BNC) is the result of a project carried out by an academic-industrial consortium from between 1991 and 1994, and version 1.0 was released in May 1995 (see Burnard, 1995).

The corpus is a body of text consisting of around 100 million words, about 80% of which is written language. For the purposes of this project, the 'corpus' referred to throughout is this body of written text. The corpus is also annotated with grammatical information, which was of no interest within the scope of this project.

A number of factors were taken into consideration when selecting the texts to be included in the corpus. These include :

- Domain (commerce / leisure / sciences etc.)
- Time (the date of publication of a text)
- Medium (book / periodical etc.)
- Author information (gender / age / nationality etc.)
- Target audience (child / adult etc.)
- Place of publication

In short, the BNC offers a large selection of widely varying types of text, and provides a valuable general snapshot of the nature of written language at this point in history.

It is therefore ideal (in terms of size and content) for extracting statistical information about the interaction of words for a collocation dictionary.

A reproduction of a section of the BNC can be found in **Appendix C**.

4.2.2.2 - Pre-processing the BNC

As mentioned above, the BNC, far from being a straightforward body of text, contains detailed grammatical information about the text.

The process of creating my collocation dictionary required merely a set of delimited sentences with no extraneous information whatsoever. Therefore a pre-processing step was required to 'clean up' the BNC.

Firstly, each BNC file contains a header giving information about the contents of that file. This header had to be stripped away from each file.

In the BNC text itself, each word is marked with a part-of-speech tag. So a typical sentence might look like :

<w PNP>It <w PNP>was <w ATO>the
<w NN1>sort <w PRF>of <w NN1>sight
—<w NN1-VVB>the <w AJO>poor
<c PUN>, <w ATO>the <w AJO>strange
&mdash <w NN1>which <w AVO>usually
<w VVD>alarmed <w NPO>Graham<c PUN>.

Stripping away the grammatical information gives the sentence :

It was the sort of sight - the poor, the strange
- which usually alarmed Graham.

A decision was made at this stage that the system would not be case-sensitive. Therefore all uppercase letters were converted to lowercase.

As the system deals with orthographic sentences, and not with smaller subdivisions into phrases, it was also decided to omit all punctuation apart from full stops.

Further complications were encountered in the original BNC text. For instance, the pound sign, rather than being represented symbolically is represented as **&pound** (this is called an *entity reference* in the BNC).

The pre-processing stage dealt with these entity references on a case by case basis, as necessary. For example an entry such as :

£100

would be rendered as :

100 pounds

So the example sentence given above would be filtered through the pre-processing stage and be rendered as :

it was the sort of sight the poor the strange

which usually alarmed graham .

See **Fig. 4.8**.

A file containing a set of sentences represented in this way forms the input to the collocation list creation stage.

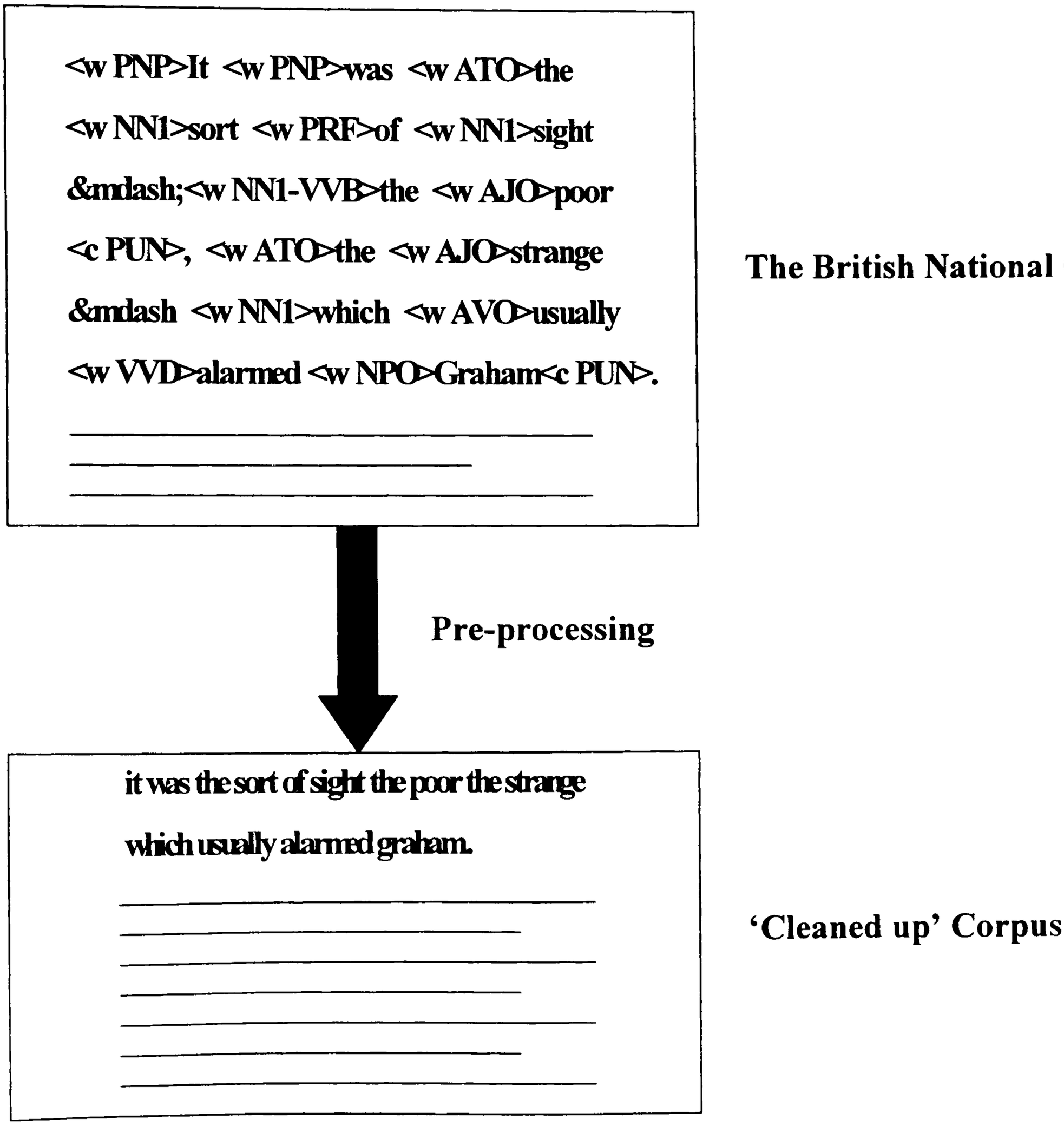

Fig. 4.8 – Pre-processing of the BNC

4.2.2.3 - The Creation of a Collocation List

The aim of this stage was a file with entries of the form :

Node Offset Collocate Offset Position

The fields **Node Offset** and **Collocate Offset** represent words in the input text. Words are actually represented as pointers to their entries in the lexicon. This is for ease of processing later (see the next section).

Due to the 'cascade' structure of the final collocation dictionary, the word represented by **Node Offset** is always alphabetically before the word represented by **Collocate Offset**. The first word is considered as the node in the collocation merely for the sake of convenience. Each entry in the list can equally be used to represent the collocational relationship between the two words when the second word is the node, and the first is the collocate.

The **Position** field gives the position in the sentence of the word represented by **Collocate Offset** in relation to the word represented by **Node Offset**.

Note that there is no need for a **Frequency** field at this stage of processing. As each collocation in the input text is represented individually, the frequency for each entry is by implication **1**.

It was important at this stage to decide on the *span* to be used when considering collocations. It was decided after reference to the literature that a span of **4** would be used (see Jones & Sinclair, 1974, Rose & Evett, 1992 etc). This means that word **a** is considered to collocate with word **b** if **a** occurs within four word positions of **b** in a sentence.

So when considering the collocations of a particular word, any process must deal with up to nine words at a time (the node and the four words on either side of it).

The practicalities of creating the collocation list involved a 'pipeline' of five words at a time. This is best explained using an example. Consider the input :

the boy stood on the burning deck .

The first five words in the sentence would be loaded into the pipeline :

the	**boy**	**stood**	**on**	**the**

Owing to the alphabetical structure of the collocation list, all the collocations in this sequence are considered in relation to the alphabetically earliest word, i.e. **boy**.

This word is selected as the node, and information is stored about its relationships with the other words in the sequence :

boy	**the**	**-1**
boy	**stood**	**1**
boy	**on**	**2**
boy	**the**	**3**

i.e. the word **the** appears immediately before **boy**, the word **stood** immediately after it etc.

It should be remembered that in the physical representation of this information, words are actually stored as pointers to their entries in the lexicon.

Now the next word alphabetically is considered, i.e.**on**. Information is stored only about its relationships with the words in the sequence thatcome alphabetically after it, as its relationship with the word**boy** has already been noted). So the following information is stored :

on	**the**	**-3**
on	**stood**	**-1**
on	**the**	**1**

This process continues until the alphabetically last word has been dealt with. In this case, there are two occurrences of **the**. It is important that the collocation relationship between these two occurrences is stored only once, i.e. :

the	**the**	**4**

as during processing the relationship :

the the -4

will be gleaned from this one entry, and need not be stored explicitly.

Now the next word can be fed into the word pipeline, and the leftmost word will be shunted out.

The pipeline will thus look like :

boy	**stood**	**on**	**the**	**burning**

The collocational relationships of this new word **burning** must now be stored. Its relationship with the word **boy** will be represented in relation to the word **boy**, as it is alphabetically earlier. Hence the first information stored will be :

boy	**burning**	**4**

The other relationships can now be stored. These are all in relation to the word **burning**, as that is next in line alphabetically after **boy** :

burning	**stood**	**-3**
burning	**on**	**-2**
burning	**the**	**-1**

Now the next word, **deck** can be fed into the pipeline :

stood	**on**	**the**	**burning**	**deck**

The following information is stored :

burning	**deck**	**1**
deck	**stood**	**-4**
deck	**on**	**-3**
deck	**the**	**-2**

When it is attempted to feed in the next word, a **.** is encountered, signifying the end of the sentence. So information about all the relationships between the words in this sentence has been stored (apart from the significance of these relationships, which comes at a later stage of processing).

All the sentences in the input file are processed in this manner, giving a list of word relationships that can be used in the next stage of processing (see **Fig. 4.9**).

This stage of processing is of course extremely time consuming. There are no real shortcuts available - the input must be ploughed through sequentially.

There is also a large filestore requirement. The collocation list for an input of just over 13 million words contained over **38,785,000** entries (occupying over **750Mb** of filestore).

An extract from the collocation list can be found in **Appendix D**, section **D . 1**.

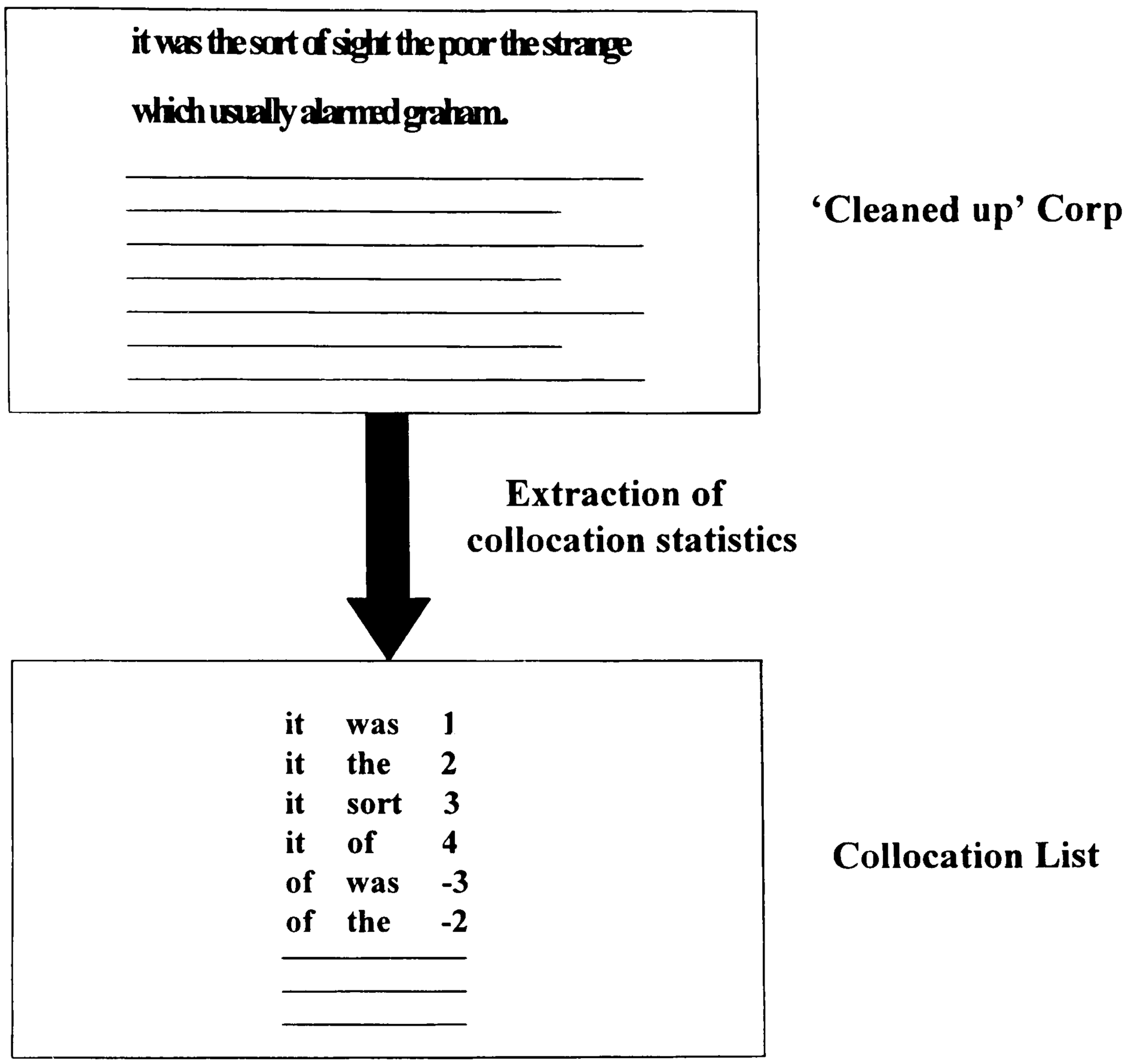

Fig. 4.9 – Creation of the collocation list

4 . 2 . 2 . 4 - The Creation of the Collocation Dictionary

The collocation list created by the process described in the previous section first had to be sorted to give an alphabetical list of collocational information as input to the next stage.

The aim of this stage of processing was to convert this large, unwieldy and, in its present form, largely meaningless list of information into a usable collocation dictionary.

The differences between the two versions of the dictionary have been discussed in some depth earlier, so in this section I will refer to the generic *strength* of a collocation, and not consider how this strength is represented.

The information in the collocation list is rationalised to give a more elegant representation (see **Fig. 4.10** overleaf).

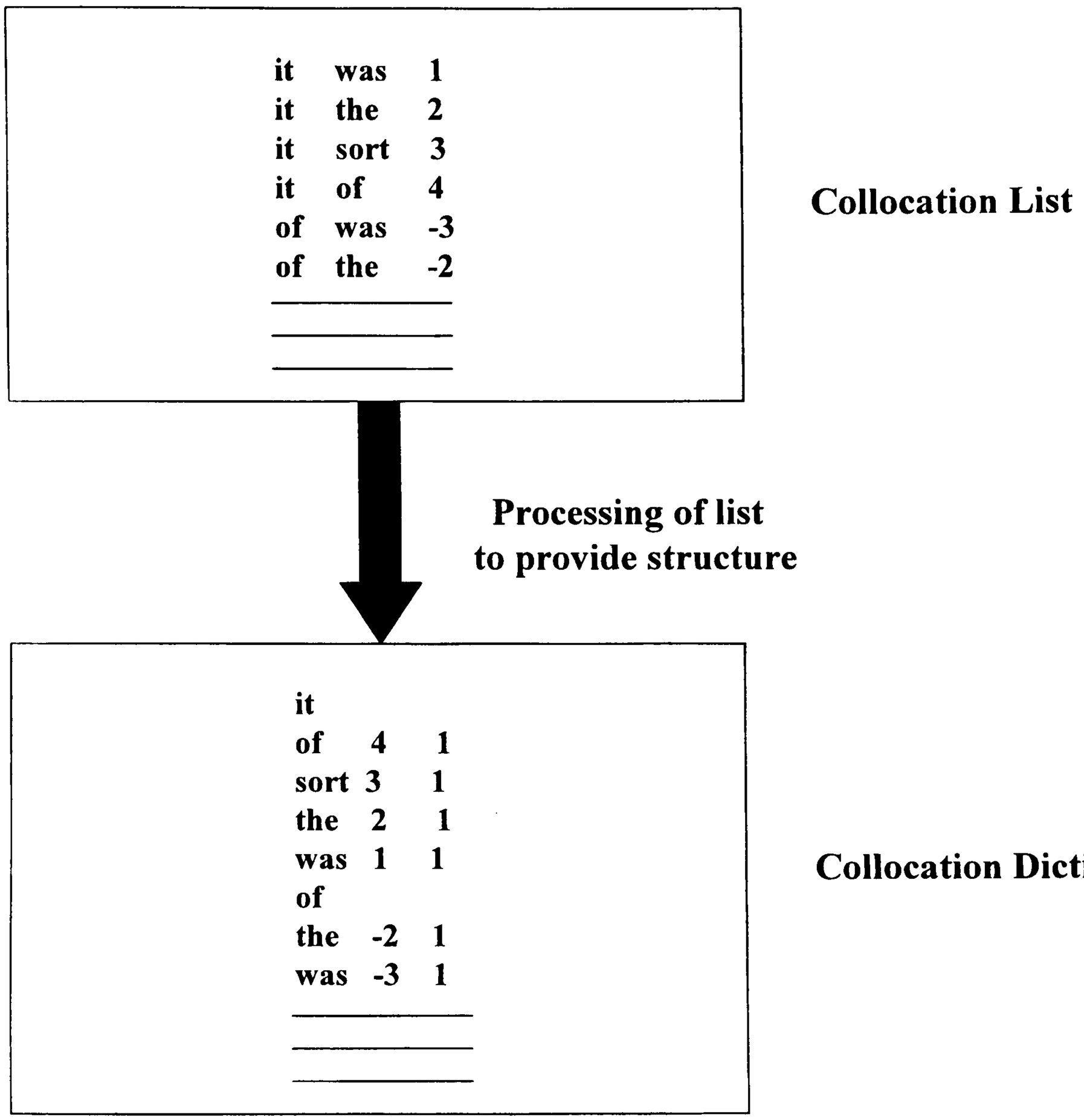

Fig. 4.10 – Creation of the collocation dictionary

So for instance, a passage in the collocation list :

a	**the**	**-3**
a	**the**	**-3**
a	**the**	**-3**
a	**the**	**4**
a	**them**	**-2**
a	**them**	**-2**
a	**them**	**1**
an	**the**	**-3**
an	**the**	**-3**
an	**the**	**-2**

would become :

a			
the	**-3**	**3**	***Strength***
the	**4**	**1**	***Strength***
them	**-2**	**2**	***Strength***
them	**1**	**1**	***Strength***
an			
the	**-3**	**2**	***Strength***
the	**-2**	**1**	***Strength***

in the Percentage Score Dictionary in which position information is stored, and :

a		
the	**4**	***Strength***
them	**3**	***Strength***
an		
the	**3**	***Strength***

in the Z-score Dictionary, which does not take position into account.

As stated previously, words in the collocation dictionary are stored as lexicon entry numbers. This is more efficient of space than an explicit representation.

Words in the collocation list created in the previous stage of processing were stored as pointers to word entries in the lexicon. This was to speed the process of calculating the strength of the collocational relationships being stored in the collocation dictionary.

To calculate both percentage scores and z-scores, the number of times both the node and the collocate occur independently in the input text is required. This information is stored in the lexicon entry for each word (see **4 . 1 . 1 - The Lexicon Structure**).

Storing each word as a pointer to its lexicon word entry allows a direct jump to the correct place in the lexicon during processing.

As a comparison with the raw collocation list, the Percentage Score Dictionary contains **9,942,446** lines (including entry headings) occupying around **288Mb** of filestore, while the Z-score Dictionary contains**2,079,124** lines, occupying around **49Mb** of filestore.

Extracts from the Percentage Score Dictionary and the Z-score Dictionary can be found in **Appendix D**, sections **D.1** and **D.2** respectively.

Chapter 5

Text Recognition Using Collocational Analysis

This chapter describes the stages of processing carried out in filtering an input word lattice to remove invalid words, and then choosing the best path through that lattice based on analysis of the collocational relationships contained within it.

5 . 1 - Pre-processing - the Construction of a Valid Word Lattice

In practical terms, this stage is carried out by the low-level output simulator described in **Chapter 3**.

In a system using actual low-level recogniser output however, this stage would be seen as a filter through which the low-level output would pass before the collocational analysis stage.

This filter carries out two processes :

- the removal of invalid words from the word lattice.

- the reduction of the number of valid words at each word position.

5 . 1 . 1 - The Removal of Invalid Words from the Word Lattice

This process must be carried out first on the words initially presented as input to the low-level output simulator, and then on the alternatives to these words produced by the simulator.

If one of the simulated alternatives is found to be invalid, then it can simply be discarded from the final word lattice.

However, if an original input word is found to be invalid, it must still be passed to the collocation processor, as it occupies a particular position in the sentence relative to other words which may be valid. This positional information is vital to the operation of the collocation processor.

In practice, such an invalid word will be passed to the collocation processor as a 'dummy' word, giving no information other than what position it occupies in the input sentence.

First, I must define what I mean by a valid word. To be considered valid by the system, a word (or at least the root form of a word) must be present in the system lexicon *and* in the version of the collocation dictionary in use at the time. Both of these conditions are checked by consulting the lexicon.

Before the full lexicon needs to be checked however, there are two other sources which can be consulted to check whether a word is present in the lexicon :

- the **jump matrix**
- the **illegal letter pair matrix**

As described in Chapter 4, the **jump matrix** is a 26 * 26 matrix containing the location in the lexicon of the first word beginning with each letter pair. So for instance, the first word beginning with**aa** is stored in entry **[0][0]**, the first word beginning with **ab** in **[0][1]** and so on. If there is no word in the lexicon beginning with a particular letter pair, then a-**1** is stored in that pair's entry.

So the first step in checking whether a word is present in the lexicon is to take its first two letters and consult the relevant entry in the jump matrix. If the entry is **-1** then the word is not present and can be rejected as invalid without consulting the full lexicon.

The **illegal letter pair matrix** is organised along similar lines, but refers to all letter pairs in the lexicon and not just those at the beginning of a word.

The matrix is initialised so that each entry contains a **-1**. Then the entire lexicon is scanned. If a pair of letters appears consecutively in a word in the lexicon, then a **1** is written to the relevant entry in the illegal letter pairs matrix.

At the end of the lexicon scan, if an entry in the matrix contains a **-1**, then the corresponding consecutive letter pair never occurs in any word in the lexicon. This knowledge can be used when the low-level simulator is constructing the candidate word lattice.

Each candidate word is produced by traversing a letter lattice (see **Chapter 3**). Traversing the letter lattice from a given letter position to the next letter position gives a letter pair. The entry for this pair in the illegal letter pair matrix is checked, and if it contains a **-1**, then any path through the letter matrix containing that traversal letter pair can be rejected as invalid.

These two methods are shortcuts to restrict the number of lexicon look-ups required, but many still have to be made.

If it is established that a word begins with a legal letter pair, and contains no illegal consecutive letter pairs, then this word must be looked up in the full lexicon, using the look-up method described in**Chapter 4**. If it is found in the lexicon, then it has passed the first test of validity.

The word must now overcome the second hurdle, i.e. does it occur in the collocation dictionary? To check the collocation dictionary itself would be extremely time consuming, but because of the structure of the lexicon, this is not necessary.

It will be recalled from **Chapter 4** that each lexicon entry is of the form :

word name

word number

word frequency

pointer to collocation dictionary

and that the **word frequency** field represents the number of times that the word occurs in the text used to construct the collocation dictionary.

Clearly, if this value is **0**, then the word will not be present in the collocation dictionary.

The converse is not necessarily true. In the case of the Percentage Score Dictionary, if the **word frequency** field is greater than **0**, then that word *is* in the collocation dictionary, but this may not be true in the case of the Z-score Dictionary. If the **word frequency** field is greater than **0**, then the word does occur in the text used to create the collocation dictionary, but it may not be involved in a *significant* collocation, and therefore will not be included in the Z-score Dictionary.

However, as the only alternative would be to search the Z-score Dictionary for the word, then this test of the **word frequency** field is assumed to be an adequate, if not perfect, test of validity.

So, using lexicon look-ups, the validity of the input words, and each of the candidate words produced by the low-level simulator has been checked, with invalid words being filtered out.

It was found to be necessary to restrict the number of lexicon look-ups during the construction of the candidate lattice by the low-level simulator. For each word position in an input sentence, a maximum of 500 candidate words were produced, which were then checked for validity. This number was found with testing to produce a satisfactory number of valid candidate words, without placing an undue burden on the system.

Once the lattice of valid words is thus produced, it is now necessary to restrict the number of valid words present at each word position. The criteria under which this process is carried out is described in the next section.

Fig. 5.1 shows the steps involved in the removal of invalid words from a word lattice.

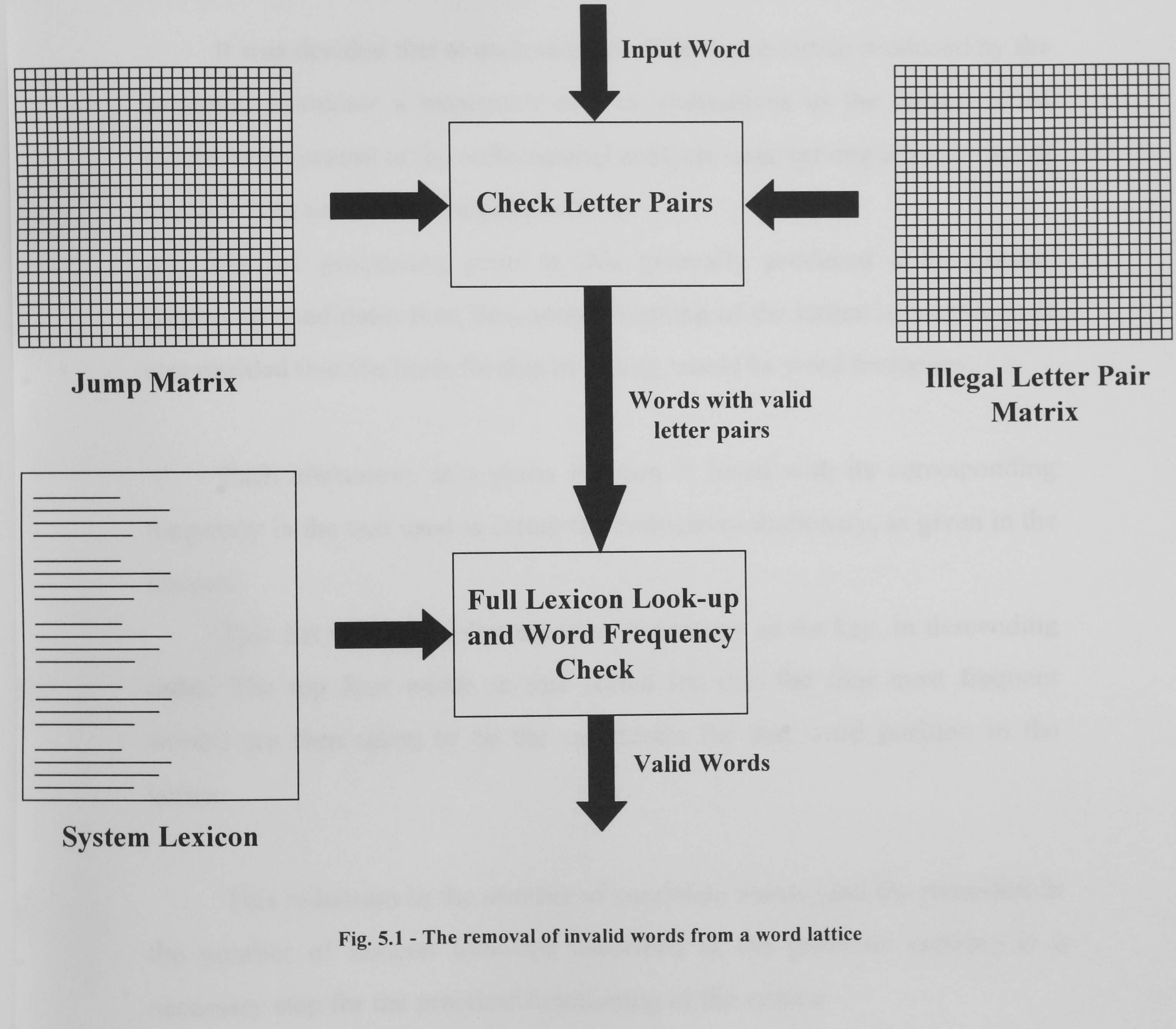

Fig. 5.1 - The removal of invalid words from a word lattice

5 . 1 . 2 - Trimming the Word Lattice

It was decided that at each word position in the lattice produced by the low-level simulator a maximum of four alternatives to the correct word should be presented to the collocational analysis stage (giving a maximum of five words in total at each word position).

As the processing prior to this generally produced a far greater number of candidates than this, some trimming of the lattice is necessary. It was decided that the basis for this trimming would be word frequency.

Each alternative at a given position is listed with its corresponding frequency in the text used to create the collocation dictionary, as given in the lexicon.

This list is sorted using the word frequency as the key, in descending order. The top four words in this sorted list (i.e. the four most frequent words) are then taken to be the candidates for that word position in the lattice.

This reduction in the number of candidate words (and the reduction in the number of lexicon look-ups described in the previous section) is a necessary step for the practical functioning of the system.

Most low-level recognisers produce some sort of score for each word in the lattice that they produce, indicating the probability that that is the correct word in the position.

If input was provided by an actual low-level recogniser then these scores could be used as the criteria by which the word lattice is trimmed, which would be a much more satisfactory solution.

However, as the input to this system is simulated, it was necessary to find some other means of making the input manageable, and basing this on word frequency seems to be a sensible solution to the problem.

At the end of the processing steps described above, we are left with a trimmed lattice of valid words to present to the collocational analysis stage.

For ease of processing, the words in this lattice are each represented as a number pair :

pointer to collocation dictionary	**word number**

This number pair is mapped to the word itself after processing is complete.

5 . 1 . 3 - An Example of the Pre-processing of an Input Sentence

The steps described in the previous two sections are best illustrated by use of an example. Consider the sentence :

the cat sat

The low-level recogniser simulator will take the word **the**, and first of all check its validity. Of course it is valid, but if it wasn't the following processing would not take place for this word. Instead, it would be written directly to the final word lattice as a 'dummy' word. As **the** *is* a valid word, the low-level simulator, using the information stored in the letter substitution database, will produce the letter lattice shown in **Fig. 5.2**.

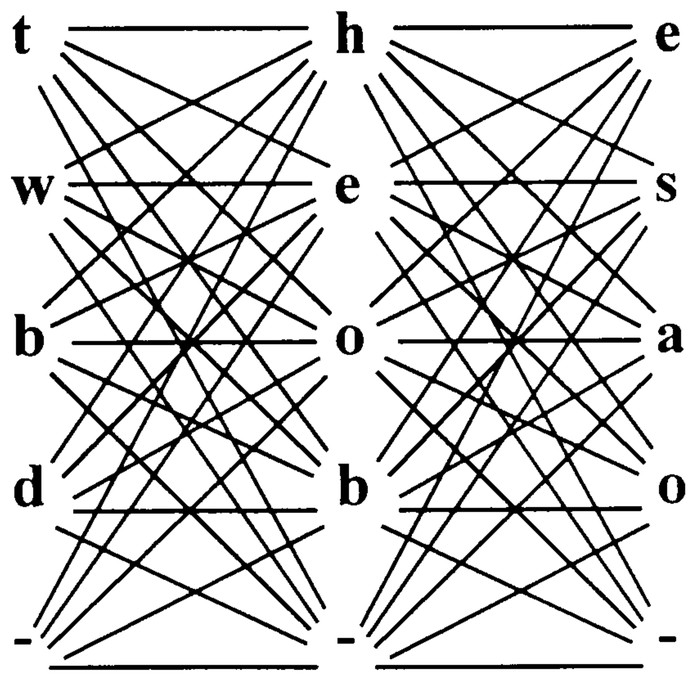

Fig. 5.2 - The letter lattice produced for the word 'the'

The '-' characters are special characters denoting that that letter position may be left blank.

This letter lattice is now traversed to produce a list of candidate words for the word **the**.

This list will contain 125 (5^3) lines, including the following :

the

ths

tha

tho

th-

.

.

.

--e

--s

--a

--o

Next, the first word in the list (i.e. the correct word) is written to the final word lattice. Using the remaining words, a look-up file is created, from which are removed all the words containing illegal letter pairs (as stored in the **jump** matrix and the **illegal letter pair matrix**). If there were more than **500** words in the list above, this would be truncated. All the special characters, '-' are also discarded at this stage.

The words in the look-up file are checked by consulting the lexicon, and all invalid words are removed. The list of valid words is then written to a file, along with each word's frequency of occurrence, the pointer to the corresponding entry in the collocation dictionary, and the word number, as given in the lexicon.

This file is sorted on the frequency field to give the following list of **62** items :

Frequency	Word	Pointer	Word no.
355226	**to**	**284705782**	**70430**
306410	**a**	**0**	**1**
204593	**be**	**50404856**	**6081**
84004	**he**	**180741109**	**31108**
.	.	.	.
.	.	.	.
.	.	.	.
1	**dh**	**121671756**	**18561**

The top four words in this list are taken to be the alternatives to the word **the**.

The other words in the input sentence are now processed in the same way, to produce the final word lattice to be presented for collocational analysis, as shown in **Fig. 5.3**.

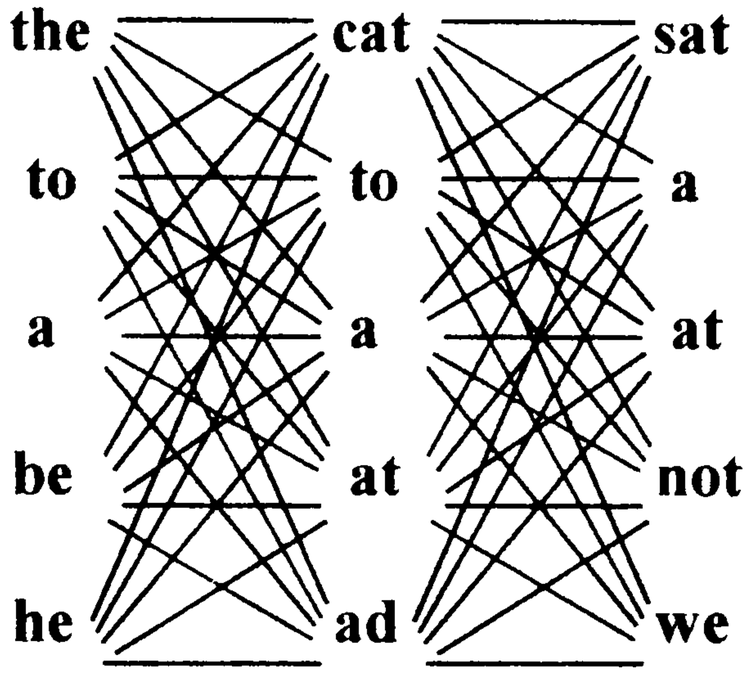

Fig. 5.3 - The word lattice produced for the sentence 'the cat sat'

This lattice would of course be different if the low-level recogniser simulation was based on a different set of example lattices. The example lattices used demonstrate a great amount of conflation (i.e. the candidate words are shorter than the input words) and very little expansion (the candidate words are longer than the output words).

This is reflected in the lattice above, in that the candidate words produced are the same length as, or shorter than, the input word.

This lattice can now be passed on to the next stage, which analyses the collocational relationships contained within it and suggests the best path through it based on this analysis.

5 . 2 - The Collocational Analysis of a Word Lattice

After the pre-processing phase we have a word lattice consisting of the input sentence plus four valid candidate words for each word position in that sentence.

The one exception to this is when an input word is not a valid word, in which case that word is represented on its own with no candidates, and is marked as a 'dummy' word. The only part that such a dummy word plays in the collocational analysis is to provide positional information.

The lattice is analysed using a word 'pipeline' similar to that used in the creation of the collocation dictionary.

The practical operation of this pipeline is described in the next section. Following this, the processing of an example lattice will illustrate this operation.

5 . 2 . 1 - The Collocation Pipeline

As described previously, each word in the lattice is represented by a number pair consisting of a pointer to that word's entry in the collocation dictionary (if any) and the word number as stored in the lexicon.

It is possible that a valid word will not have its own entry as a node in the collocation dictionary. This is in the case of a word collocating only with words that are alphabetically before it, in which case these collocations will be represented in the entries for the other words.

Initially, the first five word positions in the lattice are considered. Assuming that for each word position there is an input word plus four candidates, this gives **3125** ($\mathbf{5^5}$) potential paths through the lattice. Each of these is analysed in sequential order.

As in the creation of the collocation dictionary, the words are considered in alphabetical order, so for the path being analysed, the first word alphabetically is identified, and treated as the **node**.

As each word is stored partly in terms of the location of its entry in the collocation dictionary, this can be accessed directly. Once the entry for the node is located, it is searched sequentially for the word numbers of the other words in the sequence (the collocates).

If the Percentage Score Dictionary is being used, then the positions of the collocates in relation to the node are also taken into account.

If a match is found then the collocational score between the node word and that collocate is calculated. In each version of the collocation dictionary, the strength of the collocation is represented by two values (taking account of upward and downward collocation). The collocational score of a relationship is calculated by multiplying these two values together. This is done for each collocate found in the search of the node word's entry in the dictionary, and the scores added together.

This process is repeated for each word in the path until all the scores have been calculated and added up. We now have an overall collocational score for the path under consideration.

When the above process is carried out for each path the scores for each path are compared to find the highest.

The first word of the path with the highest score is assigned as the system's hypothesis for the correct word in the first word position.

This decision can be made at this stage because that word position will play no further direct part in the collocational analysis, as words later in the sentence fall outside its collocational span.

The next word can now be fed into the pipeline, and the candidates in the first word position now disappear from the pipeline. Another set of paths is now created, but the collocation scores from the previous calculation are retained.

This is crucial, as the collocational influence of the word in position number one on the next four word positions must be retained and considered in conjunction with the relationships between these words and the new word in the pipeline.

The process is now repeated, with the collocation score for each path being added to by the influence of the new word. Once all the scores have been worked out, then the path with the highest score provides us with the system's hypothesis for word position two.

This continues until the last word position in the lattice has been reached, and therefore a hypothesis has been proposed for each word position in the lattice.

These hypotheses can then be compared with the initial input sentence, to give a measure of success.

Figs. 5.4 and **5.5** shows the collocational analysis process described above in the form of a flowchart.

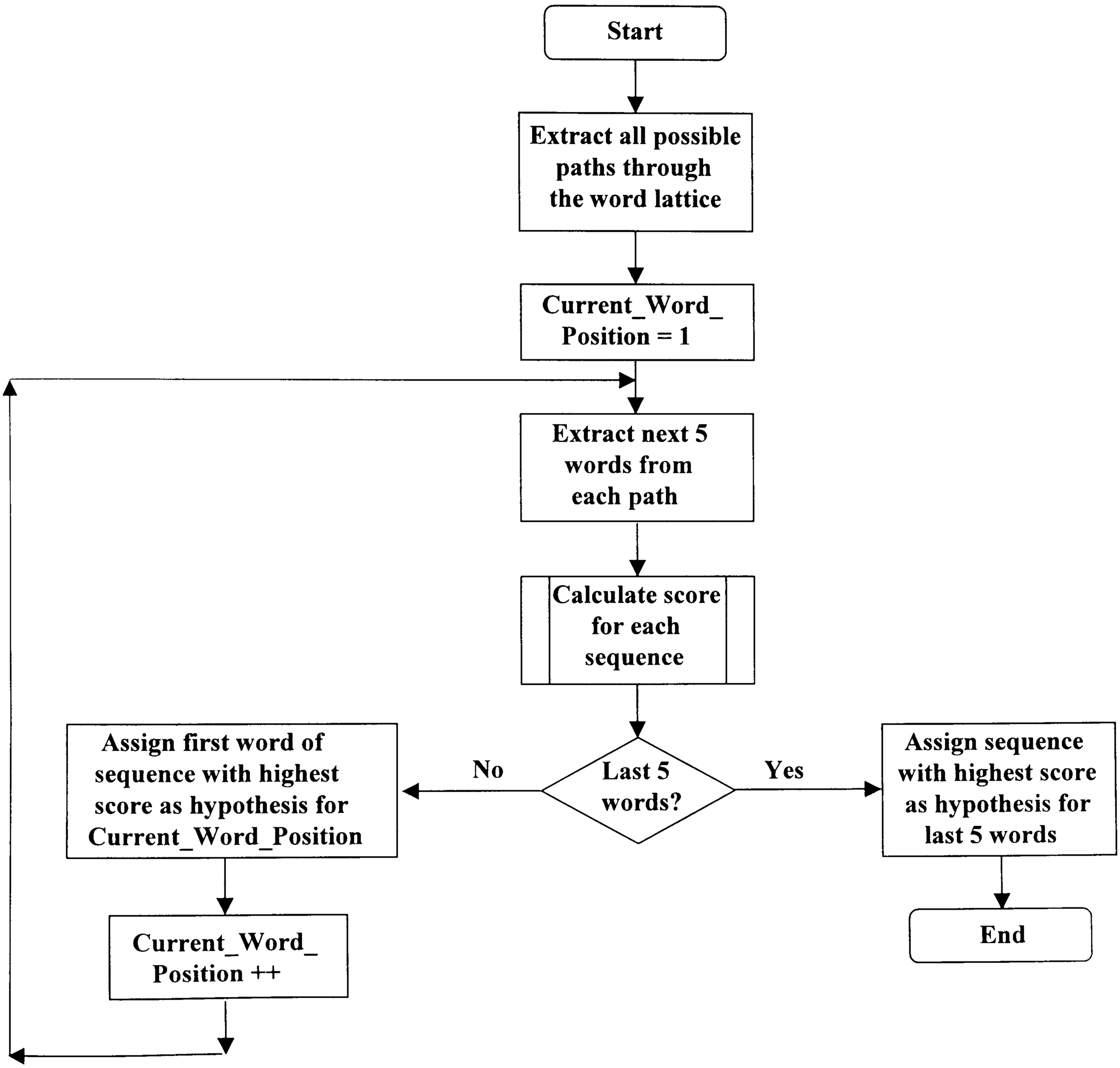

Fig. 5.4 - Flowchart showing the collocational analysis process

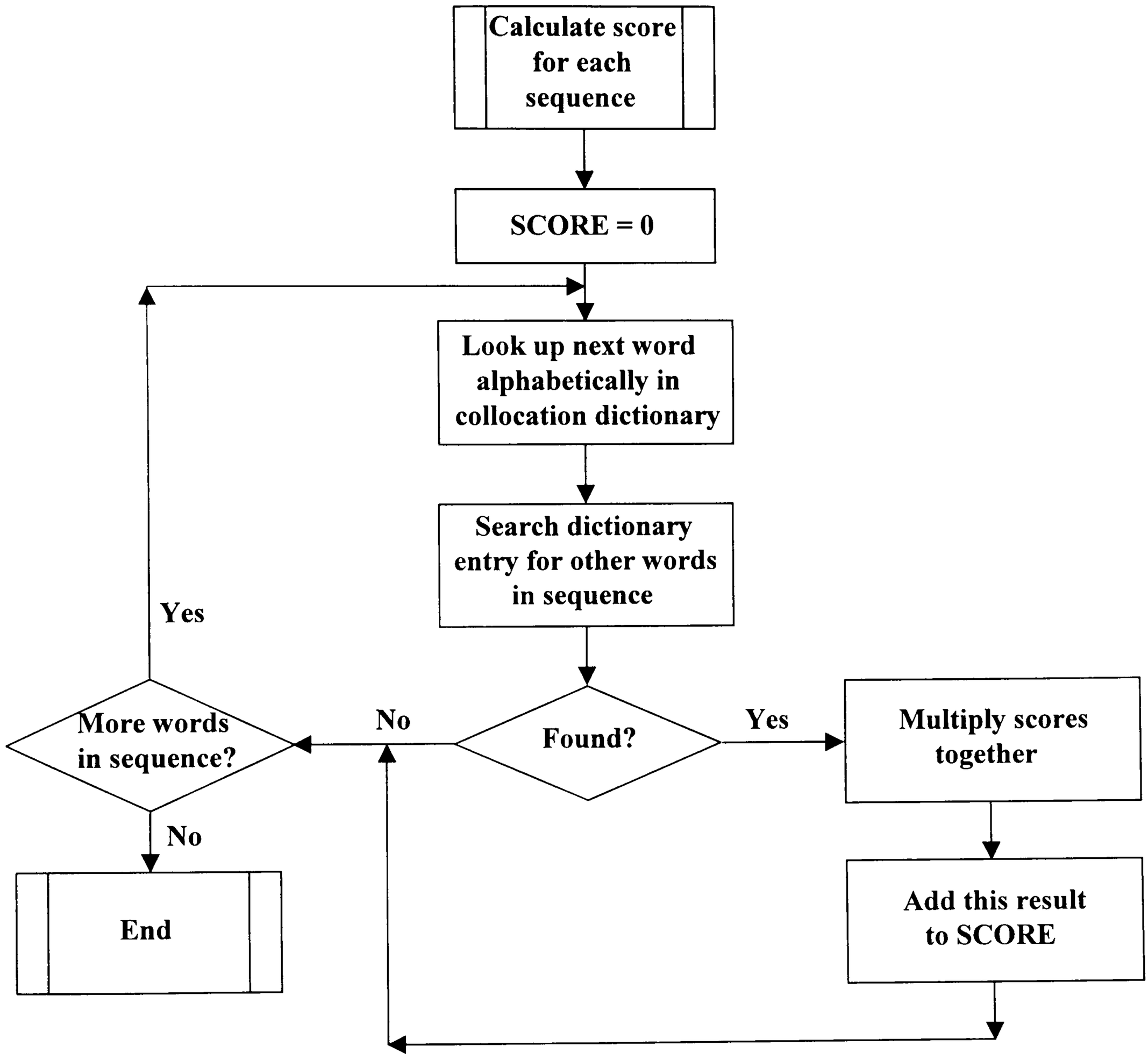

Fig. 5.5 - Flowchart showing the process for calculating collocation scores for a word sequence

5 . 2 . 2 - An Example of the Collocational Analysis of a Word Lattice

Consider the sentence :

the boy stood on the burning deck

This input sentence will produce the word lattice shown in **Fig. 5.6**.

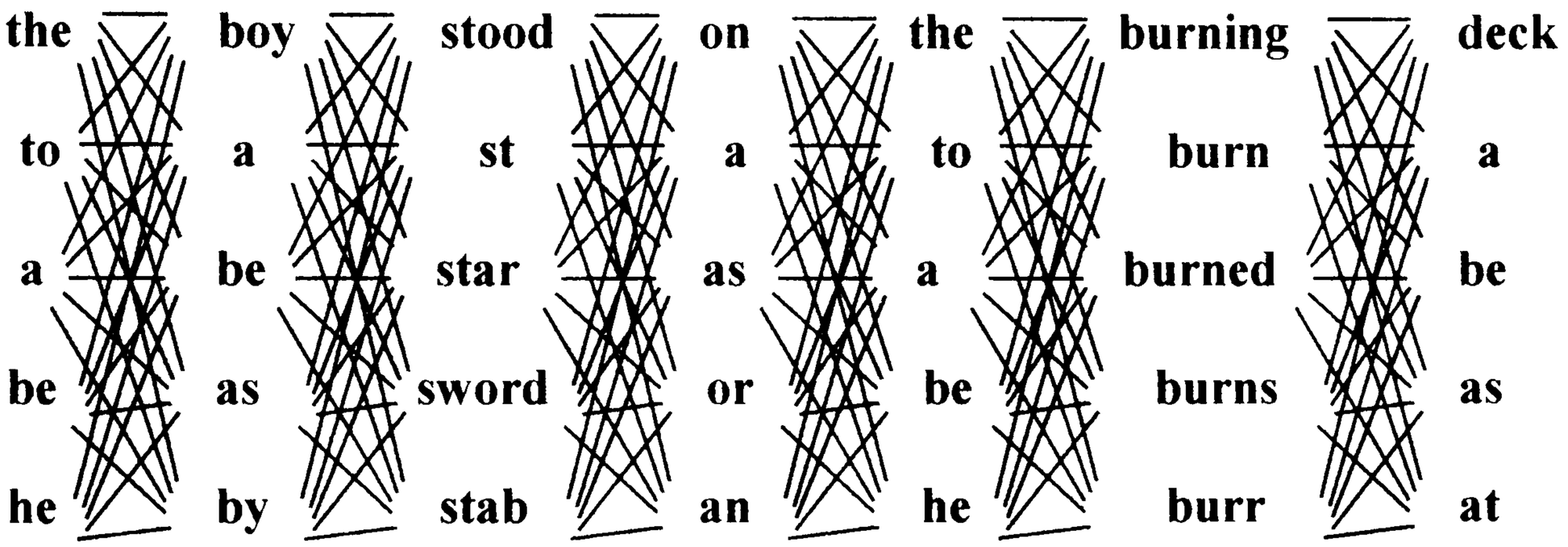

Fig. 5.6 - Word lattice produced for the sentence 'the boy stood on the burning deck'

First, all the possible paths through the lattice for the first five word positions are extracted :

the boy stood on the
the boy stood on to
the boy stood on a
. . . . .
. . . . .
. . . . .
he by stab an a
he by stab an be
he by stab an he

Taking the first path :

the boy stood on the

the word **boy** is alphabetically first, so this word is selected as the node. The position of its entry in the collocation dictionary is found, and the dictionary is accessed at that point. For this example we will assume that the Percentage Score Dictionary is being used.

Within the collocation dictionary entry, we must search for following :

word no. for *the* **-1**

word no. for *stood* **1**

word no. for *on* **2**

word no. for *the* **3**

If and when these are found, the two percentage scores attached to them are multiplied together, and added to the cumulative collocation score for the path under consideration.

When scores have been calculated for all the paths, we must find the highest one. Let's say that the highest score was calculated for path number two :

the boy stood on to

The first word of this path, **the** is put forward as the hypothesis for word position number one of the input sentence. In this case, as in all good examples, it is correct!

This word position now drops out of the pipeline, and the next word position is fed in, giving a whole new set of paths, although the cumulative collocation scores are retained from the previous loop of processing as explained previously.

The new set of paths will look like :

boy stood on the burning

boy stood on the burn

boy stood on the burned

. . . . .

. . . . .

. . . . .

by stab an he burned

by stab an he burns

by stab an he burr

The same loop of processing as before is now carried out on this set of paths to produce a hypothesis for word position number two.

This repeats until the final five word positions of a path are being processed. Once the cumulative score for these five word positions for each path has been calculated, the entire five words of the path with the highest cumulative scored are put forward as the hypotheses for those five word positions.

The next input sentence can now be processed.

Chapter 6

Description of Experiments

A number of experiments were carried out to test the effectiveness of the system described in the previous chapters.

This chapter describes these experiments and records their results. The implications of these results are discussed in **Chapter 7 – Analysis of Results**.

During experimentation there were three variable entities :

- the input text
- the version of the collocation dictionary
- the lexicon

As for the first entity, there were two sources of input text used : the Susanne Corpus and the British National Corpus.

As has been described, the collocation knowledge base was created using part of the British National Corpus (the BNC) as input.

The experiments using the Susanne Corpus are intended to test the generality of the collocation knowledge base. In other words, the system is attempting to recognise input about which it has no direct information.

The hope is that the information provided by the BNC during the creation of the knowledge base gives a general view of the collocational relationships that occur in the English language in general, and can therefore be applied to any input text.

Carrying out experiments using sentences from the BNC as input is intended to give a benchmark against which to compare the results attained using the Susanne Corpus.

The knowledge base contains direct information about the collocational relationships within a section of the BNC, and therefore should perform with a high level of accuracy when analysing sentences from that section of the BNC. This use as input of a text with which the post-processing system is familiar can be seen as analogous to the operation of a natural language recognition system (particularly a speech recognition system) being used having first been trained to recognise the language of a particular user.

How well the system performs when analysing sentences from the Susanne Corpus (i.e. input text for which the system has not been 'trained') will give an indication as to the generality of the collocation knowledge base.

The sentences used as input to the experiments from both the BNC and the Susanne Corpus are listed in **Appendix F**.

The second variable in the experiments is the version of the collocation dictionary used in the analysis of the input text.

There is a choice of the **Percentage Score** Dictionary and the **Z-score Dictionary**. The nature of the two versions of the collocation dictionary available to the system is described fully in **Chapter 4**.

Briefly, they differ in the way that they represent the strength of a collocational relationship. Comparing the recognition accuracy of the system using one version of the collocation dictionary against its performance when using the other will give an indication as to which representation of collocational relationships is most suitable for this application.

The third and final variable in the experiments is the lexicon used by the system. Changing the lexicon is much more fundamental than it at first appears.

The lexicon and the collocation dictionary are inextricably linked in that the collocation dictionary contains only words that feature in the lexicon.

To change the lexicon therefore is to change the collocation knowledge base.

A number of experiments were carried out using a wholly different (and considerably smaller) collocation knowledge base from that created using the BNC.

The design and creation of this knowledge base is described in

6 . 3 - A Tailored Knowledge Base.

The experiments on the main collocation knowledge based using input from the BNC are described in section **6 . 1**.

The experiments using input from the Susanne Corpus are described in section **6 . 2**.

6 . 1 - Experiments with the British National Corpus

As explained previously, these experiments were carried out chiefly to set a benchmark against which to measure the results of other experiments.

The collocation knowledge base used by the system was compiled using a section of the BNC and a lexicon derived from the Collins Dictionary.

Therefore, these experiments give an indication of how well collocational analysis performs with a dedicated collocation dictionary but with a general purpose lexicon.

It would be expected that using test data from the same source as was used to create the knowledge base would produce a high level of accuracy.

The experiments are described below.

6 . 1 . 1 - Experiments Using the Percentage Score Dictionary

Fifty sentences containing a total of**338** words were selected from the same subset of the BNC that was used to create the collocation knowledge base.

The only criterion used in selecting sentences as suitable for testing throughout all the experiments was the number of words contained in those sentences.

This was a purely practical measure, as a sentence containing many words would take considerably longer to process than a shorter sentence. With this in mind, sentences containing more than fifteen words were rejected.

There was emphatically no selection based on the language content of the sentence.

The selected sentences were passed to the system for processing, and the results produced are shown in the tables below :

Number Recognised	Number Mis-recognised	
	20	
318	Not in Lexicon	Wrong Word
	9	11

Percentage Recognised	Percentage Mis-recognised	
	5.92%	
94.08%	Not in Lexicon	Wrong Word
	2.66%	3.26%

The first table gives the results of the experiment in terms of the numbers of words recognised or mis-recognised. The second table gives the same results in terms of percentages.

The **Percentage Recognised** column gives the percentage of the words in the input text that were correctly recognised using collocational analysis on simulated word lattices generated from the input text. A word is considered to have been correctly recognised if that word (or its root form in the case of a derivation) is offered as the system's hypothesis.

The **Percentage Mis-Recognised** column gives the percentage of the words in the input text for which an incorrect hypothesis was offered by the system.

This category is sub-divided into input words that are not in the lexicon and are therefore considered invalid by the system, and input words that are in the lexicon but were incorrectly identified by the system.

These results are fully analysed in **Chapter 7**.

6 . 1 . 2 - Experiments Using the Z-score Dictionary

One hundred sentences containing a total of **700** words were selected from the subset of the BNC used to create the collocation knowledge base, and fed to the system for processing.

More input sentences were used in the experiments involving the Z-score Dictionary for purely practical reasons.

As explained in **Chapter 4**, the Percentage Score Dictionary represents *all* the collocations (within the defined span of 4) contained in the selected subset of the BNC, whereas the Z-score Dictionary represents only those collocations considered to be significant according to the z-score measure.

Consequently the Z-score Dictionary is considerably smaller than the Percentage Score Dictionary (in fact it is approximately one-sixth of the size).

It follows from this that the processing of sentences can be carried out much more quickly when the Z-score Dictionary is used as the collocation knowledge base.

Proceeding on the basis that the larger the set of input data the more accurate the results produced, it was decided to double the number of sentences used for testing, simply because it was practically feasible to do so.

Such practical feasibility issues are discussed in **Chapter 8**.

The tables below show the results produced by processing the one hundred sentences :

Number Recognised	Number Mis-recognised	
	142	
558	Not in Lexicon	Wrong Word
	30	112

Percentage Recognised	Percentage Mis-recognised	
	20.29%	
79.71%	Not in Lexicon	Wrong Word
	4.29%	16%

These results are analysed fully in **Chapter 7**.

6 . 2 - Experiments with the Susanne Corpus

6 . 2 . 1 - The Susanne Corpus

The first release of the Susanne Corpus became available in 1992. My experiments use sentences from release 3, created in 1994.

A full description of the construction of the Susanne Corpus can be found in Sampson, (1994).

Briefly, the Susanne Corpus is based on a subset of approximately 130,000 words of the Brown Corpus of American English (see Ellegard, 1978).

Like the British National Corpus, the Susanne Corpus is annotated with detailed grammatical information.

This is clearly of no use for the purposes of this study, so a pre-processing step was required to leave a body of plain English.

A 'before' and 'after' view of a small section of the corpus can be found in **Appendix E**.

I selected my set of input sentences from section A of the corpus which contains examples of press reportage.

It was thought that this rather specialised form of language would provide a reasonable test of the generality of the collocation knowledge base as it was not 'trained' explicitly to recognise this style of writing (although examples of journalism are found in the section of the BNC used to create the knowledge base).

The texts found in section A of the Susanne Corpus are drawn from a variety of American Newspapers, including *The New York Times*, *The Chicago Daily Tribune*, as well as a number of smaller publications.

6 . 2 . 2 - Experiments Using the Percentage Score Dictionary

As in the experiments using the BNC, fifty sentences containing a total of **404** words were selected from the Susanne Corpus and fed to the system for processing. The tables below show the results produced by processing these sentences :

Number Recognised	Number Mis-recognised	
	118	
286	Not in Lexicon	Wrong Word
	27	**91**

Percentage Recognised	Percentage Mis-recognised	
	29.21%	
70.79%	Not in Lexicon	Wrong Word
	6.68%	**22.53%**

These results are analysed fully in **Chapter** 7.

6 . 2 . 3 - Experiments Using the Z-score Dictionary

As in the experiments with the BNC, one hundred sentences containing a total of **796** words were selected from the Susanne Corpus and fed to the system for processing. The tables below show the results produced by processing these sentences :

Number Recognised	Number Mis-recognised	
	318	
478	**Not in Lexicon**	**Wrong Word**
	54	**264**

Percentage Recognised	Percentage Mis-recognised	
	39.95%	
60.05%	**Not in Lexicon**	**Wrong Word**
	6.78%	**33.17%**

These results are analysed fully in **Chapter** 7.

6 . 3 - A Tailored Knowledge Base

It was decided to carry out a number of experiments using a collocation knowledge base tailored for a specific input text.

Section A of the Susanne Corpus, described previously, was used as the basis for this knowledge base.

The lexicon consisted of all the words contained in section A of the corpus, and all the collocations within a defined span of 4 were extracted from the text and stored in a collocation list, as described in **Chapter 4**.

From this collocation list was derived a Percentage Score Dictionary and a Z-score Dictionary (also described in **Chapter 4**).

All the components of the knowledge base were considerably smaller than those which make up the main collocation knowledge base :

- Lexicon : **5722** words, **188** Kbytes
- Collocation List : **110433** lines, **1.9** Mbytes
- Percentage Score Dictionary : **91676** lines, **1.8** Mbytes
- Z-score Dictionary : **53713** lines, **1** Mbyte

This tailored system was then tested using sentences from section A of the Susanne Corpus.

6 . 3 . 1 - Experiments Using the Percentage Score Dictionary

One hundred sentences containing a total of **796** words were selected from section A of the Susanne corpus and passed to the system to be processed using the specially tailored collocation database.

The results produced are shown in the tables below :

Number Recognised	Number Mis-recognised
755	**41**

Percentage Recognised	Percentage Mis-recognised
94.85%	**5.15%**

Note that there is no longer any need to sub-divide the **Mis-recognised** columns, as the input words are guaranteed to be in the system lexicon, as it is based on a superset of the input text.

These results are fully analysed in **Chapter 7**.

6 . 3 . 2 - Experiments Using the Z-score Dictionary

One hundred sentences containing a total of **796** words were selected from section A of the Susanne corpus and passed to the system to be processed using the specially tailored collocation database.

The results produced are shown in the table below :

Number Recognised	Number Mis-recognised
646	**150**

Percentage Recognised	Percentage Mis-recognised
81.16%	**18.84%**

These results are fully analysed in **Chapter 7**.

Chapter 7

Analysis of Results

This chapter is structured along the same lines as **Chapter 6**, in that it will examine the experimental results by contrasting the performance of the system when various conditions were in place.

Section **7.1** will look at the results of experiments using the BNC as input in contrast to those obtained when using the Susanne Corpus.

Section **7.2** will compare the results obtained using the Percentage Score Dictionary with those obtained using the Z-score Dictionary.

Section **7.3** will examine the results obtained using the specially tailored knowledge base.

Section **7.4** will summarise all these findings and attempt to extract a number of general conclusions from them.

Section **7.5** gives a general discussion of the computational costs involved in performing collocational analysis.

Section **7.6** compares the experimental results attained with those reported elsewhere.

7 . 1 - The BNC vs. the Susanne Corpus

If we study the results of the experiments using the main collocation knowledge base in terms of the input text used and ignore for the time being the version of the collocation dictionary used, taking the average of the results obtained for each input source gives us :

The BNC :

<table>
<tr><th>Percentage Recognised</th><th colspan="2">Percentage Mis-recognised</th></tr>
<tr><td rowspan="3">86.90%</td><td colspan="2">13.10%</td></tr>
<tr><td>Not in Lexicon</td><td>Wrong Word</td></tr>
<tr><td>3.48%</td><td>9.62%</td></tr>
</table>

The Susanne Corpus :

Percentage Recognised	Percentage Mis-recognised	
	34.58%	
65.42%	**Not in Lexicon**	**Wrong Word**
	6.73%	**27.85%**

The first and most obvious conclusion to be drawn from these results is that the recognition accuracy is considerably greater when processing words from the BNC than when processing words from the Susanne Corpus.

This is hardly surprising considering that the collocation knowledge base used in these experiments is based on information derived from the BNC.

The input from the Susanne Corpus is not only outside the direct 'experience' of the knowledge base, but is also in a rather idiosyncratic style (i.e. journalistic language).

How can we judge the system's performance in analysing this input?

If just the lexical level of processing were carried out to give a word lattice containing five valid words at each word position, the chances of the system correctly guessing the input word at each word position is one in five, i.e. 20%. Clearly, making the choice between the potential words at each word position based on collocational knowledge gives a far greater chance of choosing the correct word, based on the results of these experiments.

The results of the experiments using the BNC as input suggest that using a knowledge base created with information derived from the same source as the test data gives us a very high chance of predicting the correct word at any particular word position in a word lattice. In almost nine out of ten cases the correct word will be hypothesised, based on these results.

Another point to note from these results is the relatively strong performance of the general purpose lexicon (based on the Collins Dictionary).

Based on these results, only **5.93%** of words in the test data - approximately one in twenty - were not recognised during lexical processing. Moreover, most of the words that were given as invalid were either proper nouns or numbers (numbers are not included in the system lexicon derived from the Collins Dictionary).

Of course, one of the most interesting points to be extracted from the results of all the experiments is the reason or reasons*why* the mis-recognised words were mis-recognised.

This will be discussed in section **7 . 1 . 4**.

7 . 2 - The Percentage Score Dictionary vs. the Z-score Dictionary

Concentrating on the version of the collocation dictionary used during processing, and ignoring the source of the input text, if we calculate the average of the results obtained for each version of the collocation dictionary, we get :

The Percentage Score Dictionary :

Percentage Recognised	**Percentage Mis-recognised**	
	17.56%	
82.44%	**Not in Lexicon**	**Wrong Word**
	4.67%	**12.89%**

The Z-score Dictionary :

Percentage Recognised	**Percentage Mis-recognised**	
	30.12%	
69.88%	**Not in Lexicon**	**Wrong Word**
	5.54%	**24.58%**

Clearly, if these results are analysed purely in terms of recognition accuracy, the Percentage Score Dictionary outperforms the Z-score Dictionary. This is due to the fact that the Percentage Score Dictionary contains an entry for every collocation contained in the section of the BNC upon which the collocation dictionary is based.

The Percentage Score Dictionary also offers a higher level of detail than the Z-score Dictionary, in that it records the specific relative positions within the defined span of the two words involved in any collocation.

The Z-score Dictionary contains only collocations that are considered to be significant (see **Chapter 4**). This means that rarely occurring collocations will be omitted from the Z-score Dictionary. If these rare collocations occur in the test data they will not contribute any score to the collocational analysis, so the words involved may be mistaken for alternative words which contribute to a more commonly occurring collocation.

The trade-off for this lessening in recognition performance is better performance in terms of computing resources - both filestore and speed of processing.

As the Percentage Score Dictionary boasts complete coverage of the collocational relationships within a text, it is several times larger than the Z-score Dictionary. As well as being more wasteful of filestore, this also has serious implications for the time taken to process an input text.

Experiments using the Z-score Dictionary were carried out in approximately one quarter of the time taken by those using the Percentage Score Dictionary.

It is my feeling that the slight reduction in accuracy suffered when using the Z-score Dictionary is acceptable in view of the great improvement in computational performance.

So the *difference* in accuracy between experiments using the Percentage Score Dictionary and the Z-score Dictionary can be explained by the less comprehensive coverage of the Z-score Dictionary.

The reasons for mistakes occurring even when using the full coverage Percentage Score Dictionary are discussed in section **7 . 1 . 4**.

7 . 3 - The Tailored Knowledge Base

The results obtained from the experiments carried out using the knowledge base specially created from a section of the Susanne Corpus show a high level of recognition accuracy.

Once more, the Percentage Score Dictionary outperforms the Z-score Dictionary in terms of recognition accuracy, and for this very specialised knowledge base, the difference in size between the two versions of the collocation dictionary wasn't as pronounced as for the main knowledge base.

Processing times were therefore very similar for both versions of the collocation dictionary.

The problem of being unable to process invalid words was eradicated in these experiments, as the lexicon was derived from the same source as the collocation dictionary.

However, there is not a great deal of improvement in recognition accuracy in comparison to the experiments carried out on the BNC with a general purpose lexicon.

Interestingly, the reasons for the mis-recognition of words were similar using the tailored knowledge base to those encountered using the main knowledge base.

The main reasons for mis-recognition are discussed in the next section.

7 . 4 - Summary of Results

The results obtained from a variety of experiments strongly suggest that the use of a collocation knowledge base in a post-processing capacity can indeed enhance the performance of a handwriting recognition system based on visual information.

Inevitably however, mistakes are made, and the reasons behind these errors provide an insight into the nature of collocation.

Clearly, some of the errors are made for the simple reason that a relatively rarely occurring combination of words is present in the test data, and a more common sequence of words is mistakenly chosen as the system's hypothesis.

This is to be expected, and is a hazard for any system based solely on statistical measures. Uncommon but valid instances (whether they be letter of word sequences) are always likely to be overlooked in favour of a more common instance in such a system.

Of more interest are the errors which occurred due to the difference in the nature of collocation when it involves 'grammar' words as opposed to 'lexical' words, or a combination of the two. (See**Chapter 2** for an explanation of the terms 'lexical word' and 'grammar word').

At each word position in a lattice being processed, the system must choose the most likely word from a number of alternatives based on collocation information.

We can think of this process as a number of comparisons between pairs of alternative words to ascertain which is the most likely to match the input word at that word position.

There are three kinds of comparison to consider. At each word position the system must make a number of choices between either :

- two lexical words
- two grammar words
- one grammar word and one lexical word

based on the relationships with the words surrounding that word position.

In the first case, the system makes the correct choice the vast majority of the time. As discussed earlier, lexical words tend to have a strong ability to predict their own environment. Problems may arise when a very rare combination of words occurs, in which case this combination will be assigned a very low collocational score (or possibly no score at all if the Z-score Dictionary is being consulted, and the collocation in question is not considered to be significant), and may lose out to a more commonly occurring combination.

No real pattern emerged for the second case - choosing between two grammar words. These words tend to have a very low ability to predict their own environment, so the choice between two grammar words is often decided by a lexical word in a nearby word position having a very strong collocational relationship with one of the grammar words being scrutinised. This is a case of 'upward' and 'downward' collocation as discussed in **Chapter 2**.

Quite a number of the errant hypotheses put forward by the system feature one grammar word being incorrectly suggested in place of another grammar word (e.g. **to** being hypothesised, whereas **the** is the actual word in that position in the test data).

The third case produces a similar effect. While it is very rare for a lexical word to be hypothesised in place of a grammar word, the converse appears to be fairly common, based on these results.

Many of the errors made by the system involve suggesting a grammar word where there is a lexical word in the input text.

This is again due to lexical words within the defined span having very strong collocational links with the grammar word (although the collocation may be weak when the grammar word is taken to be the node).

This is explained by the relative frequencies of grammar words and lexical words. Grammar words tend to have high relative frequencies of occurrence in texts. This means that their collocational behaviour is rather dissipated over such a large number of occurrences.

On the other hand, a particular lexical word may occur only a handful of times in a text and therefore its collocational behaviour is tightly focused on that small number of occurrences.

This is a drawback of the Percentage Score Dictionary's representation of collocation. A very infrequently occurring lexical word can distort the collocational score of a particular path through a word lattice.

Interestingly, such infrequent words seem to be very rarely chosen as matching the input word themselves, but seem to cause the wrong words to be chosen around them.

To sum up then, based on the results of this series of experiments, the collocational relationships between lexical words tend to be quite stable and reliable indicators as to the collocational patterns prevalent in a particular text.

Considering the relationships between grammar words and lexical words can, however, five a rather distorted view of these patterns of co-occurrence, due to the differing relative frequencies of grammar words and lexical words.

Possible ways to deal with this phenomenon are discussed in the next chapter.

7 . 5 – The Computational Cost of the Collocation Analysis System

The first point to make about the collocation analysis system is that it does not currently run in real-time.

The time taken to carry out the analysis of a sentence depends upon a number of factors. There are two stages of processing to consider :

- the creation of a letter lattice for each input word
- the analysis of a word lattice

The process of creating a letter lattice from an input word is described in **Chapter 3**. The time taken to carry out this process depends on the number of letters in the input word as the potential alternatives for each letter must be calculated, and also on the number of lexicon look-ups required.

In practical terms, the time taken to generate a letter lattice from a word is increased due to the fact that a number of intermediate stages are carried out involving the processing of temporary files. The lattice generation process would have been far more rapid if these intermediate stages were carried out using internal data structures. This would have been feasible using a PC with a reasonable specification, but unfortunately was not possible in the environment available for this project.

The upshot of all this is that the generation of a letter lattice from an input word would take on average in the region of three minutes to complete.

The length of the words in an input sentence also affects the time taken to carry out the collocational analysis. As four alternative letters are given for each letter in an input word (giving five choices in total), the number of paths through a word lattice is given as $\mathbf{5^n}$ where **n** is the number of letters in the original word. So we can see that an increase of one letter in an input word means a **400%** increase in the number of paths to be traversed and therefore also in the processing time.

The time taken to process a word lattice also depends on which collocation dictionary is used to provide the collocational knowledge.

The differences between the **Percentage Score Dictionary** and the **Z-score Dictionary** are described in **Chapter 4**.

The Percentage Score Dictionary represents every collocation in the training corpus. The Z-score Dictionary represents only those collocations that are considered to be significant.

As a result of this, collocational analysis based on the Percentage Score Dictionary takes considerably longer than that based on the Z-score Dictionary (in practice approximately three times longer on average).

In real terms, analysing 50 sentences using the Z-score Dictionary would take around one hour. This would become approximately three hours when using the Percentage Score Dictionary.

Analysis based on either dictionary is slowed down by the sequential searches which are necessary due to the dictionaries being stored as text files. It is suggested that storing the collocation information in the form of a relational database would considerably improve performance by passing on the "drudgery" of searching to a database engine optimised for the task. (Applying this principle to the lexicon would also improve the performance of the word lattice generation process).

It is doubtful that even this improvement in structure would result in real-time performance in the case of the Percentage Score Dictionary. It is simply not feasible in a real-time recognition system based on statistics to represent all possible cases however insignificant.

As commercial recognition systems which employ an element of statistical linguistic knowledge (particularly speech recognition systems) become increasingly homogenised in terms of the recognition algorithms that they use, the key differences are found in the ways that they determine which word combinations are significant (see **Chapter 2**).

The Percentage Score Dictionary was constructed and used to see how a collocation knowledge base could perform in its purest form, without being constrained by performance considerations.

It is hoped that given improvements in file structure and search algorithms, collocational analysis based on Z-score statistics could be carried out in something approaching real-time.

Another factor to be considered in any discussion of the computational cost of the system is the amount of filestore available.

A system based on statistical information stands and falls on the data used to compile the statistics. While the 13,000,000+ words used in the compilation of the collocation statistics for this project is a respectable sample, and larger than that used in most other studies in this field, a larger section of the BNC would have been preferable. The limiting factor here was the amount of filestore available to me – **1 Gbyte**. The BNC in total takes up **4 Gbytes**. This is well within the reach of a modern PC, and would inevitably have produced more comprehensive collocation statistics.

7 . 6 – A Comparison of Experimental Results with Other Systems

It would appear that the experimental results achieved using collocational analysis compare favourably with others reported elsewhere when judged purely in terms of recognition accuracy.

Similar work carried out using collocational analysis (see Rose & Evett, 1992, Hull , 1994 for example) have yielded results in the region of 70 – 80% recognition accuracy. The best results achieved by this system (94% accuracy when using the Percentage Score Dictionary to analyse input from the BNC) clearly outstrip these other systems.

The worst case recognition accuracy achieved by the system (60% when using the Z-score dictionary to analyse input from the Susanne Corpus) falls short of the results of previously reported systems. This apparently poor showing is mitigated somewhat when we consider that the system had no prior knowledge of the input text whatsoever (no "training"), and indeed, the input text was in a highly distinctive and idiosyncratic style (the style of American newspapers) and could almost have been designed to flummox a recognition system based on linguistic statistics!

However, the low recognition accuracy does suggest that the Z-score dictionary would benefit from some further work to possibly define a more appropriate threshold of collocation significance.

A comparison of the experimental results with those reported by a number of commercially available systems is also of interest.

Hand-held machines have demonstrated much greater handwriting recognition accuracy in recent years compared to earlier efforts.

Machines such as the Apple Newton™ available in the early 1990s were very constraining in the way that handwriting could be entered – often letter by letter, with each letter having to be entered in a separate box – and suffered from relatively low recognition rates (often around 70-80% *letter* recognition).

Now though, much higher recognition accuracy is achieved (90% and upward word recognition) with varying amounts of training (see for instance Yaeger, 1997).

The main speech recognition packages (IBM's ViaVoice™, Dragon's Naturally Speaking™ and L&H's VoiceXpress™ being the main players) also claim, and in my experience achieve, over 90% recognition accuracy with varying amounts of training. Without adequate training, recognition accuracy drops to around 70-80%.

The results achieved in the experiments described in this thesis suggest that the use of collocation statistics can offer recognition rates close to those demonstrated by fully-trained commercial systems, and in some cases superior to an untrained commercial system, although, as discussed previously, much work is needed to find the most efficient and effective representation of the statistics in order to make a system based on collocation run in anything approaching real-time.

It may be the case that the efficient use of collocation statistics could be used to reduce the amount of training required for a commercial system.

Chapter 8

Conclusions

This thesis outlines the design and construction of a collocation knowledge base, and a process by which a word lattice can be analysed using the knowledge base to discover the path through it which is most likely to correspond to the input.

The notable features of the collocation knowledge base are :

- the size of the sample upon which it is based
- the structure of the collocation dictionary
- the different representations of collocation
- the use of positional information

I shall discuss these features one at a time.

The size of the sample of the language upon which any collocational analysis is based is vital.

If a general view of the collocational behaviour of the words in a language is required, than as large a sample of that language as possible is required.

The sample used in this study consisted of **13,142,316** words. While it is clearly impossible to represent the entirety of a language, I feel that the analysis of a sample of this size will inevitably yield a great deal of generic information about the relationships in a language. As Discussed in **Chapter 7** however, the collocation statistics would ideally have been based on the entire BNC (around 100, 000, 000 words in total), but sufficient filestore was not available.

Collocation is a two-way relationship. The collocational relationship between word **A** and word **B** actually consists of two relationships, one in which **A** is the node and **B** is the collocate, and the other in which **B** is the node and **A** is the collocate.

The system outlined in this thesis captures the two-way nature of collocation by the use of a cascade structure. The cascade is based on alphabetical ordering, so given that word **A** is alphabetically earlier than word **B**, the collocational relationship between the two is represented as :

A

B **Position** **Strength1** **Strength2**

where **Position** is the position of **B** in relation to **A**. Clearly the position of **A** in relation to **B** is implicitly represented as this can be attained simply by reversing the polarity of **Position**.

Strength1 and **Strength2** represent some measure of the strength of the collocational relationship between the two words when **A** is the node and **B** is the collocate, and when **B** is the node and **A** is the collocate respectively.

Therefore the two-way collocational relationship between words **A** and **B** is represented by a single entry, and can therefore be retrieved by a single look-up operation.

Collocations are represented in two ways in this study. The **Percentage Score Dictionary** represents the strength of a collocation of a word **A** with a word **B**, which is in a specific position ***p*** in relation to **A** by the measure :

$$\frac{\textbf{No. of occurrences of A with B in position } p}{\textbf{Total no. of occurrences of A}} * 100$$

and

$$\frac{\textbf{No. of occurrences of B with A in position } -p}{\textbf{Total no. of occurrences of B}} * 100$$

In this representation *every* collocation within a given span contained in the text under analysis is included.

This method of representation is therefore aimed at comprehensive coverage rather than speed of processing.

The **Z-Score Dictionary** meanwhile represents the strength of the collocation of a word **A** with a word **B**, which is in any position within a defined span of words in relation to **A**, by the z-scores calculated when **A** is the node and **B** is the collocate and *vice versa*. The calculation of z-scores is described in **Chapter 2**.

Only collocations with a z-score above a particular level of significance are stored.

This method of representation is therefore aimed more at speed of performance than at complete coverage.

Finally, the representation of positional information in the Percentage Score Dictionary is worthy of note.

The explicit representation of the relative positions of the two words of a collocation means that an extremely accurate picture of the patterns of collocation in a text can be built up.

The use of positional information means that the collocation of word **A** and word **B** in the section of text :

A B C D E

has a separate entry in the collocation dictionary from the collocation of the *same two words* in the section of text :

A C D B E

In the first example, the position stored is **1**, meaning that word **B** is one word position to the right of word **A**. In the second, the position stored is **3**, meaning that word **B** is three word positions to the right of word **A**. The position of word **A** in relation to word **B** is obtained simply by reversing the polarity of these figures, giving **-1** in the first case, and **-3** in the second.

Having discussed the noteworthy points of my system I shall now make suggestions as to further work which may be usefully carried out in this area – see section **8.1**.

Section **8.2** summarises the main points of this thesis.

8 . 1 - Suggestions for Future Work

I shall divide this section into subsections concentrating on aspects of the current system which need improving in some way, or which could benefit from further development.

8 . 1 . 1 - The Lexicon

The lexicon based on the Collins Dictionary offered reasonable coverage in this project. As mentioned previously, many of the words found not to be included in the lexicon were either proper nouns or digits. While it would not be a great problem to modify the lexicon to cope with digits, proper nouns will always pose problems for a system such as this.

There is one further area where the lexicon could be improved.

The Collins Dictionary contains many derivations alongside their root forms, and these were transferred to the lexicon.

Ideally, only the root form of a word would be stored in the lexicon and its derivations would be dealt with by the morphological processing component of the system.

As things stand, if the word **jumping** is represented in the lexicon as well as the word **jump** then both of these words may have entries in the collocation dictionary, whereas from a collocational viewpoint, **jump** and **jumping** are essentially the same word.

It should not be too great a task to devise an automatic method to filter the lexicon in order to remove derivations from it.

8 . 1 . 2 - The Provision of Input

There are a number of inadequacies in the current method of simulating the output of a low-level recogniser.

Ideally of course a genuine, functional recogniser would provide input to the collocation processor with the word lattice being trimmed according to the confidences bestowed on candidate words by the low-level processor.

As such a recogniser is not currently available however, it would be advisable to refine the simulation process in a number of ways.

The current simulator bases its output on single letter substitutions derived from a number of sample word lattices.

Far more believable output would be achieved by basing the extrapolation on letter *pairs*.

This would be a more complex process computationally, but would capture the essence of low-level output more accurately.

Secondly, the processing of letter lattices to produce candidate words is computationally quite intensive. When dealing with a letter lattice with more than around eight columns the processing time becomes unacceptably long.

Some method of dynamically rejecting paths through a letter lattice is required.

Currently, paths containing consecutive letter pairs that never occur in the lexicon are instantly rejected. Perhaps a system exploiting bigram statistics would be preferable, so that paths containing *unlikely* letter pairs (i.e. those whose probability of occurrence falls below a certain threshold) could be instantly discarded.

Finally, I think it is worth investigating the method by which the four candidate words are chosen.

Currently the four most frequent words according to their lexicon entries are chosen. This is not necessarily a suitable method. It does tend to come up with a large proportion of grammar words (due to their high frequency of occurrence), particularly when dealing with shorter words. This quite often produces word lattices which are intuitively inappropriate.

In the absence of the confidence scores habitually supplied by a low-level recogniser, a random method of choosing four candidate words from the list of valid words may be preferable to the method employed at present.

8.1.3 - Taking Word Frequency Into Account

This final subsection looks at the current method of calculating the collocational score for a pair of co-occurring words.

As discussed earlier, problems can arise when dealing with collocations involving high frequency grammar words.

The collocational score for a pair of words is currently calculated by multiplying together the two collocational strength measures stored for the words in the collocation dictionary.

The results of my experiments suggest that this gives a reasonable, though far from infallible, indication of the strength of a collocation between two words.

Mistakes are made largely due to the different relative frequencies of grammar words and lexical words.

I suggest that some sort of weighting be applied to the collocation score calculation, based on the frequencies of the words involved, and the number of times that collocation between the two words occurs, and whether some representation of this frequency information breaches a certain threshold.

It may even be possible to carry out a calculation with a different weighting depending on the classes of the words involved. Are they both grammar words? Both lexical? Or a mixture of the two?

A fairly comprehensive list of grammar words is available for this purpose.

I suspect that the specific nature of the weightings appliedwould have to be discovered by trial and error, and there will still inevitably be exceptions to the rule, as is the case with any system based solely on statistical criteria.

8 . 1 . 4 - Future Developments

It would be interesting to integrate the collocational analysis system into a larger system, exploiting various sources of linguistic knowledge.

Combination with a syntactic processing system would be especially interesting. An important issue would be how to link the two components.

If they were linked in 'series', the syntactic processor could act as a filter, discarding grammatically incorrect paths from the word lattice under consideration.

Linked in 'parallel' the two components would work independently on the same input, then each pass their output hypotheses to a decision-making module which would apply various criteria in choosing the most appropriate hypotheses.

8 . 2 - Summary

In this thesis I have reviewed the literature relating to automatic text recognition, including the motivation behind the attempts to implement the recognition of text by computer, and the efforts to do this based on strictly visual information, but concentrating on the efforts to utilise linguistic knowledge in automatic recognition systems to reflect human performance.

This review concludes that the use of various levels of linguistic knowledge to aid in the task of recognition is a desirable goal.

I have described the theory, design and construction of a collocation knowledge base, consisting of a lexicon and a collocation dictionary. I have also described the implementation of a system which simulates the output of a low-level recognition system.

Collocations are represented in two different ways :

- in the Percentage Score Dictionary, in which every collocation in a text is represented, and explicit positional information is stored.

- in the Z-score Dictionary, in which only collocations above a particular significance threshold and within a defined span are stored.

A variety of experiments are described which test various aspects of the system. These experiments highlight the advantages and disadvantages of the different representations of collocation, and the value of tailoring a knowledge base to suit the input on which it operates. The results of these experiments suggest that the exploitation of a collocation knowledge base can indeed aid in the task of automatic text recognition.

Also, a number of general conclusions about collocation are drawn, particularly relating to the differing collocational behaviour of grammar words and lexical words.

Finally, suggestions for potentially fruitful future development in this field are put forward.

Bibliography

Aarts, J. and **Meijs, W.** (eds.) 1986. *Corpus Linguistics II : New Studies in the Analysis and Exploration of Computer Corpora.* Amsterdam : Rodopi.

Aarts, J. and **Meijs, W.** (eds.) 1990. *Theory and Practice in Corpus Linguistics.* Amsterdam : Rodopi.

Aijmer, K. 1986. 'Discourse Variation and Hedging', in Aarts, J. and Meijs, W. (eds.), pp.1-18.

Aijmer, K. and **Altenberg, B.** (eds.) 1991. *English Corpus Linguistics : Studies in Honour of Jan Svartik.* London : Longmans.

Aitken, A.J., Bailey, R.W. and **Hamilton-Smith, N.** (eds.) 1973. *The Computer and Literary Studies*. Edinburgh : Edinburgh University Press.

Alshawi, H. 1988. 'Analysing the dictionary definitions', in Boguraev, B. and Briscoe, E. (eds.), pp.153-169.

Altenberg, B. and **Eeg-Olofsson, M.** 1990. 'Phraseology in Spoken English : Presentation of a Project', in Aarts, J. and Meijs, W. (eds.), pp.1-26.

Alvertos, N. and **D'Cunha, I.** 1991. 'Optical Machine Recognition of Handwritten and Printed Lowercase Greek Characters of any size', *Optical Engineering*, 30, no 12, pp.1920-1930.

Amsler, R.A. 1981. 'A Taxonomy for English Nouns and Verbs', *Proceedings of the 19th Annual Meeting of the ACL*, Stanford, pp.133-8.

Arnold, D. 1990. 'Text Typology and Machine Translation : an overview', *Translating and the Computer 10. Translation Environment 10 years on*, pp.73-9.

Atwell, E.S. 1987. 'How to Detect Grammatical Errors in a text without Parsing it', *Proceedings of the Third Conference of the European Chapter of the ACL*, New Jersey, pp.38-45.

Badie, K. and **Shimura, M.** 1982. 'Machine Recognition of Roman Cursive Scripts', *6th International Conference on Pattern Recognition. IEEE*, pp.28-30.

Balestri, M. and **Masera, L.** 1988. 'A system for isolating characters in cursive script', *Signal Processing IV : Theories and Applications. Proceedings of EUSIPCO-88, Fourth European Signal Processing Conference*, 2, pp.845-6.

Bazell, C.E. *et al.* (eds.) 1966. *In Memory of J.R.Firth*. London : Longman.

Beale, R. and **Finlay, J.** (eds.) 1992. *Neural Networks and Pattern Recognition in Human-Computer Interaction*. New York : Ellis Horwood.

Becker, C.A. 1979. 'Semantic Context and Word Frequency Effects in Visual Word Recognition', *Journal of Experimental Psychology : Human Perception and Performance*, 5 no 2, pp.252-9.

Bellaby, G.J. and **Evett, L.J.** 1994. 'The Integration of Knowledge Sources for Word Recognition', in Evett, L.J. and Rose, T.G. (eds.), position paper no 3.

Berry, M. 1977. *Introduction to Systemic Linguistics, Vol.2 : Levels and Links*. London : Batsford.

Berry-Rogghe, G.L.M. 1973. 'The Computation of Collocations and their relevance in Lexical Studies', in Aitken, A.J., Bailey, R.W. and Hamilton-Smith, N. (eds.), pp.103-112.

Biber, D. 1986. 'Spoken and Written Textual Dimensions in English : Resolving the Contradictory Findings', *Language*, 62, pp.384-414.

Biber, D. and **Finegan, E.** 1986. 'An Initial Typology of English Text Types', in Aarts, J. and Meijs, W. (eds.)

Black, D. 1958. *The Theory of Committees and Elections*. London : Cambridge University Press.

Blankenship, J. 1962. 'A Linguistic Analysis of Oral and Written Style', *Quarterly Journal of Speech*, 48, pp.419-22.

Bledsoe, W.W. and **Browning, I.** 1959. 'Pattern Recognition and Reading by Machine', *Proceedings of the Eastern Joint Computer Conference*, pp.225-232.

Boccignone, G. *et al.* 1993. 'Recovering Dynamic Information from Static Hnadwriting', *Pattern Recognition*, 26, no 3, pp.409-418.

Boguraev, B. and **Briscoe, E.** (eds.) 1988. *Computational Lexicography for Natural Language Processing*. London : Longmans.

Bourbakis, N.G. and **Gumahad II, A.T.** 1991. 'Knowledge-Based Recognition of Typed Text Characters', *International Journal of Pattern Recognition and Artificial Intelligence*, 5, nos 1&2, pp.293-309.

Bozinovic, R.M. and **Srihari, S.N.** 1984. 'Knowledge-Based Cursive Script Interpretation', *VIIth International Conference on Pattern Recognition*, Montreal, 30 July-2 August, pp.774-776.

Bozinovic, R.M. and **Srihari, S.N.** 1989. 'Off-line Cursive Script Word Recognition' *IEEE Transactions on Pattern Analysis and Machine Intelligence*, II, no 1, pp.68-83.

Breuel, T.M. 1994. 'Language Modelling for a Real-World Handwriting Recognition Task', in Evett, L.G. and Rose, T.G. (eds.), paper no 10.

Briggs, R.O. *et al.* 1992. 'Is the Pen mightier than the Keyboard?', *Proceedings of the 25th Hawaii International Conference on System Sciences*, 3, pp.201-10.

Briggs, R.O. *et al.* 1992-93. 'Whither the Pen-based Interface?', *Journal of Management Information Systems*, 9, no 3, pp.71-90.

Brown, R.M. 1964. 'On-line Computer Recognition of Handprinted Characters', *IEEE Transactions on Electronic Computers*, EC-13, pp.750-2.

Burnard, L. (ed.) 1995. *Users Reference Guide for the British National Corpus*. Oxford : Oxford University Computing Services.

Butler, C.S. 1992. *Computers and Written Texts*. Oxford : Blackwell.

Caesar, T. *et al.* 1994. 'Handwriting Recognition by Statistical Methods', *NATO ASI series. Series F, Computer and System Sciences*, 124, pp.218-222.

Carpenter, P.A. and **Just, M.A.** 1983. 'What your Eyes do while your Mind is Reading', in Rayner, K. (ed.).

Carr, R.M. 1991. 'Handwriting Recognition in the GO Operating System', *COMPCON Spring '91. Digest of Papers*, pp.483-6.

Carroll, J. and **Briscoe, T.** 1994. 'Integrating Probabilistic and Knowledge-based Approaches to Corpus Parsing', in Evett, L.J. and Rose, T.G. (eds.), paper no 1.

Casey, R.G. and **Nagy, G.** 1968. 'An Autonomous Reading Machine', *IEEE Transactions on Computers*, C-17, no 5, pp.492-503.

Cattell, J.M. 1885. 'The Inertia of the Eye and Brain', *Brain*, 8, pp.295-312.

Chafe, W.L. 1982. 'Integration and Involvement in Speaking, Writing and Oral Literature', in Tannen, D. (ed.), pp.35-53.

Cheriet, M. 1994. 'Towards a Visual Recognition of Cursive Script', *NATO ASI series. Series F, Computer and System Sciences*, 124, pp.223-7.

Chomsky, N. 1957. *Syntactic Structures*. The Hague : Mouton & Co.

Chomsky, N. 1966. *Topics in the Theory of Generative Grammar*. The Hague : Mouton & Co.

Church, K. and **Hanks, P.** 1989. 'Word Association Norms, Mutual Information and Lexicography', *Proceedings of the 27th Meeting of the ACL*, pp.76-83.

Churcher, G. *et al.* 1994. 'Bigram and Trigram models for language identification and classification', in Evett, L.J. and Rose, T.G. (eds.), position paper no 4.

Cofer, C.N. (ed.) 1976. *The Structure of Human Memory*. San Francisco : W.H.Freeman.

Cooper, W.E. and **Walker, E.C.T.** (eds.) 1979. *Sentence Processing : Psycholinguistic Studies Presented to Merrill Garrett*, Hillsdale, NJ : Lawrence Erlbaum Associates.

Diaper, D. 1988. 'Natural Language Communication with Computers : Theory , Needs and Practice', *KBS in Government '88. Proceedings of the 2nd European Conference, Blenheim Online Ltd. for the Central Computer and Telecommunications Agency*, pp.19-44.

Dimauro, G., Impedovo, S. and **Pirlo, G.** 1991. 'Uncertainty in the Recognition Process : some considerations of human variable behaviour', *Proceedings, 2nd International Workshop on Frontiers in Handwriting Recognition*, Bonas, France, pp.133-147.

Dimauro, G., Impedovo, S. and **Pirlo, G.** 1992. 'From Character to Cursive Script Recognition : Future Trends in Scientific Research', *Proceedings, 11th IAPR International Conference on Pattern Recognition*, The Hague, Netherlands, pp.516-9.

Dimov, D.T. 1994. 'An Approximate String Matching Method for Handwriting Recognition Post-Processing Using a Dictionary', *NATO ASI series. Series F, Computer and System Sciences*, 124, pp.323-332.

Doermann, D.S. and **Rosenfeld, A.** 1992. 'Recovery of Temporal Information from Static Images of Handwriting', *Proceedings. 1992 IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pp.162-8.

Downton, A.C. *et al.* 1992. 'Recognition of handwritten British postal addresses', in Impedovo, S. and Simon, J.C. (eds.), pp.129-144.

Dunn, C.E. and **Wang, P.S.P.** 1992. 'Character Segmentation Techniques for Handwritten Text - A Survey', *Proceedings, 11th IAPR International Conference on Pattern Recognition*, The Hague, Netherlands, pp.577-580.

Earnest, L.D. 1962. 'Machine Recognition of Cursive Script', in Information Processing, London : Butterworths, pp.462-6.

Edelman, S., Flash, T. and **Ullman, S.** 1990. 'Reading Cursive Handwriting by Alignment of Letter Prototypes', *International Journal of Computer Vision*, 5, no 3, pp.303-331.

Eden, M. 1962. 'Handwriting and Pattern Recognition', *IRE Transactions on Information Theory*, pp.160-166.

Eden, M. and **Halle, M.** 1961. 'The Characterisation of Cursive Writing', *4th London Symposium on Information Theory*, pp.287-299.

Ehrich, R.W. and **Koehler, K.J.** 1975. 'Experiments in the Contextual Recognition of Cursive Script', *IEEE Transactions on Computers*, C-24, pp.182-194.

Ehrlich, S.F. and **Rayner, K.** 1981. 'Contextual Effects on Word Perception and Eye Movements during Reading', *Journal of Verbal Learning and Verbal Behavior*, 20, pp.641-655.

Ellegard, A. 1978. 'The Syntactic Structures of English Texts', *Gothenburg Studies in English*, 43.

Estes, W.K. 1977. 'On the Interaction of Perception and Memory in Reading', in Laberge, D. and Samuels, S.J. (eds.), pp.1-25.

Evett, L.J. *et al.* 1992. 'Using Linguistic Information to aid Handwriting Recognition', in Impedovo, S. and Simon, J.C. (eds.), pp.339-348.

Evett, L.J. and **Rose, T.G.** (eds.) 1994. *Computational Linguistics for Speech and Handwriting Recognition.* AISB '94, One day Workshop, Leeds University.

Farag, R.F.H. 1979. 'Word-level recognition of Cursive Script', *IEEE Transactions on Computers*, C-28, no 2, pp.172-5.

Firth, J.R. 1935. 'The Technique of Semantics', in Firth, 1957a, pp.7-33.

Firth, J.R. 1951. 'Modes of Meaning', in Firth, J.R., 1957a, pp.190-215.

Firth, J.R. 1957a. *Papers in Linguistics, 1934-51.* London : Oxford University Press.

Firth, J.R. 1957b. 'A synopsis of linguistic theory, 1930-55', in Palmer, F.R. (ed.), pp.168-205.

Fischer, G.L. *et al.* 1962. *Optical Character Recognition.* Washington DC : Spartan Books.

Fisiak, J. (ed.) 1976. *Papers and Studies in Contrastive Linguistics, Volume 5.* Virginia : Poznan.

Forney Jr, G.D. 1973. 'The Viterbi Algorithm', *Proceedings of the IEEE*, 61, pp.268-278.

Forster, K.I. 1979. 'Levels of Processing and the Structure of the Language', in Cooper, W.E. and Walker, E.C.T. (eds.), pp.27-85.

Freedman, D.H. 1993. 'Recognizing Handwriting in Context',*Science*, 260, p.1723.

Frishkopf, L.S. and **Harmon, L.D.** 1961. 'Machine Reading of Cursive Script', *4th London Symposium on Information Theory*, pp.300-316.

Fujisaki, T. *et al.* 1991. 'On-line Run-on Character Recognizer : Design and Performance', *International Journal of Pattern Recognition and Artificial Intelligence*, 5, nos 1 & 2, pp.123-137.

Garside, R., Leech, G. and **Sampson, G.** (eds.) 1987. *The Computational Analysis of English : a corpus-based approach*. London : Longman.

Gilloux, M. 1994. 'Hidden Markov Models in Handwriting Recognition', *NATO ASI series. Series F, Computer and System Sciences*, 124, pp.264-288.

Gorsky, N.D. 1994. 'Off-line Recognition of Bad Quality Handwritten Words using Prototypes',*NATO ASI series. Series F, Computer and System Sciences*, 124, pp.199-217.

Govindaraju, V., Wang, D. and **Srihari, S.N.** 1992. 'Using Temporal Information in Off-line Word Recognition',*USPS 5th Advanced Technology Conference*.

Grishman, R. 1986. *Computational Linguistics : an Introduction*. Cambridge : Canbridge University Press.

Grishman, R. and **Kittredge, R.** (eds.) 1986. *Analyzing Language in Restricted Domains : Sublanguage Description and Processing*. Hillsdale, NJ : Lawrence Erlbaum Associates.

Grosz, B.J., Spark-Jones, K. and **Webber, B.L.** (eds.) 1986. *Readings in Natural Language Processing*. Los Altos, California : M.Kaufmann Publishers.

Guthrie, J. 1993. 'A Note on Lexical Disambiguation', in Souter, C. and Atwell, E.S. (eds.), pp.227-238.

Haber, R.N. and **Haber, L.R.** 1981. 'The shape of a word can specify its meaning', *Reading Research Quarterly*, XVI, no 3, pp.334-345.

Halliday, M.A.K. 1961. 'Categories of the Theory of Grammar', *Word*, 17, no 3 pp.241-292.

Halliday, M.A.K. 1966. 'Lexis as a Linguistic Level', in Bazell, C.E. *et al.* (eds.), pp.148-162.

Halliday, M.A.K. 1991. 'Corpus Studies and Probabilistic Grammar', in Aijmer, K. and Altenberg, B. (eds.), pp.30-43.

Halliday, M.A.K. and **Fawcett, R.P.** (eds.) 1987. *New Developments in Systemic Linguistics. Volume 1 : Theory and Description*. London : Pinter.

Halliday, M.A.K. and **Hasan, R.** 1976. *Cohesion in English*. London : Longman.

Hanlon, S.J. and **Boyle, R.D.** 1992. 'Syntactic Knowledge in Word Level Text Recognition', in Beale, R. and Finlay, J. (eds.), pp.173-189.

Hanson, A.R., Riseman, E.M. and **Fisher, E.** 1976. 'Context in Word Recognition', *Pattern Recognition*, 8, pp.35-45.

Harmon, L.D. 1962. 'Automatic Reading of Cursive Script', in Fischer, G.L. *et al.* (eds.), pp.151-2.

Harmon, L.D. 1972. 'Automatic Recognition of Print and Script', *Proceedings of the IEEE*, 60, no 10, pp.1165-1176.

Hasan, R. 1987. 'The Grammarian's Dream : Lexis as most delicate grammar', in Halliday, M.A.K. and Fawcett, R.P. (eds.), pp.184-211.

Healy, A.F. 1976. 'Detection Errors on the Word "The" : Evidence for Reading Units larger than letters', *Journal of Experimental Psychology : Human Perception and Performance*, 2, pp.235-242.

Hebrail, G. and **Suchard, M.** 1990. 'Classifying Documents : A Discriminant Analysis and an Expert System Work Together',*COMPSTAT. Proceedings in Computational Statistics. 9th Symposium*, pp.63-8.

Henderson, L. 1982. *Orthography and Word Recognition in Reading.* London : Academic Press.

Ho, T.K., Hull, J.J. and **Srihari, S.N.** 1992. 'On Multiple Classifier Systems for Pattern Recognition', *Proceedings, 11th IAPR International Conference on Pattern Recognition*, The Hague, Netherlands, pp.84-7.

Hoffman, J., Skrzypek, J. and **Vidal, J.J.** 1993. 'Cluster Network for Recognition of Handwritten, Cursive Script Characters',*Neural Networks*, 6, pp.69-78.

Holt, M.J.J., Beglou, M.M. and **Datta, S.** 1992. 'Slant-independent letter segmentation for off-line cursive script recognition', in Impedovo, S. and Simon, J.C. (eds.), pp.41-6.

Hong, T. and **Hull, J.J.** 1993. 'Text Recognition Enhancement with a Probabilistic Lattice Chart Parser', *Proceedings of the Second International Conference on Document Analysis and Recognition*, Tsukuba Science City, Japan, pp.222-5.

Houle, G.F. 1994. 'A Hierarchical Handwritten Word Segmentation',*NATO ASI series. Series F, Computer and System Sciences*, 124, pp.228-232.

Hughes, J. and **Atwell, E.S.** 1994. 'A Methodical Approach to Word Class Formation using Automatic Evaluation', in Evett, L.J. and Rose,T.G. (eds.), paper no 5.

Hull, J.J. 1987. 'A Computational Theory and Algorithm for Fluent Reading', *Proceedings of the 3rd IEEE Conference on Artificial Intelligence*, pp.176-181.

Hull, J.J. 1992. 'A Hidden Markov Model for Language Syntax in Text Recognition', *Proceedings, 11th IAPR International Conference on Pattern Recognition*, The Hague, Netherlands, pp.124-7.

Hull, J.J. 1994. 'Language-Level Syntactic and Semantic Constraints Applied to Visual Word Recognition',*NATO ASI series. Series F, Computer and System Sciences*, 124, pp.289-312.

Hull, J.J. *et al.* 1992. 'Combination of Segmentation-based and Wholistic Handwritten Word Recognition Algorithms', in Impedovo, S. and Simon, J.C. (eds.), pp.261-272.

Hull, J.J., Khoubyari, S. and **Ho, T.K.** 1992. 'Word Image Matching as a Technique for Degraded Text Recognition', *Proceedings, 11th IAPR International Conference on Pattern Recognition*, The Hague, Netherlands, pp.665-8.

Hull, J.J. and **Srihari, S.N.** 1982. 'Experiments in Text Recognition with Binary n-grams and Viterbi Algorithms', *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 4, pp.520-530.

Hunnicutt, S. 1989. 'Using Syntactic and Semantic Information in a Word Prediction Aid', *EUROSPEECH '89. European Conference on Speech Communication and Technology*, 1, pp.191-3.

Impedovo, S. and **Simon, J.C.** 1992. *From Pixels to Features III : frontiers in handwriting recognition*. London : North-Holland.

Johansson, S. (ed.) 1982. *Computer Corpora in English Language Research*. Bergen : Norwegian Computing Centre for the Humanities.

Jones, M.A., Story, G.A. and **Ballard, B.W.** 1991. 'Integrating Multiple Knowledge Sources in a Bayesian OCR Post-Processor', *First International Conference on Document Analysis and Recognition*, Saint-Malo, France, pp.925-933.

Jones, S. and **Sinclair, J.McH.** 1974. 'English Lexical Collocations', *Cahiers de Lexicologie*, 24, pp.15-61.

Jost, U. and **Atwell, E.S.** 1994. 'A Hierarchical, mutual-information based Probabilistic Language Model', in Evett, L.J. and Rose,T.G. (eds.), paper no 6.

Just, M.A. and **Carpenter, P.A.** 1987. *The Psychology of Reading and Language Comprehension*. Newton, Mass. : Allyn and Bacon.

Kahan, S., Pavlidis, T. and **Baird, H.S.** 1987. 'On the Recognition of Printed Characters of any Font and Size', *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 9, no 2, pp.274-288.

Kay, M. 1986. 'Algorithm Schemata and Data Structures in Syntactic Processing', in Grosz, B.J., Spark-Jones, K. and Webber, B.L. (eds.), pp.35-70.

Keenan, F.G. 1992. *Large Vocabulary Syntactic Analysis for Text Recognition*. PhD Thesis, Nottingham Trent University.

Keenan, F.G. and **Evett, L.J.** 1989. 'Lexical Structure for Natural Language Processing', *Proceedings, First International Language Acquisition Workshop, IJCAI-89*.

Keenan, F.G. and **Evett, L.J.** 1994. 'Applying Syntactic Information to Text Recognition', in Evett, L.J. and Rose, T.G. (eds.), paper no 2.

Keenan, E.L. 1975. *Formal Semantics of Natural Language*. Cambridge : Cambridge University Press.

Kelly, D.A. 1992. 'Neural Networks for Handwriting Recognition', *Proceedings of the SPIE - The International Society for Optical Engineering*, 1709, pt 1, pp.143-154.

Kjellmer, G. 1982. 'Some Problems Relating to the study of Collocations in the Brown Corpus', in Johansson, S. (ed.), pp.25-33.

Kjellmer, G. 1990. 'Patterns of Collocability', in Aarts, J. and Meijs, M. (eds.), pp.163-178.

Kjellmer, G. 1991. 'A Mint of Phrases', in Aijmer, K. and Altenberg, B. (eds.), pp.111-127.

Koh, I.G. *et al.* 1994. 'Improvement of OCR by Language Model', *NATO ASI series. Series F, Computer and System Sciences*, 124, pp.318-322.

Kolers, P.A., Wrolstad, M. and **Bouma, H.** (eds.) 1979. *Processing of Visible Language, Volume 1*. New York : Plenum.

Koriat, A., Greenberg, S.N. and **Goldshmid, Y.** 1991. 'The Missing Letter Effect in Hebrew : Word Frequency or Word Function?', *Journal of Experimental Psychology : Learning, Memory and Cognition*, 17, no 1, pp.66-80.

Kuhn, R. 1988. 'Speech Recognition and the Frequency of recently used words : A Modified Markov Model for Natural Language', *Proceedings of the 12th International Conference on Computational Linguistics*, Budapest, Hungary, pp.348-350.

Laberge, D. and **Samuels, S.J.** (eds.) 1977. *Basic Processes in Reading : Perception and Comprehension*. Hillside, NJ : Lawrence Erlbaum Associates.

Lancashire, I. 1987. 'Using a Textbase for English-language Research', *Proceedings of the 3rd Annual Conference of the UWC for the New Oxford English Dictionary*, Waterloo, pp.51-64.

Lecolinet, E. and **Baret, O.** 1994. 'Cursive Word Recognition : Methods and Strategies', *NATO ASI series. Series F, Computer and System Sciences*, 124, pp.235-263.

Lettera, C. *et al.* 1986. 'Use of a Dictionary in conjunction with a Handwritten Text Recognizer', *Proceedings of the VIII International Conference on Pattern Recognition*, Paris, pp.699-701.

Lindgren, N. 1965. 'Machine Recognition of Human Language. Part III - Cursive Script Recognition', *IEEE Spectrum*, 2, pt 1, pp.104-116.

Mackin, R. 1978. 'On Collocations : "Words shall be known by the company they keep"', in Strevens, P. (ed.), pp.149-165.

Mahach, K.R. 1989. 'A Comparison of Computer Input Devices : Linus Pen, Mouse, Cursor keys and Keyboard', *Proceedings of the Human Factors Society 33rd Annual Meeting. Perspectives*, 1, pp.330-4.

Mandler, E., Oed, R. and **Doster, W.** 1985. 'Experiments in on-line script recognition', *Proceedings, 4th Scandinavian Conference on Image Analysis*, pp.75-86.

Mantas, J. 1986. 'An Overview of Character Recognition Methodologies', *Pattern Recognition*, 19, no 6, pp.425-430.

Markowitz, J., Ahlswede, T. and **Evans, M.** 1986. 'Semantically Significant Patterns in Dictionary Definitions', *Proceedings of the 24th Annual Meeting of the Association for Computational Linguistics*, Columbia, pp.112-9.

Marslen-Wilson, W. 1975. 'The Limited Compatibility of Linguistic and Perceptual Explanations', *Papers from the Parasession on Functionalism*, University of Chicago, pp.409-420.

Matthews, G.H. 1961. 'Analysis by Synthesis of Sentences of Natural Languages', *Proceedings, 1961 Conference on Machine Translation and Applied Language Analysis*, II, pp.531-540.

Mermelstein, P. and **Eden, M.** 1964. 'Experiments on Computer Recognition of Connected Handwritten Words',*Information and Control*, 7, pp.255-270.

Meyer, D.E. and **Schvaneveldt, R.W.** 1971. 'Facilitation in Recognizing Pairs of Words : Evidence of a Dependence between Retrieval Operations', *Journal of Experimental Psychology*, 90, pp.227-234.

Meyer, D.E. and **Schvaneveldt, R.W.** 1976. 'Meaning, Memory Structure and Mental Processes', in Cofer, C.N. (ed.), pp.54-89.

Miller, G.A. and **Isard, S.** 1963. 'Some Perceptual Consequences of Linguistic Rules', *Journal of Verbal Learning and Verbal Behavior*, 41, pp.329-335.

Modd, D.T. and **Atwell, E.S.** 1994. 'A Word Hypothesis Lattice Corpus - a benchmark for linguistic constraint models', in Evett, L.J. and Rose, T.G. (eds.), position paper no 9.

Morasso, P. *et al.* 1990. 'Self-organisation of an Allograph Lexicon', *INNC 90, Paris. International Neural Network Conference*, 1, pp.141-4.

Morasso, p. and **Pagliano, S.** 1991. 'A Neural Architecture for the Recognition of Cursive Handwriting', *Fourth Italian Workshop. Parallel Architectures and Neural Networks*, pp.250-4.

Mori, S. 1994. 'Historical Review of Theory and Practice of Handwritten Character Recognition', *NATO ASI series. Series F, Computer and System Sciences*, 124, pp.43-69.

Morton, J. 1969. 'Interaction of Information in Word Processing', *Psychological Review*, 76, no 2, pp.165-178.

Morton, J. 1979. 'Facilitation in Word Recognition : Experiments causing change in the Logogen Model', in Kolers, P.A., Wrolstad, M. and Bouma, H. (eds.), pp.259-268.

Nadal, C. and **Suen, C.Y.** 1993. 'Applying Human Knowledge to improve Machine Recognition of Confusing Handwritten Numerals', *Pattern Recognition*, 26, no 3, pp.381-9.

Nagy, G. 1992. 'Teaching a Computer to Read', *Proceedings, 11th IAPR International Conference on Pattern Recognition*, The Hague, Netherlands, pp.225-9.

Neisser, U. and **Weene, P.** 1960. 'A Note on Human Recognition of Hand-Printed Characters', *Information and Control*, 3, no 2, pp.191-6.

Nilson, N.J. 1971. *Problem-Solving Methods in Artificial Intelligence*. New York : McGraw-Hill Inc.

O'Donnell, R.C. 1974. 'Syntactic Differences between Speech and Writing', *American Speech*, 49, pp.102-110.

Ouladj, H. *et al.* 1989. 'From Primitives to Letters. A Structural Method to Automatic Cursive Handwriting Recognition', *Proceedings of the 6th Scandinavian Conference on Image Analysis*, 1, pp.593-8.

Paquet, T. and **Lecourtier, Y.** 1993. 'Recognition of Handwritten Sentences using a Restricted Lexicon', *Pattern Recognition*, 26, no 3, pp.391-407.

Palmer, F.R. (ed.) 1968. *Selected Papers of J.R.Firth, 1952-59*. London : Longman.

Patten, T. 1992. 'Computers and Natural Language Parsing', in Butler,C.S. (ed.), pp.29-52.

Pittman, J.A. 1991. 'Recognizing Handwritten Text', *Human Factors in Computing Systems. Reaching Through Technology. CHI '91. Conference Proceedings*, pp.271-5.

Plate, T. 1988. 'Obtaining and using Co-occurrence Statistics from LDOCE', in Boguraev, B. and Briscoe, E. (eds.), pp.202-210.

Poole, M.E. and **Field, T.W.** 1976. 'A Comparison of Oral and Written Code Elaboration', *Language and Speech*, 19, pp.305-311.

Rayner, K. (ed.) 1983. *Eye Movements in Reading : Perceptual and Language Processes*. New York : Academic Press.

Rayner, K., Carlson, M. and **Frazier, L.** 1983. 'The Interaction of Syntax and Semantics during Sentence Processing : Eye Movements in the Analysis of Semantically Biased Sentences', *Journal of Verbal Learning and Verbal Behaviour*, 22, pp.358-374.

Register, M.S. and **Kannan, N.** 1992. 'A Hybrid Architecture for Text Classification', *Proceedings. Fourth International Conference on Tools with Artificial Intelligence, TAI '92*, pp.286-292.

Reilly, R. and **Sharkey, N.E.** (eds.) 1990. *Connectionist Approaches to Natural Language Processing*. Hove : Lawrence Erlbaum Associates.

Renouf, A. and **Sinclair, J.McH.** 1991. 'Collocational Frameworks in English', in Aijmer, K. and Altenberg, B. (eds.), pp.128-143.

Riseman, E.M. and **Ehrich, R.W.** 1971. 'Contextual Word Recognition using Binary Digrams', *IEEE Transactions on Computers*, C-20, no 4, pp.397-403.

Roos, E. 1976. 'Contrastive Collocational Analysis', in Fisiak, J. (ed.), pp.65-77.

Rose, T.G. and **Evett, L.J.** 1992. 'A Large Vocabulary Semantic Analyzer for Handwriting Recognition', *AISB Quarterly*, 80, pp.34-9.

Rose, T.G. and **Evett, L.J.** 1993. 'Semantic Analysis for Large Vocabulary Cursive Script Recognition', *Proceedings of the Second International Conference on Document Analysis and Recognition*, Tsukuba Science City, Japan, pp.236-9.

Rose, T.G. and **Evett, L.J.** 1995. 'The use of Context in Cursive Script Recognition', *Machine Vision and Applications*, 8, no 4, pp.241-8.

Rose, T.G., Evett, L.J. and **Lee, M.J.** 1994. 'Contextual Analysis for Text Recognition : A Comparison with Human Performance', in Evett, L.J. and Rose, T.G. (eds.), paper no 4.

St.John, M.F. and **McClelland, J.L.** 1990. 'Parallel Constraint Satisfaction as a Comprehension Mechanism', in Reilly, R. and Sharkey, N.E. (eds.), pp.97-136.

Sakoe, H. and **Chiba, S.** 1971. 'A Dynamic Programming Approach to Continuous Speech Recognition', *Proceedings of the 7th International Congress on Acoustics*, Paper 20, C13, pp.65-8.

Salvendy, G. and **Smith, M.J.** (eds.) 1989. *Designing and Using Human-Computer Interfaces and Knowledge-based Systems*.
Amsterdam : Elsevier.

Sampson, G. 1987. 'Probabilistic Models of Analysis', in Garside, R., Leech, G. and Sampson, G. (eds.), pp.16-29.

Sampson, G. 1995. *English for the Computer*. London : Oxford University Press.

Sayre, K.M. 1973. 'Machine Recognition of Handwritten Words : A Project Report', *Pattern Recognition*, 5, no 3, pp.213-228.

Schuberth, R.E. and **Eimas, P.D.** 1977. 'Effects of Context on the Classification of Words and Non-words', *Journal of Experimental Psychology : Human Perception and Performance*, 3, no 1. pp.27-36.

Senior, A.W. 1994. 'Normalisation and Preprocessing for a Recurrent Network Off-line Handwriting Recognition System', *NATO ASI series. Series F, Computer and System Sciences*, 124, pp.360-5.

Shannon, C.E. 1951. 'Prediction and Entropy of Printed English', *Bell Systems Technical Journal*, 30, pp.50-64.

Shingal, R. and **Toussaint, G.T.** 1979. 'A Bottom-up and Top-down Approach to using Context in Text Recognition', *International Journal of Man-Machine Studies*, II, pp.201-212.

Simon, J.C. 1992. 'Off-line Cursive Word Recognition', *Proceedings of the IEEE*, 80, pp.1150-1161.

Simon, J.C. 1994. 'On the Robustness of Recognition of Degraded Line Images', *NATO ASI series. Series F, Computer and System Sciences*, 124, pp.175-8.

Sinclair, J.McH. 1966. 'Beginning the Study of Lexis', in Bazell, C.E. *et al.*, pp.410-430.

Sinclair, J.McH. 1982. 'Reflections on Computer Corpora in English Language Research', in Johansson, S. (ed.), pp.1-6.

Sinclair, J.McH. 1987. 'Collocation : a Progress Report', in Steele, R. and Threadgold, T. (eds.), pp.319-331.

Sinclair, J.McH. 1991. *Corpus, Concordance, Collocation*. London : Oxford University Press.

Sinclair, J.McH., Jones, S. and **Daley, R.** 1970. *English Lexical Studies*. Report No.5060, Office of Scientific and Technical Information, London.

Souter, C. and **Atwell, E.S.** (eds.) 1993. *Corpus-Based Computational Linguistics*. Amsterdam : Rodopi Press.

Srihari, S.N 1985. *Computer Text Recognition and Error Correction*. Silver Spring, Md. : IEEE Computer Society Press.

Srihari, S.N. 1992. 'High-Performance Reading Machines', *Proceedings of the IEEE*, 80, pp.1120-1132.

Srihari, S.N. 1996. 'Recent Advances in Off-line Handwriting Recognition', *Fifth International Workshop on Frontiers in Handwriting Recognition*, Essex, England.

Srihari, S.N., Hull, J.J. and **Choudhari, R.** 1983. 'Integrating Diverse Knowledge Sources in Text Recognition', *ACM Transactions on Office Information Systems*, 1, pp.68-87.

Steele, R. and **Threadgold, T.** (eds.) 1987. *Language Topics. Essays in Honour of Michael Halliday*. Philadelphia : John Benjamins.

Strevens, P. (ed.) 1978. *In Honour of A.S.Hornby*. London : Oxford University Press.

Suen, C.Y. 1994. 'Automatic Recognition of Handwritten Characters', *NATO ASI series. Series F, Computer and System Sciences*, 124, pp.70-80.

Suen, C.Y., Guo, J. and **Li, Z.C.** 1992. 'Computer and Human Recognition of Handprinted Characters by Parts', in Impedovo, S. and Simon, J.C. (eds.), pp.223-236.

Suen, C.Y. *et al.* 1993. 'Building a New Generation of Handwriting Recognition Systems', *Pattern Recognition Letters*, 14, iss 4, pp.303-15.

Tang, Y.Y. and **Suen, C.Y.** 1992. 'Parallel Character Recognition Based on Regional Projection Information (RPT)', *Proceedings, 11th IAPR International Conference on Pattern Recognition*, The Hague, Netherlands, pp.631-4.

Tannen, D. (ed.) 1982a. *Spoken and Written Language : Exploring Orality and Literacy*. New Jersey : Ablex.

Tannen, D. 1982b. 'Oral and Literate Strategies in Spoken and Written Narratives', *Language*, 58, pp.1-21.

Tappert, C.C. 1982. 'Cursive Script Recognition by Elastic Matching',*IBM Journal of Research and Development*, 26, no 6, pp.765-771.

Tappert, C.C., Suen, C.Y. and **Wakahara, T.** 1990. 'The State of the Art in On-line Handwriting Recognition', *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 12, no 8, pp.787-808.

Tomita, M. 1986. 'An Efficient Word Lattice Parsing Algorithm for Continuous Speech Recognition', *Proceedings of the International Conference on Acoustics, Speech and Signal Processing*, pp.1569-1572.

Toussaint, G.T. 1978. 'The Use of Context in Pattern Recognition',*Pattern Recognition*, 10, pp.189-204.

Vlontzos, J.A. and **Kung, S.Y.** 1989. 'A Hierarchical System for Character Recognition', *1989 IEEE International Symposium on Circuits and Systems*, 1, pp.1-4.

Vossler, C.M. and **Branston, N.M.** 1964. 'The Use of Context for Correcting Garbled English Text', *Proceedings, ACM 19th National Conference*, pp.D2.4-1 - D2.4-13.

Wells, C.J. *et al.* 1990. 'Fast Dictionary Look-up for Contextual Word Recognition', *Pattern Recognition*, 23, no 5, pp.501-8.

Wheelwright, G. 1996. 'Handwriting Recognition Comes in from the Cold', Canada Computer Paper.

Whitrow, R. and **Higgins, C.** 1987. 'The Application of n-grams for Script Recognition', *Proceedings of the Third International Symposium on Handwriting and Computer Applications*, pp.92-4.

Wilks, Y.A. 1975. 'Preference Semantics', in Keenan, E.L. (ed.), pp.329-348.

Wolf, C.G. 1990. 'Understanding Handwriting Recognition from the User's Perspective', *Proceedings of the 34th Annual Meeting of the Human Factors Society*, pp.249-253.

Wolf, C.G., Glasser, A.R. and **Fujisaki, T.** 1991. 'An Evaluation of Recognition Accuracy for Discrete and Run-on Writing',*Proceedings of the Human Factors Society 35th Annual Meeting*, 1, pp.359-363.

Wolf, C.G., Rhyne, J.R. and **Ellozy, H.A.** 1989. 'The Paper-like Interface', in Salvendy, G. and Smith, M.J. (eds.), pp.494-501.

Xu, L., Krzyzak, A. and **Suen, C.Y.** 1992. 'Methods of Combining Multiple Classifiers and their Applications to Handwriting Recognition', *IEEE Transactions on Systems, Man and Cybernetics*, 22, no 3, pp.418-435.

Yeager, L. 1997. 'Neural Networks Provide Robust Character Recognition For Newton PDAs', published on-line at
http://www.kiva.net/~larryy/ANHR.html

Appendix A

Word Lattice Simulation

This appendix features a selection of the sample word lattices from which the letter substitution database was derived, and the whole of the letter substitution database itself.

The figures given in brackets in the extract from the word lattices are confidence scores generated by the low-level recognition system.

A . 1 - Sample Word Lattices

the - th(0.61), me(0.49), we(0.48), bra(0.47), bp(0.14), ma(0.01),

quick - gird(0.72), wick(0.71), grid(0.70), rick(0.60), rid(0.50), gcd(0.50), rub(0.42), ed(0.42), reck(0.42), rd(0.42), rib(0.41), ouch(0.41), reub(0.41), rich(0.40), reid(0.39), girth(0.38), erich(0.37), if(0.36), uk(0.35), wed(0.35), reich(0.33), red(0.28), rex(0.27), qed(0.27), ok(0.26), ox(0.26), web(0.23), with(0.22), ruth(0.21), reed(0.21), reek(0.21), reb(0.19), orb(0.17), th(0.08), eh(0.03), ow(0.03), oh(0.01), of(0.01), reef(0.01),

brown - bum(0.96), brim(0.95), boron(0.78), browns(0.68), buns(0.66), own(0.66), mum(0.65), boson(0.60), moron(0.58), wren(0.58), worm(0.57), brier(0.57), wrens(0.52), buss(0.51), when(0.50), borer(0.49), brew(0.46), brows(0.46), bmw(0.45), horns(0.44), brow(0.42), broom(0.42), bosom(0.40), bien(0.39), bini(0.38), wins(0.34), bins(0.33), ohm(0.32), boon(0.29), moon(0.28), owns(0.27), boor(0.27), moor(0.26), moen(0.22), open(0.21), mien(0.20), osier(0.20), wier(0.20), moyer(0.20), bier(0.19), mini(0.19), moser(0.17), opens(0.14), bows(0.08), mows(0.07), boos(0.06), woos(0.06), boss(0.05), moons(0.05), moss(0.04), bow(0.04), how(0.04), boom(0.04), wow(0.03), mow(0.03), miss(0.03),

fox - fm(0.99), for(0.82), jon(0.64), bon(0.64), box(0.53), foss(0.51), mi(0.49), boos(0.30), book(0.30), joss(0.26), boss(0.26), wok(0.06), ms(0.00), jo(0.00), mu(0.00), wu(0.00),

jumps - jump(0.86), pumps(0.79), pimp(0.61), pump(0.61), jurors(0.60), gums(0.59), juror(0.53), mrs(0.52), moss(0.50), jeers(0.50), ms(0.49), son(0.47), firm(0.47), skis(0.46), purrs(0.42), junks(0.40), sons(0.40), gems(0.39), pins(0.39), fins(0.39), gins(0.39), puns(0.39), guns(0.39), soon(0.38), norm(0.37), mr(0.33), peers(0.33), ross(0.30), nm(0.28), nooks(0.26), jinx(0.26), purr(0.26), germ(0.22), pinks(0.20), ron(0.20), non(0.20), pens(0.19), fens(0.19), noon(0.18), songs(0.17), nor(0.16),

over - ova(0.75), oven(0.73), aver(0.66), greer(0.59), own(0.54), deer(0.50), giver(0.42), den(0.40), green(0.39), gar(0.38), oar(0.37), dar(0.34), men(0.34), am(0.33), deep(0.25), given(0.23), ma(0.17), ow(0.09), gm(0.08), gap(0.05), rap(0.04), ran(0.04), dan(0.01),

the - me(0.51), we(0.50), dr(0.01),

lazy - lop(0.91), lam(0.75), lamp(0.74), hazy(0.73), low(0.67), bp(0.66), laze(0.59), hop(0.58), lay(0.50), ham(0.50), law(0.50), lays(0.48), lars(0.48), hasp(0.44), lane(0.41), lame(0.40), katz(0.40), haze(0.39), wry(0.36), dry(0.34), how(0.34), hans(0.29), hays(0.29), hams(0.28), wop(0.28), kay(0.26), hay(0.26), haw(0.26), mop(0.25), kane(0.21), hare(0.21), doze(0.15), bye(0.15), mrs(0.10), wow(0.04), ow(0.02), dye(0.02), dow(0.02), of(0.01), mow(0.01), my(0.01),

dog - do(0.58),

they - thy(0.94), buy(0.84), thee(0.81), the(0.75), frey(0.75), bey(0.75), due(0.72), dry(0.64), free(0.61), th(0.61), du(0.61), bee(0.58), wee(0.54), fee(0.52), wei(0.51), beep(0.50), die(0.50), wu(0.47), mu(0.47), weep(0.46), chi(0.45), why(0.43), be(0.41), dice(0.41), chip(0.37), we(0.37), me(0.36), fe(0.34), wry(0.33), fry(0.33), dim(0.33), bicep(0.29), fm(0.28), fbi(0.25), dis(0.25), whip(0.24), wise(0.23), by(0.19), my(0.13), ms(0.03), cbs(0.01),

would - word(0.97), world(0.77), worm(0.72), wold(0.62), sword(0.57), wow(0.52), seoul(0.38), void(0.30), yond(0.29), sided(0.27), soul(0.27), siegel(0.21), sewed(0.18), seeded(0.08), vow(0.05), yow(0.05),

have - haw(0.99), brave(0.91), wise(0.75), bow(0.66), brae(0.64), base(0.60), bra(0.60), boise(0.60), bait(0.59), hose(0.56), knave(0.52), wu(0.51), ku(0.51), due(0.51), kraut(0.51), wee(0.50), ha(0.50), trait(0.49), wit(0.49), die(0.48), kit(0.48), mi(0.47), twit(0.42), hast(0.41), bose(0.40), ho(0.40), we(0.40), tau(0.39), taut(0.39), bout(0.38), mist(0.36), mit(0.35), tow(0.33), mu(0.33), bee(0.32), host(0.31), west(0.31), time(0.30), bit(0.30), but(0.26), try(0.26), wet(0.26), knee(0.26), dee(0.25), trw(0.25), bin(0.25), disc(0.25), diet(0.25), knit(0.24), unit(0.24), me(0.21), du(0.18), meet(0.17), be(0.15), tout(0.14), best(0.12), beet(0.12), met(0.12), bet(0.07), de(0.07), toe(0.05), tin(0.01), dec(0.01), to(0.01), kin(0.01), un(0.00),

had - jim(0.66), jig(0.66), jew(0.66), wok(0.39), ok(0.36), is(0.34), pig(0.33), zig(0.33), gig(0.33), pew(0.33), pus(0.30), gus(0.30), jess(0.29), so(0.29), piss(0.26), pies(0.26), zeiss(0.25), puss(0.24), go(0.16), peak(0.12), peas(0.08), gas(0.07), as(0.06), was(0.06), peek(0.05), peg(0.05), gem(0.05), peed(0.05), sd(0.02), gm(0.01), ed(0.01),

a

few - wu(0.97), fe(0.76), fee(0.75), fir(0.72), fit(0.69), jew(0.67), fm(0.64), we(0.64), feet(0.57), foe(0.55), fur(0.49), for(0.40), jeer(0.39), jet(0.38), jo(0.37), joe(0.36), jut(0.33), wet(0.33), wit(0.32), i(0.32), jr(0.25), fort(0.22), jot(0.18), et(0.01), it(0.01),

drinks - kink(0.80), kivu(0.73), drink(0.69), knew(0.67), wink(0.62), kirk(0.60), disk(0.59), drive(0.54), hints(0.49), disks(0.47), knits(0.46), hive(0.46), dunk(0.46), wive(0.45), dive(0.45), dime(0.44), dusk(0.42), bmw(0.40), desk(0.40), winks(0.36), dusts(0.35), bests(0.34), dents(0.32), write(0.32), dirts(0.32), desks(0.31), besets(0.27), dims(0.24), wish(0.23), dish(0.22), brew(0.22), bribe(0.22), drew(0.22), brett(0.16), writs(0.16), hews(0.12), herbs(0.06), hems(0.05),

before - bore(0.79), wore(0.79), bop(0.73), wop(0.73), bon(0.72), woe(0.72), won(0.72), boy(0.67), bebop(0.64), bone(0.55), more(0.55), mop(0.48), moe(0.39), ma(0.30), ms(0.30), mae(0.07),

we - eve(0.67), ira(0.34), ewe(0.34), erie(0.27), rio(0.02), a(0.01), era(0.00),

get - go(0.50),

there - these(0.81), then(0.76), mere(0.73), were(0.73), ken(0.68), thai(0.51), brae(0.50), dress(0.42), den(0.39), men(0.35), mae(0.32), mess(0.27), bras(0.26), buss(0.04), bun(0.04), dun(0.02), was(0.00),

working - evoking(0.86), wormy(0.60),

people - pore(0.89), pork(0.88), pole(0.87), peale(0.78), ore(0.73), pope(0.73), dole(0.67), gore(0.66), wore(0.66), work(0.65), role(0.64), peak(0.64), owls(0.64), eye(0.60), sore(0.58), peals(0.58), tore(0.58), sole(0.58), orb(0.57), grope(0.57), poke(0.56), dale(0.52), dope(0.52), dome(0.52), rope(0.50), rome(0.50), gale(0.50), wale(0.50), word(0.50), pend(0.48), pops(0.48), sale(0.46), tale(0.45), sorb(0.45), some(0.45), tome(0.45), pod(0.42), pow(0.42), preys(0.42), wok(0.41), ok(0.41), dye(0.41), woke(0.40), reals(0.40), works(0.40), props(0.39), peaks(0.39), doe(0.36), tom(0.35), woe(0.34), end(0.32), prod(0.32), prow(0.32), toe(0.32), grow(0.30), safe(0.26), sake(0.26), take(0.26), read(0.26), rend(0.26), tomb(0.25), wake(0.25), tops(0.25), womb(0.25), dod(0.24), dow(0.23), sod(0.23), sow(0.23), tow(0.23), sob(0.22), god(0.21), gob(0.21), wow(0.21), ow(0.21), rod(0.12), row(0.12), rob(0.12), bmw(0.08), sad(0.08), tad(0.08), saw(0.08), tab(0.07), dad(0.03), dab(0.03), gad(0.01), wad(0.01), gab(0.01),

will - ill(0.66), uk(0.60), well(0.60), till(0.55), sill(0.55), gull(0.48), ell(0.46), em(0.44), gulf(0.40), tid(0.39), tell(0.39), irk(0.39), sell(0.38), reid(0.38), elf(0.38), wed(0.34), self(0.32), rill(0.29), reek(0.28), gill(0.28), used(0.25), ed(0.23), gem(0.22), ted(0.19), reed(0.19), rid(0.02), rd(0.01),

make - snake(1.00), maw(0.92), sow(0.84), sake(0.83), rake(0.75), soak(0.75), saw(0.69), raw(0.59), sob(0.58), snow(0.52), now(0.51), sam(0.44), nab(0.35), mow(0.35), nook(0.34), ram(0.34), nose(0.27), rose(0.27), nob(0.25), nov(0.03), row(0.02),

the - me(0.51), de(0.49), we(0.49), th(0.47), bra(0.35), die(0.34), ma(0.05), do(0.01),

world - wow(1.00), wold(0.91), word(0.76), work(0.75), vow(0.75), wool(0.72), wok(0.67), void(0.61), york(0.46), orb(0.39), wad(0.35), yow(0.34), old(0.27), owl(0.24), yond(0.22), vend(0.14), orr(0.02),

a

better - beds(0.73), bon(0.68), wei(0.67), bed(0.67), boss(0.48), on(0.16), ho(0.16),

place - lace(0.79), pace(0.77), face(0.75), lao(0.73), lo(0.60), owe(0.57), we(0.57), be(0.54), lax(0.50), lab(0.49), poe(0.49), lad(0.49), old(0.47), foe(0.46), po(0.46), pax(0.39), pad(0.36), do(0.34), fad(0.34), me(0.33), ode(0.33), de(0.33), die(0.26), pm(0.09), fm(0.05), odd(0.02), ida(0.01), ma(0.00),

to

live - lute(0.74), wee(0.65), lisa(0.55), bite(0.51), ha(0.49), ma(0.38), lew(0.36), bee(0.36), usa(0.08),

in - or(0.70), is(0.67), en(0.67), ir(0.66), inn(0.50), mu(0.39), on(0.37), wu(0.01), nu(0.01),

which - whim(0.79), whiz(0.34),

one - ore(0.99), me(0.97), are(0.66), gsa(0.63), ow(0.57), we(0.52), or(0.50), gm(0.48), ma(0.48), ape(0.33), go(0.08), ana(0.00), mr(0.00), and(0.00),

of

you - yow(0.99), yore(0.78), wu(0.68), sou(0.51), spore(0.49), ow(0.49), ore(0.48), sore(0.39), sow(0.34), vow(0.33), or(0.30), soy(0.26), spa(0.03), epa(0.02), oh(0.00), em(0.00),

has - bar(0.78), bras(0.74), bray(0.74), hay(0.66), bas(0.66), bay(0.66), ho(0.51), her(0.46), mr(0.42), hoy(0.35), hey(0.34), kay(0.33), boy(0.33), boo(0.32), ms(0.24), my(0.24), by(0.22), dr(0.19), key(0.01), do(0.00),

never - news(0.82), newer(0.76), sever(0.73), meier(0.73), rever(0.72), saver(0.72), swiss(0.66), saves(0.63), sews(0.58), mews(0.57), saws(0.57), sewer(0.57), meyer(0.56), sayer(0.56), mew(0.43), weiss(0.28), stew(0.13),

been - ken(0.94), beets(0.68), ben(0.66), bess(0.61), men(0.61), ban(0.60), mess(0.57), wets(0.57), bass(0.56), teem(0.40), bets(0.39), mets(0.36), bats(0.35), lean(0.29), wan(0.28), keats(0.17), levi(0.11),

to - la(0.73), lo(0.24),

high - brig(0.77), brigs(0.76), bugs(0.70), hid(0.69), bug(0.69), bud(0.68), brats(0.59), bred(0.51), hats(0.50), brew(0.49), wigs(0.46), bobs(0.45), big(0.37), wig(0.37), bid(0.36), hoi(0.34), hop(0.34), bah(0.34), hew(0.34), wah(0.33), bats(0.25), begs(0.21), beg(0.04), bed(0.03), wed(0.03), wok(0.02), bop(0.01), wop(0.00),

school - wool(0.98), stool(0.85), scum(0.75), wok(0.71), woo(0.66), strom(0.66), brook(0.60), book(0.50), slim(0.50), slosh(0.45), boo(0.34), moo(0.34), rum(0.33), wad(0.33), star(0.33), war(0.33), stash(0.31), wash(0.31), stink(0.30), wink(0.29), scar(0.26), slam(0.26), slash(0.26), wind(0.25), wino(0.25), bask(0.07), bash(0.07), mask(0.07), mash(0.07), mink(0.05), rusk(0.05), bad(0.02), mad(0.01), mao(0.01), bind(0.01), bam(0.01), bar(0.01), mind(0.01), mar(0.01),

before - tempo(0.24), mba(0.18), lewd(0.05),

each - oh(0.96), ax(0.81), em(0.73), tax(0.68), ok(0.66), tack(0.61), ad(0.59), ac(0.58), ox(0.57), to(0.55), tad(0.51), tab(0.50), a(0.47), am(0.41), tam(0.36), tabu(0.26), ow(0.01),

woman - worm(0.89), woos(0.70), yeoman(0.67), vows(0.31),

through - though(0.95), thou(0.80), trough(0.78), dough(0.74), kim(0.63), torah(0.62), thor(0.61), bough(0.60), tough(0.58), theorem(0.58), twos(0.55), kid(0.54), brood(0.53), them(0.52), brim(0.52), womb(0.50), trim(0.50), throb(0.49), throw(0.49), thai(0.44), broad(0.43), brash(0.42), known(0.40), more(0.40), wore(0.40), bum(0.40), trash(0.39), mood(0.39), wood(0.39), dim(0.38), briar(0.38), tum(0.37), twas(0.36), morrow(0.35), brigs(0.35), there(0.35), drops(0.34), bomb(0.33), brew(0.32), tomb(0.30), bud(0.30), drew(0.29), mid(0.29), did(0.29), tori(0.27), dud(0.27), don(0.26), won(0.26), torr(0.26), bore(0.26), bred(0.26), mash(0.26), mawr(0.25), dash(0.25), wash(0.25), brown(0.25), kerr(0.24), borrow(0.24), burr(0.24), tore(0.24), widow(0.23), brand(0.23), drown(0.23), boob(0.23), bras(0.23), bugs(0.22), brae(0.22), brawn(0.22), mops(0.22), digs(0.21), wigs(0.21), dope(0.21), mire(0.21), dire(0.21), wire(0.20), doze(0.20), dome(0.20), burrow(0.20), tugs(0.20), drawn(0.19), dregs(0.18), mew(0.13), dew(0.13), bon(0.09), worm(0.07), ton(0.06), wed(0.06), tops(0.05), maim(0.05), dear(0.05), wear(0.05), moe(0.05), doe(0.05), down(0.04), woe(0.04), tome(0.04), mar(0.04), dar(0.04), warm(0.04), war(0.03),

here - bore(0.86), bose(0.75), hose(0.68), bois(0.67), boss(0.52), bins(0.26), mrs(0.20),

will - uk(0.66), well(0.55), irk(0.49), sill(0.42), yell(0.41), seek(0.35), wed(0.35), sited(0.29), swell(0.25), sell(0.24), silk(0.23), iced(0.19), seem(0.10), scm(0.00),

get - got(0.66), go(0.50),

her - mr(0.97), hen(0.67), bus(0.66), bun(0.66), ms(0.47), mi(0.47), hi(0.40), bp(0.35), ben(0.34), dr(0.34), des(0.34), den(0.34), men(0.33), xi(0.01), du(0.00), mu(0.00),

own - outs(0.77), orin(0.77), ohm(0.73), men(0.68), omits(0.64), sits(0.57), our(0.57), den(0.50), osier(0.50), sin(0.49), ores(0.47), seen(0.40), ms(0.40), open(0.40), omen(0.40), ones(0.40), sets(0.38), sis(0.37), mr(0.34), sir(0.32), ow(0.31), orr(0.28), six(0.26), sen(0.25), des(0.25), seer(0.20), seem(0.20), sees(0.20), dr(0.10), sex(0.01),

hot - lid(0.99), lo(0.84), lids(0.80), hide(0.80), hole(0.78), kid(0.75), hid(0.75), low(0.74), lot(0.74), lola(0.72), la(0.66), los(0.64), hoe(0.63), lake(0.63), kids(0.61), lots(0.60), late(0.59), kale(0.59), hale(0.58), lit(0.52), ho(0.51), how(0.50), law(0.50), hal(0.48), kola(0.48), kate(0.40), hate(0.40), ha(0.33), kit(0.28), hit(0.28), has(0.26), haw(0.25), hat(0.25), hats(0.21),

house - howe(0.99), how(0.95), wu(0.90), home(0.85), bus(0.83), muse(0.81), hoi(0.79), brow(0.78), owe(0.74), boise(0.74), boris(0.73), bow(0.72), buss(0.70), wow(0.70), brows(0.67), hoy(0.67), hoes(0.66), bois(0.65), hoses(0.65), bun(0.63), busy(0.63), bows(0.61), haw(0.61), ow(0.60), boys(0.59), mu(0.58), ku(0.57), bores(0.56), hose(0.55), hoe(0.54), brain(0.54), bmw(0.53), braes(0.51), basis(0.50), wren(0.49), orin(0.49), ores(0.46), boy(0.45), bore(0.45), bien(0.44), wore(0.43), by(0.40), bares(0.40), bays(0.39), brae(0.39), hares(0.38), bose(0.38), hays(0.37), basin(0.36), mien(0.35), bases(0.33), boo(0.33), woo(0.32), bam(0.30), boos(0.29), woos(0.29), ore(0.29), woe(0.29), ham(0.28), hams(0.25), my(0.25), bare(0.25), hare(0.23), bay(0.20), hay(0.18), base(0.18),

the - th(0.97), ku(0.89), me(0.87), we(0.51), de(0.50), he(0.50), mi(0.38), hi(0.00),

king - kim(0.88), ding(0.77), wig(0.66), dim(0.65), kong(0.44), big(0.34), mig(0.34), key(0.27), dog(0.07), dey(0.05), do(0.04), dew(0.04), ho(0.03), by(0.00), my(0.00),

was - eras(0.64), ires(0.53), as(0.52), seas(0.40), to(0.34), sew(0.34), ewes(0.25), estes(0.21), sees(0.05), sets(0.04), seer(0.03),

quite - ice(0.93), we(0.90), give(0.73), quits(0.69), its(0.62), pub(0.62), gusts(0.61), guts(0.60), ewe(0.57), wits(0.54), puts(0.52), owe(0.43), pits(0.42), outs(0.42), peste(0.40), pete(0.40), gets(0.34), eve(0.34), poe(0.34), web(0.31), wets(0.29), is(0.28), ovid(0.26), sd(0.25), pie(0.25), ed(0.25), pests(0.23), pets(0.23), qed(0.08), wed(0.02), pew(0.01), ow(0.01), peed(0.01), ox(0.00),

crazy - craw(0.82), caw(0.75), crag(0.63), raw(0.57), crags(0.51), ray(0.50), cry(0.50), cow(0.46), rag(0.43), cage(0.42), nagy(0.41), coy(0.37), rage(0.35), bay(0.34), rags(0.34), my(0.34), paw(0.32), rye(0.31), cog(0.31), bag(0.29), nay(0.26), pay(0.26), nag(0.23), zag(0.23), age(0.22), bags(0.21), page(0.19), nags(0.18), bye(0.14), ow(0.13), nyu(0.12), by(0.06), we(0.04), me(0.03), wu(0.03), ms(0.02), ny(0.01), mi(0.01), mu(0.01), nm(0.01), pm(0.00),

for - fop(0.67), jo(0.50), bop(0.34), bp(0.07), fm(0.05),

expensive - bemuse(0.68), missive(0.68), bruise(0.37), bunnies(0.35), business(0.26), openness(0.26), brownies(0.22),

boiled - bmw(0.92), bled(0.76), blot(0.74), foiled(0.67), boiler(0.66), harlot(0.65), world(0.63), holed(0.61), baler(0.60), bold(0.53), bowl(0.52), bald(0.52), wild(0.51), bawd(0.51), bawl(0.51), balm(0.51), failed(0.50), hailed(0.50), faked(0.41), faded(0.41), haler(0.40), wold(0.32), fold(0.28), hold(0.28), howl(0.27), holm(0.27), mild(0.27), hawk(0.26), homo(0.26), wafer(0.24), faker(0.21), fader(0.21), hobo(0.02),

ham - gm(0.96), jane(0.74), jan(0.73), jon(0.66), joy(0.66), jaw(0.65), gnu(0.64), pam(0.63), gam(0.62), jar(0.50), pane(0.49), pan(0.49), pore(0.49), zan(0.49), gore(0.48), wu(0.48), gasp(0.39), gs(0.35), jew(0.34), paw(0.32), gem(0.32), jose(0.27), pm(0.27), pare(0.26), par(0.26), pen(0.26), gene(0.26), pope(0.26), zone(0.26), zen(0.26), gar(0.26), posy(0.26), gone(0.26), gnp(0.26), gas(0.25), pop(0.25),

A . 2 - The Letter Substitution Database

***a**
>r
>e
>o
>-
***b**
>l
>w
>m
>-
***c**
>e
>t
>a
>-
***d**
>a
>k
>b
>-
***e**
>s
>a
>o
>-
***f**
>r
>b
>j
>-
***g**
>m
>h
>d
>-

***h**
>e
>o
>b
>-
***i**
>r
>u
>e
>-
***j**
>f
>g
>p
>-
***k**
>e
>b
>s
>-
***l**
>h
>e
>d
>-
***m**
>s
>r
>n
>-
***n**
>e
>s
>r
>-

***o**
>e
>r
>a
>-
***p**
>t
>e
>s
>-
***q**
>o
>r
>g
>-
***r**
>s
>e
>o
>-
***s**
>n
>w
>e
>-
***t**
>w
>b
>d
>-
***u**
>i
>r
>e
>-

***v**
>s
>a
>e
>-
***w**
>r
>e
>s
>-
***x**
>l
>o
>h
>-
***y**
>p
>-
>e
>s
***z**
>n
>y
>g
>-

Appendix B

The System Lexicon

This appendix features an extract from the Collins Dictionary which formed the basis of the system lexicon, and an extract from the lexicon itself.

B . 1 - The Collins Dictionary

a n e$I a

A n e$I A

a (PSNULL) @@ a

a (PSNULL) e$I a

a vb @@ a

a prep @@ a

a (PSNULL) (PNULL) a

A (PSNULL) (PNULL) A

a abbrev (PNULL) a.

A abbrev (PNULL) A.

a' adj @$$: a'

aa adj @$$: aa

aw adj @$$: aw

a- prefix (PNULL) a-

a- prefix (PNULL) a-

A1 adj #|e$I#|w?n A1

A-1 adj #|e$I#|w?n A-1

A-one adj #|e$I#|w?n A-one

A4 n (PNULL) A4

A5 n (PNULL) A5

aa n #|a:a: a.a

AA abbrev (PNULL) AA

AAA abbrev (PNULL) A.A.A.

Aachen n #|a:k@@n Aa.chen

Aachen n #|a$:$x%@n Aa.chen

Aalborg n #|@$lb@$r Aal+borg

Aalesund n #|o$:l@@@|s$un Aa.le+sund

aalii n a:#|li$:i$: aa.li.i

Aalst n a$:lst Aalst

Aalto n #|a:lt@$ Aal.to

AAM abbrev (PNULL) AAM

A abbrev (PNULL) A

A abbrev (PNULL) A

Aarau n #|a$:ra$u Aar.au

aardvark n #|a:d@|va:k aard+vark

aardwolf n #|a:d@|w$ulf aard+wolf

Aargau n #|a$:rga$u Aar+gau

Aarhus n #|@rhu:s Aar+hus

Aaron n #|$e@@r@@n Aa.ron

Aaronic adj $e@@#|r$An$Ik Aa.ron+ic

Aaron's n (PNULL) Aa.ron's

A'asia abbrev (PNULL) A'asia

AB (PSNULL) (PNULL) AB

AB abbrev (PNULL) A.B.

Ab n @ab Ab

ab- prefix (PNULL) ab-

ab- prefix (PNULL) ab-

aba n #|@ab@@ ab.a

ABA abbrev (PNULL) A.B.A.

abac n @|e$Ib@ak a.bac

abaca n #|@ab@@k@@ ab.a+ca

aback adv @@#|b@ak a.back

abactinal adj @ab#|@akt$In%@l ab+ac+ti+nal

abactinally adv #|@ab@@k@@s ab+ac+ti+nal+ly

Abadan n @|@ab@@#|da:n Ab.a+dan

Abaddon n @@#|b@ad%@n A.bad+don

abaft adv @@#|ba:ft a.baft

abaft adj @@#|ba:ft a.baft

Abakan n aba#|kan A.ba+kan

abalone n @|@ab@@#|l@@unI ab.a+lo+ne

abamp n #|@ab@|@amp ab+amp

abampere n @ab#|amp$e@@ ab+am+pere

abandon vb @@#|b@and@@n a.ban+don

abandonment n (PNULL) a.ban+don+ment

abandoned adj @@#|b@and@@nd a.ban+doned

abandonee n @@@|b@and@@#|ni$: a.ban+don.ee

abase vb @@#|be$Is a.base

abasement n (PNULL) a.base+ment

abash vb @@#|b@a#s a.bash

abashedly adv @@#|b@a#sIdlI a.bash+ed+ly

abate vb @@#|be$It a.bate

abatement n @@#|be$Itm@@nt a.bate+ment

abatis n #|@ab@@t$Is,#|@ab@@ti$: ab.a+tis

abattis n #|@ab@@t$Is,#|@ab@@ti$: ab+at+tis

abator n @@#|be$It@@ a.ba+tor

abattoir n #|@ab@@@|tw$a$: ab+at+toir

abaxial adj @ab#|@aks$I@@l ab+ax+i.al

Abba n #|@ab@@ Ab.ba

abbacy n #|@ab@@s$I ab+ba+cy

Abbasid n #|@ab@@@|s$Id,@@#|b@as$Id Ab.bas.id

abbatial adj @@#|be$I#s@@l ab+ba+tial

abb n #|@abe$I ab.b

B . 2 - The System Lexicon

a

00001

0306410

0*********

a1

00002

0000039

3188715***

a4

00003

0000025

3193193***

a5

00004

0000016

3195706***

aa

00005

0000114

3197542***

aaa

00006

0000030

3211120***

aachen

00007

0000053

3215138***

aalborg

00008

0000001

3221506***

aalesund

00009

0000000

-1********

aalii

00010

0000000

-1********

aalst

00011

0000000

-1********

aalto

00012

0000000

-1********

aam

00013

0000000

-1********

aardvark

00015

0000001

3221615***

aardwolf

00016

0000000

-1********

aargau

00017

0000000

-1********

Appendix C

The British National Corpus

This appendix offers a 'before and after' view of the British National Corpus.

Section **C . 1** features a brief extract from the BNC as it is in its original form.

Section **C . 2** features the same extract after pre-processing has stripped out all grammatical and morphological tagging and removed all capital letters and punctuation.

C . 1 - The BNC Before Processing

```
</pubStmt>
<srcDesc>
 <biblStr>
  <monogr>
<title>
 Blissed out
</title>
<author n=ReynoS1 born=1964 domicile="London">
 Reynolds, Simon
</author>
<imprint n=SERPEN1>
 <name>
  Serpent's Tail
 </name>
 <pubPlace>
  London
 </pubPlace>
 <date value=1990>
        1990
       </date>
      </imprint>
     </monogr>
    </biblStr>
  </srcDesc>
</fileDesc>
<encDesc>
 <projDesc>
  See the project description in the corpus header for
  information about the British National Corpus project.
 </projDesc>
 <refsDecl>
  Canonical references in the British National Corpus
  are to text segment (&lt;s&gt;) elements, and
  are constructed by taking the value of the n attribute
  of the &lt;cdif&gt; element containing the target text,
  and concatenating a dot separator, followed by the value
  of the n attribute of the target &lt;s&gt element.
</refsDecl>
<tagsDecl>
 <tagUsage gi=c occurs=6412>
 </tagUsage>
```

```
<tagUsage gi=div1 occurs=12>
</tagUsage>
<tagUsage gi=div2 occurs=40>
</tagUsage>
<tagUsage gi=head occurs=52>
</tagUsage>
<tagUsage gi=hi occurs=97>
</tagUsage>
<tagUsage gi=item occurs=4>
</tagUsage>
<tagUsage gi=label occurs=4>
</tagUsage>
<tagUsage gi=list occurs=1>
</tagUsage>
<tagUsage gi=note occurs=4>
</tagUsage>
<tagUsage gi=p occurs=464>
</tagUsage>
<tagUsage gi=pb occurs=89>
</tagUsage>
<tagUsage gi=ptr occurs=4>
  </tagUsage>
  <tagUsage gi=reg occurs=4>
  </tagUsage>
  <tagUsage gi=s occurs=1836>
  </tagUsage>
  <tagUsage gi=sic occurs=2>
  </tagUsage>
  <tagUsage gi=text occurs=1>
  </tagUsage>
  <tagUsage gi=w occurs=34877>
  </tagUsage>
 </tagsDecl>
</encDesc>
<profDesc>
 <creation date=1990>
 See &lt;biblStr&gt; for publication details.
</creation>
<txtClass>
 <catRef target='allAva2 wriAAg3 wriAD922 wriASe1 wriATy3 wriAud3 wriDom7
  wriLev2 wriMed1 wriPP922 wriSam2 wriSta2 wriTAS3 wriTim2'>
 <keywords>
  <term>
   rock music
```

```
    </term>
    <term>
     popular culture
    </term>
   </keywords>
  </txtClass>
</profDesc>
<revDesc>
 <change n=1>
  <date value=1994-11-24>
   1994-11-24
  </date>
  <respStmt>
   <resp>
    Initial accession to corpus
   </resp>
   <name>
    dominic
   </name>
  </respStmt>
 </change>
 </revDesc>
</header>
<text complete=Y org=SEQ  decls='CN001 QN000 SN000'>
<pb n=15>
<div1 complete=Y org=SEQ>
<head>
<s n=0001>
 <w NN1>MISERABLISM <w NN1>MORRISSEY
</head>
<p>
<s n=0002>
 <w PNP>I <w VVB>think <w PNP>I<w VHB>'ve <w VVN>met <w PNP>them <w
DT0>all
 <w AV0>now<c PUN>.
<s n=0003>
 <w PRP>For <w PNP>me<c PUN>, <w EX0>there <w VBB>are <w AT0>no <w
DT0>more
 <w NN2>heroes <w VVD-VVN>left<c PUN>.
<s n=0004>
 <w CJC>And <w AT0>no <w AJ0>new <w PNI>ones <w VVG>coming <w
AVP>along<c PUN>,
 <w PRP>by <w AT0>the <w NN1>look <w PRF>of <w PNP>it<c PUN>.
```

<s n=0005>
<w PNP>It <w VM0>could <w VBI>be <w CJT>that <w DT0>this <w VBZ>is <w AT0>a
<w NN1>time <w VVN>marked <w PRP>by <w AT0>a <w NN1>dearth <w PRF>of
<w NN2>characters<c PUN>, <w CJC>or <w CJT>that <w AT0>the <w AJ0>smart
<w NN0>people <w PRP>in <w NN1>rock <w VBB>are<w XX0>n't <w AJ0>interested <w PRP>in
<w AJ0-NN1>self-projection <w CJC>but <w PRP>in <w VVG>obliterating
<w PNX>themselves <w PRP>in <w NN1>noise<c PUN>.
<s n=0006>
<w CJC>But <w AV0>really<c PUN>, <w PNP>I <w VVB>think<c PUN>, <w PNP>it<w VBZ>'s
<w AT0>the <w NN1>case <w CJT>that<c PUN>, <w PRP>in <w DT0>this <w NN1>job<c PUN>,
<w PNP>you <w VDB>do<w XX0>n't <w VHI>have <w NN1>time <w TO0>to <w VVI>develop
<w NN2>obsessions<c PUN>, <w DTQ>what <w PRP>with <w AT0>the <w AJ0>insane
<w NN1>turnover<c PUN>, <w CJC>and <w DT0>all <w AT0>the <w NN2>incentives <w PRP>to
<w NN1>pluralism<c PUN>.

<s n=0007>
<w AT0>The <w NN2>heroes <w PNP>you <w VHB>have <w AV0>kind of <w VVB>linger
<w AVP>on <w PRP>from <w AT0>a <w AJ0>prior <w NN1>period <w AVQ>when <w AV0>only
<w AT0>a <w DT0>few <w NN2>records <w VVD-VVN>passed <w PRP>through <w DPS>your
<w NN1>life<c PUN>, <w AVQ-CJS>when <w PNP>you <w VHD>had <w AT0>the <w NN1>time
<w TO0>to <w VVI>get <w AJ0>fixated<c PUN>, <w VVB>spend <w NN2>weeks <w VVG>living
<w PRP>inside <w AT0>a <w NN1>record<c PUN>.

C . 2 - The BNC After Processing

miserablism morrisey .

I think ive met them all now .

for me there are no more heroes left .

and no new ones coming along by the look of it .

it could be that this is a time marked by a dearth of characters or that the smart people in rock arent interested in self projection but in obliterating themselves in noise .

but really I think its the case that in this job you dont have time to develop obsessions what with the insane turnover and all the incentives to pluralism .

the heroes you have kind of linger on from a prior period when only a few records passed through your life when you had time to get fixated spend weeks living inside a record .

Appendix D

The Collocation Dictionary

This appendix shows the various stages of the collocation dictionary.

Section **D . 1** features an extract from the intermediate collocation list derived from the BNC.

Section **D . 2** features an extract from the Percentage Score Dictionary deived from the collocation list.

Section **D . 3** features an extract from the Z-score Dictionary also derived from the collocation list.

D . 1 - The Collocation List

2678357 2678357 -4

2678357 2678357 -4

2678357 2678357 -4

2678357 2678357 -4

2678357 2678357 -4

2678357 2678357 -4

2678357 2678357 -4

2678357 2678357 -4

2678357 2678357 -4

2678357 2678357 2

2678357 2678357 2

2678357 2678357 3

2678357 2678357 3

2678357 2678357 3

2678357 2678357 3

2678357 2678357 3

2678357 2678357 4

2678357 2678357 4

2678357 2678388 -2

2678357 2678419 -2

2678357 2678419 -2

2678357 2678419 -3

2678357 2678419 -3

2678357 2678419 -3

2678357 2678419 -3

2678357 2678419 -4

2678357 2678419 2

2678357 2678419 2

2678357 2678419 2

2678357 2678419 2

2678357 2678419 3

2678357 2678419 3

2678357 2678419 3

2678357 2678419 3

2678357 2678419 3

2678357 2678419 3

2678357 2678419 4

2678357 2678419 4

2678357 2678419 4

2678357 2678419 4

2678357 2678419 4

2678357 2678453 -2

2678357 2678589 -4

2678357 2678589 -4

2678357 2678589 -4

2678357 2678589 –4

2678357 2678589 3

2678357 2678589 3

2678357 2678589 3

2678357 2678589 3

2678357 2678589 3

2678357 2678589 4

2678357 2678589 4

2678357 2678589 4

2678357 2678589 4

2678357 2678589 4

2678357 2680800 -3

2678388 2678388 2

2678419 2678419 -3

2678419 2678419 -3

2678419 2678419 -3

2678419 2678419 -4

2678419 2678419 -4

2678419 2678419 2

2678419 2678419 2

2678419 2678419 2

2678419 2678419 2

2678419 2678419 3

2678419 2678589 -2

2678419 2678589 -2

D . 2 - The Percentage Score Dictionary

-1

1 -1 16 0.00522176 0.00522176

1 -2 587 0.191573 0.191573

1 -3 4784 1.56131 1.56131

1 -4 6595 2.15234 2.15234

1 1 45 0.0146862 0.0146862

1 2 671 0.218988 0.218988

1 3 2985 0.974185 0.974185

1 4 1789 0.583858 0.583858

2 1 2 0.00065272 5.12821

2 2 1 0.00032636 2.5641

2 4 1 0.00032636 2.5641

3 -3 1 0.00032636 4

3 3 1 0.00032636 4

3 4 2 0.00065272 8

5 -1 1 0.00032636 0.877193

5 -3 6 0.00195816 5.26316

5 -4 2 0.00065272 1.75439

5 2 1 0.00032636 0.877193

5 3 2 0.00065272 1.75439

5 4 3 0.00097908 2.63158

6 -1 1 0.00032636 3.33333

6 4 2 0.00065272 6.66667

7 -2 2 0.00065272 3.77358

7 3 1 0.00032636 1.88679

7 4 1 0.00032636 1.88679

19 -1 1 0.00032636 2.63158

19 -2 3 0.00097908 7.89474

19 3 1 0.00032636 2.63158

23 -1 3 0.00097908 4.41176

23 -2 2 0.00065272 2.94118

23 -3 1 0.00032636 1.47059

23 -4 2 0.00065272 2.94118

23 2 2 0.00065272 2.94118

23 3 1 0.00032636 1.47059

23 4 3 0.00097908 4.41176

27 3 6 0.00195816 9.52381

37 -1 8 0.00261088 2.26629

37 -2 1 0.00032636 0.283286

37 -3 5 0.0016318 1.41643

37 -4 10 0.0032636 2.83286

37 2 1 0.00032636 0.283286

37 3 3 0.00097908 0.849858

37 4 4 0.00130544 1.13314

38 -1 6 0.00195816 1.20482

38 -2 9 0.00293724 1.80723

38 -3 12 0.00391632 2.40964

38 -4 8 0.00261088 1.60643

38 2 6 0.00195816 1.20482

38 3 5 0.0016318 1.00402

38 4 6 0.00195816 1.20482

40 -2 4 0.00130544 4.54545

40 -4 1 0.00032636 1.13636

40 2 1 0.00032636 1.13636

43 2 1 0.00032636 12.5

45 -1 2 0.00065272 6.06061

46 2 2 0.00065272 15.3846

50 -2 1 0.00032636 10

50 -3 1 0.00032636 10

50 2 1 0.00032636 10

52 -1 1 0.00032636 12.5

53 -1 1 0.00032636 3.7037

53 -2 1 0.00032636 3.7037

53 -4 1 0.00032636 3.7037

53 4 1 0.00032636 3.7037

57 -2 2 0.00065272 15.3846

57 4 3 0.00097908 23.0769

59 -2 6 0.00195816 2.34375

59 -3 10 0.0032636 3.90625

59 -4 6 0.00195816 2.34375

59 2 4 0.00130544 1.5625

59 3 3 0.00097908 1.17188

D . 3 - The Z-score Dictionary

-1

-9 0 0 0

149 32 2.55868 2.72761

306 5 3.16832 3.27155

345 92 4.88658 5.18091

371 228 3.11047 3.5087

396 181 2.38515 2.73513

409 10 2.99093 3.11013

591 306 6.92964 7.43627

610 2 4.13964 4.24889

622 2 4.13964 4.24889

726 3 2.55848 2.64041

787 18 2.82525 2.96569

870 159 8.05916 8.47076

1203 3598 8.41809 9.87274

1331 918 16.8815 17.8253

1482 9 3.40072 3.52362

1536 22 2.87404 3.02542

1591 92 8.49816 8.84979

2113 7 2.64595 2.74814

2327 3 6.42796 6.59183

2330 575 8.90913 9.59304

2467 4 2.66638 2.75558

2511 10 3.16251 3.28465

2568 2 2.61816 2.69541

2643 2 4.13964 4.24889

2778 2 2.61816 2.69541

2898 2 2.61816 2.69541

3366 5 3.16832 3.27155

3465 2 4.13964 4.24889

3515 19 3.05867 3.20542

3519 734 7.79942 8.53895

3533 13 4.10714 4.2552

3752 2 2.61816 2.69541

3974 261 3.39651 3.8233

3979 33 2.94826 3.12497

4103 21844 31.365 34.1105

4247 8 5.76303 5.9282

4282 7 3.68349 3.80441

4300 56 2.61942 2.83189

4524 20 3.784 3.94506

4552 208 3.3646 3.75029

4596 138 3.0032 3.32118

4598 499 4.07332 4.65329

4600 49 2.77707 2.98046

4611 303 4.54415 5.01557

4648 3 4.23625 4.35212

4659 5 2.50286 2.5939

4941 2 2.61816 2.69541

5097 29 2.60776 2.77111

5161 407 6.28124 6.84076

5192 476 11.0501 11.716

5202 14 5.44483 5.62027

5217 10 3.53958 3.66833

5220 32 3.13104 3.30847

5227 30 2.49495 2.65878

5237 9 3.62508 3.75201

5239 2 2.61816 2.69541

5247 250 6.9964 7.46519

5249 7 4.89813 5.04265

5253 27 4.59137 4.78174

5264 3 2.55848 2.64041

5280 32 2.50475 2.6729

5285 11 3.49029 3.6214

5309 28 5.39872 5.60492

5315 601 10.5652 11.2851

5326 57 5.24465 5.49791

5329 4 2.66638 2.75558

5341 280 14.8219 15.4345

5380 4 3.70264 3.81189

5410 26 4.99104 5.18609

Appendix E

The Susanne Corpus

This appendix features a 'before and after' picture of the Susanne Corpus.

Section **E . 1** contains an extract of the corpus in its original, unprocessed form.

Section **E . 2** contains the same extract after processing has stripped away all grammatical tagging and removed capital letters and punctuation.

E . 1 - The Susanne Corpus Before Processing

```
G01:0010a  -      YB    <minbrk>   -        [Oh.Oh]
G01:0010b  -      JJ    NORTHERN   northern [O[S[Np:s.
G01:0010c  -      NN2   liberals    liberal.Np:s]
G01:0010d  -      VBR   are        be       [Vab.Vab]
G01:0010e  -      AT    the        the      [Np:e.
G01:0010f  -      JB    chief      chief    .
G01:0010g  -      NN2   supporters supporter .
G01:0010h  -      IO    of         of       [Po.
G01:0010i  -      JJ    civil      civil    [Np.
G01:0010j  -      NN2   rights     right    .Np]
G01:0020a  -      CC    and        and      [Po+.
G01:0020b  -      IO    of         of       .
G01:0020c  -      NN1u  integration integration .Po+]Po]Np:e]S]
G01:0020d  -      YF    +.         -        .
G01:0020e  -      PPHS2 They       they     [S[Nap:s.Nap:s]
G01:0020f  -      YG    -          -        [m101.m101]
G01:0020g  -      VH0   have       have     [Vf.
G01:0020h  -      RR    also       also     [R:G101.R:G101]
G01:0020i  -      VVNv  led        lead     .Vf]
G01:0020j  -      AT    the        the      [Ns:o.
G01:0020k  -      NN1c  nation     nation.Ns:o]
G01:0020m  -      II    in         in       [P:q.
G01:0020n  -      AT    the        the      [Ns.
G01:0020p  -      NN1n  direction  direction .
G01:0030a  -      IO    of         of       [Po.
G01:0030b  -      AT1   a          a        [Ns.
G01:0030c  -      NN1u  welfare    welfare  .
G01:0030d  -      NNL1n state      state    .Ns]Po]Ns]P:q]S]
G01:0030e  -      YF    +.         -        .
G01:0030f  -      CC    And        and      [S+.
G01:0030g  -      LE    both       both     [LE:G102.LE:G102]
G01:0030h  -      II    in         in       [P:p.
G01:0030i  -      APPGh2 their     their    [Np.
G01:0030j  -      NN2   objectives objective .
G01:0030k  -      YG    -          -        [Po[102.102]
```

```
G01:0030m -     IO      of      of      .
G01:0030n -     FB      non     non<hyphen>     [Ns.
G01:0030p -     YH      +<hyphen> -     .
G01:0030q -     NN1u+discrimination     discrimination  .Ns]
G01:0040a -     CC      and     and     [Po+.
G01:0040b -     IO      of      of      .
G01:0040c -     JJ      social  social  [Ns.
G01:0040d -     NN1n    progress        progress        .Ns]Po+]Po]Np]P:p]
G01:0040e -     PPHS2   they    they    [Nap:s.Nap:s]
G01:0040f -     VH0     have    have    [Vf.
G01:0040g -     VHN     had     have    .Vf]
G01:0040h -     VVNv    ranged  range   [Tn:j[Vn.Vn]
G01:0040i -     II      against         against         [P:u.
G01:0040j -     PPHO2   them    they    .P:u]Tn:j]
G01:0040k -     AT      the     the     [Np:o105.
G01:0040m -     NN2     Southerners     southerner      .
G01:0050a -     PNQSr   who     who     [Fr[Nq:S105.Nq:S105]
G01:0050b -     VBR     are     be      [Vap.
G01:0050c -     VVNv    called  call    .Vap]
G01:0050d -     NP2s    Bourbons        Bourbon
                [Nnp:e.Nnp:e]Fr]Np:o105]S+]
G01:0050e -     YF      +.      -       .
G01:0050f -     AT      The     the     [S[Ns:s.
G01:0050g -     NN1c    name    name    .Ns:s]
G01:0050h -     RR      presumably      presumably      [R:m.R:m]
G01:0050i -     VVZv    derives         derive[Vz.Vz]
G01:0050j -     II      from    from    [P:q.
G01:0050k -     AT      the     the     [Ns:107.
G01:0060a -     JJ      French          French          .
G01:0060b -     JJ      royal   royal   .
G01:0060c -     NNL1c           house   house   .
G01:0060d -     DDQr    which   which   [Fr[Dq:s107.Dq:s107]
G01:0060e -     RR      never   never   [R:t.R:t]
G01:0060f -     VVDv    learned         learn   [Vd.Vd]
G01:0060g -     CC      and     and     [Fr+.
G01:0060h -     RR      never   never   [R:t.R:t]
G01:0060i -     VVDv    forgot  forget  [Vd.Vd]Fr+]Fr]Ns:107]P:q]
G01:0060j -     YS      +;      -       .
```

```
G01:0060k  -    ICSt since since [Fa:c.
G01:0060m  -    NP1s Bourbon    Bourbon    [Ns:S[Nns.Nns]
G01:0070a  -    NN1n whiskey    whiskey    .Ns:S]
G01:0070b  -    YC   +,    -    .
G01:0070c  -    CSg  though     though     [Fa:c.
G01:0070d  -    IO   of    of   [Po:e.
G01:0070e  -    NP1g Kentucky  Kentucky   [Ns[Nns.Nns]
G01:0070f  -    NN1n origin    origin     .Ns]Po:e]Fa:c]
G01:0070g  -    YC   +,    -    .
G01:0070h  -    YG   -     -    [h109.h109]
G01:0070i  -    VBZ  is    be   [Vzp.
G01:0070j  -    RR21 at    at   [Ds:G109[RR=.
G01:0070k  -    RR22 least least .RR=]
G01:0070m  -    RGa  as    as   .
G01:0070n  -    DA1  much  much .
G01:0070p  -    YG   -     -    [111.111]Ds:G109]
G01:0070q  -    VVNt favored   favour     .Vzp]
G01:0080a  -    IIb  by    by   [Pb:a.
G01:0080b  -    NN2  liberals  liberal[Np.
G01:0080c  -    II   in    in   [P.
G01:0080d  -    AT   the   the  [Nns.
G01:0080e  -    ND1  North north .Nns]P]Np]Pb:a]
G01:0080f  -    CSA  as    as   [Fc:G111.
G01:0080g  -    IIb  by    by   [Pb:a.
G01:0080h  -    NN2  conservatives conservative[Np.
G01:0080i  -    II   in    in   [P.
G01:0080j  -    AT   the   the  [Nns.
G01:0080k  -    ND1  South south .Nns]P]Np]Pb:a]Fc:G111]Fa:c]S]
G01:0080m  -    YF   +.    -    .O]
G01:0080n  -    YB   <minbrk>  -    [Oh.Oh]
G01:0080p  -    AT   The   the  [O[S[Ns:S.
G01:0090a  -    NN1n nature    nature.
G01:0090b  -    IO   of    of   [Po.
G01:0090c  -    AT   the   the  [Ns.
G01:0090d  -    NNJ1n opposition opposition .
G01:0090e  -    II   between   between  [P.
G01:0090f  -    NN2  liberals  liberal[NN2&.
G01:0090g  -    CC   and   and  [NP2s+.
```

```
G01:0090h  -      NP2s Bourbons    Bourbon
                  .NP2s+]NN2&]P]Ns]Po]Ns:S]
G01:0090i  -      YG   -      -      [h112.h112]
G01:0090j  -      VBZ  is     be     [Vzp.
G01:0090k  -      RGf  too    too    [Ds:G112.
G01:0090m  -      DA1  little little .Ds:G112]
G01:0100a  -      VVNv       understood understand .Vzp]
G01:0100b  -      II   in     in     [P:p.
G01:0100c  -      AT   the    the    [Nns.
G01:0100d  -      ND1  North  north  .Nns]P:p]S]
```

E . 2 - The Susanne Corpus After Processing

northern liberals are the chief supporters of civil rights and of integration .
they have also led the nation in the direction of a welfare state .
and both in their objectives of non discrimination and of social progress they have had ranged against them the southerners who are called bourbons .
the name presumably derives from the french royal house which never learned and never forgot since bourbon whiskey though of kentucky origin is at least as much favored by liberals in the north as by conservatives in the south .
the nature of the opposition between liberals and bourbons is too little understood in the north .

Appendix F

Input Sentences

This appendix features the collection of sentences used as test data in the experiments described in **Chapter 6**.

Section **F . 1** contains one hundred sentences from the British National Corpus.

Section **F . 2** contains one hundred sentences from the Susanne Corpus.

F . 1 - Sentences from the BNC

factsheet what is aids .

from an infected mother to her baby .

did you know .

the major impact is yet to come .

it can be fun as well .

its hard work but very rewarding .

its all in a good cause .

sponsored disco marathon or football .

do i need any training .

through infected blood or blood products .

there is no limit to the number of ways to raise money .

car boot sale why not have a clear out .

dont plan on selling too much at more than 10p an item .

yes but you are not expected to be a nurse .

internal kaposis sarcoma can be very painful .

the numbers with pain are also higher .

i was a very happy gay man .

letters to the editor would be welcome .

tony has been unwell over the weekend .

still no decision about tony .

meanwhile another cry for help .

i arrive at andrews house .

we make the most of this and scoot off to the hospital .

ive heard of a pub crawl but a hospital crawl .

they finally turn up .

home at last .

a friend can infect you without your knowing .

there is no cure .

friends or partners may soon be ill too .

a new germ enters the body .

the body is seriously infected .

hospital treatment is needed .

some die of the infection .

they all died through aids .

hes infected with the virus causing aids but doesnt know .

what are they .

a year later shes infected too .

now the doctor tells him he has aids .

but ive only had sex with one person you say .

it only needs one person to pass on an infection .

theyre used to these diseases .

dont hesitate to go .

those coloured red are infected .

how can you be sure your partner isnt infected .

why do people get into drugs .

stuff is getting passed round .

it can seem hard to say no .

because shooting your brain to bits isnt worth it .

trading death a big problem drugs are never cheap .

the effects never last long and you always want more .

your friends begin to wonder what is wrong .

things are always on your mind .

drugs and aids drugs damage your body .

why you should say no to drugs .

it could ruin your entire life .

some drugs can hook you almost instantly .

if you say yes that could be your last free decision .

they cant give you a purpose or meaning in life .

youre always free to say no .

if you have aids a germ can destroy your eyesight .

reduce the number of partners you have sex with .

the reality of aids is that the person can die at any time .

how long does a covenant have to last .

do i need to go to a lawyer .

you can pay by cash or by cheque .

the simple answer is no .

which spouse should make the gift aid payment .

how do i make a will .

who will deal with my estate when i die .

i dont know what id do with them .

it is during this time that torture most commonly occurs .

when its a prize in an amnesty raffle .

can anyone offer a holiday cottage .

globally however much work lies ahead .

many of these prisoners are now free .

the police deny its agents were present .

he was finally released on 19 october 1989 .

the victims deserve it .

mind your own business .

we have seen it happen .

change is possible .

the couple returned to malawi in 1981 .

the chirwas say they came to see a sick relative .

she has not been seen since .

he was arrested and jailed for two years in 1977 .

he was denied access to a lawyer for over three weeks .

he was denied sleep for five days on end .

none has used or advocated violence .

he is now held in hospital in safi .

he was arrested again in december 1981 .

the 12 also refused to wear their prison uniform .

we were one in adversity .

do not forget me dear comrade .

that makes me angry now .

a checker game on death row .

her father was sentenced to death .

the guard was still in the house .

but he was alive .

her son was very ill .

she decided to try and visit him .

F . 2 - Sentences from the Susanne Corpus

with the machine went a complete design for the hull .

it was there that the two accused civil servants were at work .

the hull was also a result of almost a decade of work .

the skipjack became the fastest submarine ever built .

reputedly it could outrun underwater the fastest destroyers .

range was a vital detail .

designs of parts were sought .

they on occasion posed as addicts and peddlers .

his losses included his money bag containing to and his paycheck .

then the youths fled with his money .

two tax revision bills were passed .

this is a poor boy bill said chapman .

his petition charged mental cruelty .

the couple were married august 2 1913 .

the mayors present term of office expires january 1 .

they would still be paid by the patient .

being at the polls was just like being at church .

there wasnt a bit of trouble .

he did not say by how much .

dallas and fort worth can vote bonds .

the hartsfield home is at 637 e pelham rd ne .

a similar resolution passed in the senate by a vote of 29 .

davis received 1119 votes in saturday election and bush got 402 .

everything went real smooth the sheriff said .

i am willing to stake my political career on it .

one validated acts of school districts .

this would help the little peanut districts .

a normal year work in college is 30 semester hours .

karns ruling pertained to eight of the 10 cases .

two other cases also were under advisement .

it was defeated in congress last year .

both figures would go higher in later years .

these would be paid for out of general not payroll taxes .

every person will choose his own doctor and hospital .

a number of scattered ayes and nos was heard .

but thus far there has been no response in kind .

some nato nations disagreed however .

that was before i studied law .

dumont spoke on the merit of having an open primary .

it can only rebound to mr hughes discredit .

that too will fail .

im not afraid to tangle with the republican nominee .

she served one four year term on the national committee .

the mayor said it didnt come from me .

its see joe see jim he says .

the hand is out .

but there are reasons for the current spotlight on the subject .

day after day some new episode is reported .

six of these were proposed by religious groups .

i am a missionary .

the hotel owner shrugged .

same thing he said .

the latter two are half brothers .

but no one was overly optimistic .

pfaff succeeds martin burke who resigned .

but from a historic viewpoint none can approach it .

we want to find out who knew about it pratt said .

certain people must have known about it .

he was in baptist memorial hospital .

the left front wheel landed 100 feet away .

martin called for patience on the part of americans .

nobody really expects to evacuate .

the election will be december 4 from 8 am to 8 pm .

polls will be in the water office .

election came on the nominating ballot .

it was ruled a difficult chance and a hit .

then we really have someplace to go .

place kicking is largely a matter of timing moritz declared .

once you get the feel of it there not much to it .

practice helps you to get your timing down .

if you kick too much your leg gets kinda dead he explained .

it didnt monday he had four longhorns in the top four .

the mustangs dont play this week .

we needed it and we got it .

both turned in top jobs for the second straight game .

and he caused the fumble that set up our touchdown .

he really crucified him he nailed it for a yard loss .

he just lay back there and waited for it meek said .

he almost brought it back all the way .

kelsey is very doubtful for the rice game meek said .

he will be out of action all this week .

just our luck exclaimed stram .

in fact our whole defensive unit did a good job .

a quick touchdown resulted .

alusik then moved cooke across with a line drive to left .

mary dobbs tuttle was back at the organ .

i had it he told a newsman .

this was the first word from jensen on his sudden walkout .

he is mad at the world .

but jackie had gone into the station .

i told him who i was and he was quite cold .

but he warmed up after a while .

he said he had never talked to liston .

i cant run .

i cant throw .

suddenly my reflexes are gone .

the record books however would favour the giants ace .

once the figure was 30 .

the crowd of 32589 had only two chances to applaud .

the selection had been expected .