The Text Contains its Own Lexicon: Extracting a Spelling Reference in the Presence of OCR Errors.

A dissertation submitted in partial fullfillment of the requirements
for the Open University's
Master of Science Degree in Computing for Commerce and Industry

By

David Neil Barnes

M3085208

21st February 2011

15,400 words

Preface

I would like to acknowledge the assistance and help of my tutor, Martin P. Lee, and my specialist advisor Dr Alistair Willis, without whom this dissertation would have been very much less focused.

The assistance of Mr Nicholas Hodson of www.athelstane.co.uk, who spent far more time than I could reasonably have expected explaining his software and the way in which the various issues of proof-reading scanned and OCR'd texts might be addressed, is gratefully acknowledged.

I would also like to thank my wife Anita Wegner, who kept me working when otherwise I might have given up, and last but not least, I would like to acknowledge the efforts of my first grand-daughter Maya-Soley Navit Salama to disrupt the writing process with her well-timed and very welcome arrival.

Contents

List of Figures

List of Tables

Abstract

Optical character recognition (OCR) systems produce text output which almost invariably includes errors. Those errors need correcting - either by a human editor or automatically - but the nature of those errors is such that conventional spelling checkers rarely offer a useful tool. While they can usually identify a mistake (a non-word), because their correction models are usually based on human input failures, they are often unable to offer the correct replacement for the non-word, hence the need for a human proof-reader with domain specific knowledge.

A proportion of detected non-words will, in context, be correct - domain specific words, proper nouns, and neologisms. Identifying these correct words and removing them from the list of tokens which need correction can vastly simplify the task of the proof-reader, reducing his workload and increasing his accuracy. Adding them to a temporary lexicon associated with the text under consideration allows their use as a reference for spelling correction.

A synthesis of statistical and lexical methods is used to identify 'correct' words with a high degree of confidence; those identified words are included in an auxiliary lexicon which with a generic lexicon combines to provide a reference for document specific lexical testing. Of the tokens absent from the default lexicon – as many as four percent of the corpus total - between 40% and 50% are revealed to be correct in context and can be removed from consideration. The majority of remaining errors are actual transcription errors from the OCR process, with a small proportion of very rare or 'alien' words. Very few unwanted non-word tokens are added to the auxiliary lexicon - less than 0.1%.

A robust solution to the problem of removing unwanted hyphens from tokens in the OCR text is presented. Consistently repeatable OCR errors are corrected automatically. Supplementary work shows excellent results in preliminary testing to resolve the issue of tokens made deliberately 'alien' by an author which are thereby dissimilar to the training corpus but present in large numbers. A practical example of the techniques used in a desktop application is demonstrated.

1. Introduction

1.1. Background to the research

More and more printed material is being converted to electronic storage. Programmes including Project Gutenberg are scanning books in bulk and making them available on-line. Gutenberg, and other groups such as Distributed Proofreaders, use a large number of volunteers to perform proof-reading tasks, ensuring that the material presented to the user is as close as possible to the original author's intentions.

The rise in popularity of e-book readers (and their close kin, e-book programs on mobile phones, tablets, and computers) requires a supply of formatted texts. The first stage in this process requires ideally an original electronic copy of the text - but until recently publishers have been unwilling to accept electronic submissions, and typesetting a book has required retyping the author's text in dedicated systems. For the majority of books twenty years and older it is likely that no original electronic copy exists; for books fifty years and older it is almost certain. Optical character recognition (OCR) systems are an obvious solution.

A common advertising feature of OCR systems is the percentage accuracy - though it is often unclear what is being measured. Values of 99% or better are expected from clean accurately scanned input documents - but if 1% of characters are errors, this equates to around one error in twenty words. At that level of error, a short text of perhaps sixty thousand words will require three thousand corrections.

Any significant text will contain domain-specific words - those relating to its subject. These might be very rare in general usage, but common in a document, and are

words which should not be marked as errors but which often are. They include not only technical terms but - in fiction - invented words; there may be proper nouns, foreign words, dialect words spelt phonetically, or partial words.

Generic spell checking software in commercial word processors is very good at identifying OCR errors. However, it will generally also classify domain-specific words as incorrect. This is undesirable behaviour for two reasons - firstly because many words are incorrectly marked as misspelt, and secondly because the visual clutter caused by this hides the actual errors which require attention.

Where domain specific words contain OCR errors, the spelling corrector cannot offer useful candidate words to correct them.

Clearly new tools are required to address these problems. This dissertation proposes a tool which both reduces the number of correct-in-context non-words indicated as errors, and provides a lexicon as a reference for later spelling correction.

1.2. Objectives of the research project

The research project is essentially engineering-based. From the problem identified above it provides a solution, synthesised from previous work in a number of related fields.

Correction of transcription errors is not the primary objective of this research; suitable techniques are well-discussed in the literature. This research aims to locate wanted non-words within the text and to catalogue them, demonstrating its usefulness with graphical examples in a test application.

Starting with the assumptions that (a) what is on the original document is what the author intended to be there and (b) important words occur frequently, it is simple to derive from a clean text a list of words which occur more than once but which are not present in a default lexicon. This leaves an irreducible minimum of words of uncertain spelling, since they occur only once[1].

[1] That is, they are known to be correct in the clean text, from our initial assumptions.

Uncorrected text from the OCR system contains transcription errors. The above method of identifying correct words cannot be used, as repeated errors would find themselves identified as correct words. Alternative techniques must be derived.

The objectives of this project are therefore:

- To derive and demonstrate an algorithm which can identify undamaged words in the OCR script and collect them to form an auxiliary lexicon.
- To evaluate the effectiveness of that algorithm by comparing the lexicon generated from training material against that of the clean text.
- To evaluate the effectiveness of that algorithm on OCR'd text other than the training material.

In this dissertation, the following conventions are followed. This usage may differ from that of other authors.

token a sequential group of characters not containing any spaces but which may contain any other characters including non-letter characters and punctuation.

word a token which is a correctly-spelt word in context. It may not generally be considered correct in normal usage, but its provenance as a word has been established either by its presence in the default lexicon or by analysis.

non-word a token which is in neither the default nor an auxiliary lexicon. A token may be a non-word at the start of analysis and be promoted to a word after analysis is complete.

wrong-word a token which is present in one of the lexicons, but which is not correct in context. For example, the token '*hat*' may have been transcribed incorrectly as '*bat*'. Such a token will always be considered a word unless and until further processing is undertaken to identify it; this is a non-trivial task which is not discussed further.

1.3. Summary

The inability of most existing tools to differentiate between actual non-word tokens and contextually correct non-word tokens - identifying all as errors - can result in a cluttered workspace with many tokens unnecessarily marked for the proof-reader's attention. This research attempts to reduce this workload by presenting a method of identifying such tokens; a demonstration application (illustrated in Chapter 3) shows how effective such a process can be in de-cluttering the workspace. The proposed method further provides a list of domain-specific tokens which may be used to inform a spelling corrector.

The next chapter will examine previous and current work in the field - in particular, existing methods of identifying correctly spelt words in the absence of a lexicon which is a fundamental part of the research process. Additionally, it will discuss some of the reasons why spelling error detection and correction is such a difficult process, provide an overview of statistical and lexical methods, and will formalise the research process discussed above.

2. Literature Review

This review of the literature examines how the various issues in the research question have been investigated by previous workers, and how that work informs my own research. There is a large body of associated work which is not examined in detail; this is simply for reasons of space. I have attempted to isolate the main threads and concentrate on the core concepts.

I begin by examining how the English language changes over time, and why it continues to change; I also consider the difficulties of proof-reading and show that supplying technical assistance to a proof-reader improves accuracy.

Shannon's work on noisy communication channels and its influence on spelling error detection and correction over the years is then discussed. Various different statistical methods, including n-grams and the Viterbi algorithm, are considered.

'Dictionary' methods of error detection and correction – some of the earliest spelling correctors – are investigated. In particular, the 'best' size for a lexicon of correctly spelt words is considered, as there are significantly different views on this subject.

I next look at the OCR process in general, with attention to the sorts of errors in transcription which can be expected and the ways in which those errors have been categorised and measured by various researchers.

The relative effectiveness of different techniques is discussed only in comparative terms. Differences in source material, language, and methodology render a direct comparison difficult, and my own method is sufficiently different that in most cases the measurements are not relevant.

An excellent and comprehensive overview of the literature to 1992 is that of Ku-

kich (1992) from which many of the earlier works are referenced. More recently, Liang (2008) surveyed and examined the main techniques used and contrasted their advantages and disadvantages.

2.1. Spelling and how it changes

Mitton (1996) discusses many of the reasons why English spelling is the way it is – inconsistent, difficult to learn, and often unrelated to the pronunciation of a word. One suggested reason is that the English written script was designed to represent a different language – Latin. Another possibility is that English includes words either borrowed from or derived from other languages. He notes the effect particularly of hand-copied documents (pp20), where the copiers either add letters in the imitation of French spelling (or perhaps because they were paid by the letter), or invent new letters to disambiguate Latin minims – sequences of unconnected vertical strokes used in 'm', 'n', and 'u'.

Howard-Hill (2006) suggests that a move towards spelling standardisation was driven not by authors, but by economies of effort in printing-houses. A standard spelling within a particular press permits the easier composition and decomposition of type between the press and the type-cases – often the first task an apprentice was taught – and it seems likely that this would have standardised over time as printers moved about.

Fischer (2005) also observes that English is written in a script not designed for it – Latin – and attributes to this no fewer than seven classes of errors betwixt speech and text, where the written text is unable precisely to enumerate the exact sound or meaning that was intended. Though not explicitly stated, this may be another explanation for changing spelling standards.

Proposals to revise English orthography occur with monotonous regularity. The Spelling Society (Bovill, 2009) has been proposing changes for over a century but resistance is high to such changes. Even so, neologisms occur frequently, there are

obvious spelling differences between for example UK and US English, and authors play games with spelling. One well-known example is the science fiction novel Feersum Endjinn (Banks, 1994) in which one dyslexic character's part is written phonetically from his point of view.

A recent paper (Michel et al. (2011)) has analysed over five million books published between 1800 and 2000; the authors report that although the use of older words declines with time, between 1950 and 2000 the number of English words used doubled; they identified over a million.

Spelling has changed over time and is likely to continue changing. It is also apparent that neologisms have been added to the language with increasing frequency. There are two important points here: first that a generic dictionary is effectively out of date as soon as it is published, and secondly that the correct spelling for a document depends when it was written. A standard lexicon cannot provide full coverage of any significant text.

Wallace (1987) points out that 'proof-reading text is a task few individuals enjoy'. He makes the significant observation that errors are more readily noted by a proof-reader if the error causes the shape of the word to change. This is significant because a system which is attempting to analyse the defining features of a glyph may well select a character with similar features – and thus, similar shape – if the original character is distorted by image noise or other process errors. In such a case, a proof-reader would have more difficulty identifying a non-word without some assistance than if the shape was more obviously changed. (As examples, '*the*' and '*tbe*'.)

Some researchers have considered whether it is possible that a computer based tool can be as good at the task of identifying and correcting spelling mistakes as a human is. Rose et al. (1994) show that there are a number of algorithmic techniques which can improve the disambiguation of text errors, but observe that in all cases, humans do better. They attribute this to human domain knowledge; a wide range of knowledge on a huge number of subjects which allows a human to judge the likeliest

word in almost any context of sufficient length. Jinkerson (1996) noted in a series of experiments that students provided with technical aids – either a physical dictionary, or a spelling checker program – performed better than those without.

Identifying misspelt words provides technical assistance when proof-reading an OCR'd text; the reduced number of non-words after automated lexicon augmentation based on the OCR text contents reduces his workload. However, a number of texts will never be proof-read by a human – large library transcriptions, for example, or technical databases. In such a case, the domain relevant lexicon should permit the better automated correction of text, thus improving the accuracy of search engines both to classify and to return documents.

2.2. Noisy channels, statistics, and error correction

Shannon (1949), (summarising part of Shannon (1948)) showed that any transmission system can be modelled as containing a noise source which in some way affects the data passing along the transmission channel. He showed theoretical maxima for data transmission based on the characteristics of the signal and of the noise; concluding that any desired signal can successfully be passed through a noisy channel provided that the signal has sufficient redundancy to enable its reconstruction.

In (1948) he had already discussed the redundancy present in English; using a 27-character alphabet (the case-free letters, and a space) he concluded a redundancy of approximately 50% over a range of eight characters, calculated by a number of methods all of which gave results of a similar level. This redundancy meant that approximately 50% of the characters in a text could be deleted or changed and yet the text could still be (theoretically) reconstructed. This assumption of useful reconstruction informs the work of many in the field of spelling correction.

The statistical characteristics of most languages are well known; code-breakers have used them for centuries. Reinke (1962) describes how it might be used to decrypt the secret messages of the Roman Caesars by looking for common letters and letter

pairs.

Forney (1973) mentions using the Viterbi Algorithm as a mechanism for providing character selection in OCR processing. The algorithm (in this case) relies on knowledge of the relative frequencies of characters when following one or more preceding characters – as discussed by Shannon. He points out that the input to the algorithm can be anything from the actual bit patterns as scanned by the system to the characters deduced by the system in the absence of any contextual information, and that the more 'raw' the input, the better the output.

By looking at the relative frequencies of groups of two, three, or more sequential letters (bigrams or digrams, trigrams, or generically n-grams) it is possible to take a given n-gram and calculate the likelihood of its appearance in a text. If one finds an n-gram within a token which is of very low probability, one may assume that the token is a non-word, impermissible in the language under consideration. Conversely, one may speculate upon the next character in a sequence, based on its probability of occurrence following the preceding characters, or even of a word in a sentence based on the previous words (see Golding and Schabes (1996) and Golding and Roth (1999)). This is similar – but not identical – to the Viterbi algorithm, which effectively looks at a rolling sequence of n-grams and calculates the probability of the next character rather than triggering an error based on a single rare n-gram.

Peterson (1980) provides examples and introduces the concept of an 'index of peculiarity (IP)', a value calculated for a word based on the n-grams contained within it. His index provides a statistical method of the probability that the n-gram was produced from the same source as the rest of the text. Words below a certain IP are assumed to be valid words, those above the threshold are considered non-words.

Zamora et al. (1981) investigate the process in some detail and show that a procedure using trigrams is extremely good at identifying both non-words themselves and the location of the error within the non-word. Their method compares trigrams in a word against the probability of that trigram appearing in the reference text; when

a word contains two or more trigrams with a probability above a defined level it is assumed to be a non-word. They note that of the possible trigrams only a fraction, usually four or five thousand, appear in English texts. They consider the overlaps of these subsets based on different source texts and find it quite high, and postulate that a sufficiently large text reference should include all legal trigrams. However, they reject the use of such a reference on the grounds that the trigram frequencies (or even their presence) may not be valid across a number of different fields.

Hull and Srihari (1982) investigated further in comparison between methods using either n-grams or the Viterbi algorithm, specifically aiming – in both cases – at a method which can improve the output from a noisy channel using contextual information about the possible transformations caused by that channel. In both cases, they intend to identify and correct non-word errors. Their n-gram method differs from that discussed by Peterson and Zamora in that it does not apply a probability to each n-gram. Rather, an array is built containing all possible n-grams (digrams, in this case) and the values of the array set to one if the digram is found in the reference text and zero otherwise. A word is considered valid only if all the digrams in it have value one in the array.

Yannakoudakis and Fawthrop (1983) present an 'intelligent' spelling corrector which uses a number of complex rules and a large lexicon. They attempt to model the way in which errors are made, but in reverse. Non-words – those not in its lexicon – are identified, and words in a limited section of the lexicon are modified by applying the rules to see if they can match the spelling of the non-word. If they can, the (unchanged) lexicon word is added to a list of candidate replacements, and once all possibilities are exhausted, a Bayesian probability decision is made as to the best replacement choice from that list.

Kernighan et al. (1990) use a noisy channel model to derive a correct spelling on a probabilistic basis with four confusion sets; the probabilities of a sequence 'xy' being entered as 'x', of 'x' entered as 'xy', of 'x' being entered as 'y', and of 'xy' being

entered as 'yx'. These transformations cover those identified by Damerau (1964) – see next section - and therefore can be expected to have the same coverage in spelling errors (their approach deals with the output of the Unix 'spell' program and so assumes every word presented is a non-word).

In more recent work, Brill and Moore (2000) have attempted to improve the noise model used in noisy channel spelling correction. Instead of using Damerau's default transformations as used by many workers (see the next section) they extend it to allow multi-character transformations (for example, 'gs' might be replaced by 'x'). They also observe the relative frequencies with which such substitutions take place in training documents. With that information, they are able to calculate the probability of any candidate word being a correct word (or inversely, a non-word) and thus permit the easy ranking of a candidate word list.

van Delden et al. (2004) show an interesting method based on the presence of words spelt both correctly and incorrectly in the same document. They take a non-word and generate from it each of the one-edit-distance words, discarding those not in the lexicon. They do the same for each two-edit distance word thus creating a list of all words within two edits of the original non-word. Because the load increases geometrically with each edit-distance increase, they use a digram measure to identify errors with larger edit-distance. Their method is claimed to resolve both spelling errors and non-words caused by faulty segmentation.

The increasing use of the Internet and of search engines have provided a ripe source of statistical data. Naturally, there have been attempts to use this data. Jacquemont et al. (2007) extract a number of features including 2- and 3-grams, the position of those n-grams in a candidate word, and whether the candidate word appears in close proximity to other markers on a web page. From that they build their candidate list with reported improvements in the probability of the correct candidate word being included in the candidate list over three common spelling correctors (though they note that the chance of it being the first word offered is somewhat lowered). A further

advantage that the authors offer is that as the context on the Internet changes, so does the context in the spelling corrector; it is therefore linguistically adaptive.

Franz and Brants (2006) at Google have released a database containing over a billion word 1- to 5-grams, based on a trillion (10^{12}) words of web text. Islam and Inkpen (2009a) discuss the practical issues of managing data of this volume and present (2009b) a spelling corrector which uses it. The statistical information allows the correction of errors within a short group of words, using a number of variants of the longest common substring algorithm as measures of difference and similarity between strings. In this case they are effectively correcting an entire word at a time (rather than letters within a word); selecting the database n-gram phrase most closely matched to the phrase under consideration. This has the further advantage that it may correct for wrong-words as well as non-words: '*the cat sat on the*' is probably more common in the 5-gram database than '*the cat set on the*'.

Many different statistical approaches have been tried over the years; they appear to be successful and efficient at identifying misspelt words, though perhaps slightly less so at correcting them except when word-based n-gram approaches are used, as in Islam and Inkpen's method. Where correction is attempted, it often uses a noise model based on human entry of text.

2.3. Lexicons and the distribution of words

A lexicon is a list of words with no supporting information; a dictionary is a lexicon with added metadata, such as part-of-speech or affix data. The use of the two is somewhat interchangeable in the literature; these definitions are used here for consistency irrespective of the word used in any quoted works.

It seems obvious that to detect a non-word one should simply use a lexicon of all the correct words in context. If a word under consideration is not in the lexicon, it is flagged as a non-word in need of correction. This is neither a complete nor sufficient condition in all cases, for reasons which will be discussed, but it is nonetheless the way

in which the earliest spelling checkers worked, and remains a common implementation today.

Damerau (1964) is referenced in almost every paper on the subject as the author of the first practical spelling corrector. He observed that 80% of errors in his test material contained a single mistake; either the insertion or deletion of a character, an incorrect character, or transposition of two characters. Having detected a non-word by failure to locate it in a lexicon, he changes the non-word by applying sequential transformations of those types until a match can be found, if there is a match, and accepts that transformed word as the correct spelling. His method is very highly tailored to, and constrained by, the computer technology of the time – for example, the size of the lexicon or the way the tokens are coded – and as such has limited relevance today. Nonetheless his work, associated with that of Levenshtein (1966) who formalised a method of constructing binary codes capable of reconstruction after similar transformations, continues to inform research. The measure of difference between two strings, the Levenshtein-Damerau distance (often simply 'LD'), is used as a critical metric in selecting candidate words and ordering them.

One critical question in any lexicon method is the size of the lexicon. It seems logical that the larger a lexicon, the more likely a given word will be in it, and therefore the less likely that words will be falsely identified as non-words. While this is true as far as it goes, it also increases the chances that a non-word (in the context of the text) might be identified as a rare word.

Zipf (1932) showed that the frequency of the occurrence of a word in English is non-linear – indeed, the distribution closely follows an inverse power law. This is easily demonstrated using the word frequencies of the British National Corpus (BNC) as tabulated by Kilgarriff (1998) (See Figure 2.1).

In the small sample of the most common words presented, the most common ('*the*') is ten times as prevalent as the twentieth most common ('*at*'). I have observed similar distributions in my own experimentation with digrams and trigrams. Perhaps

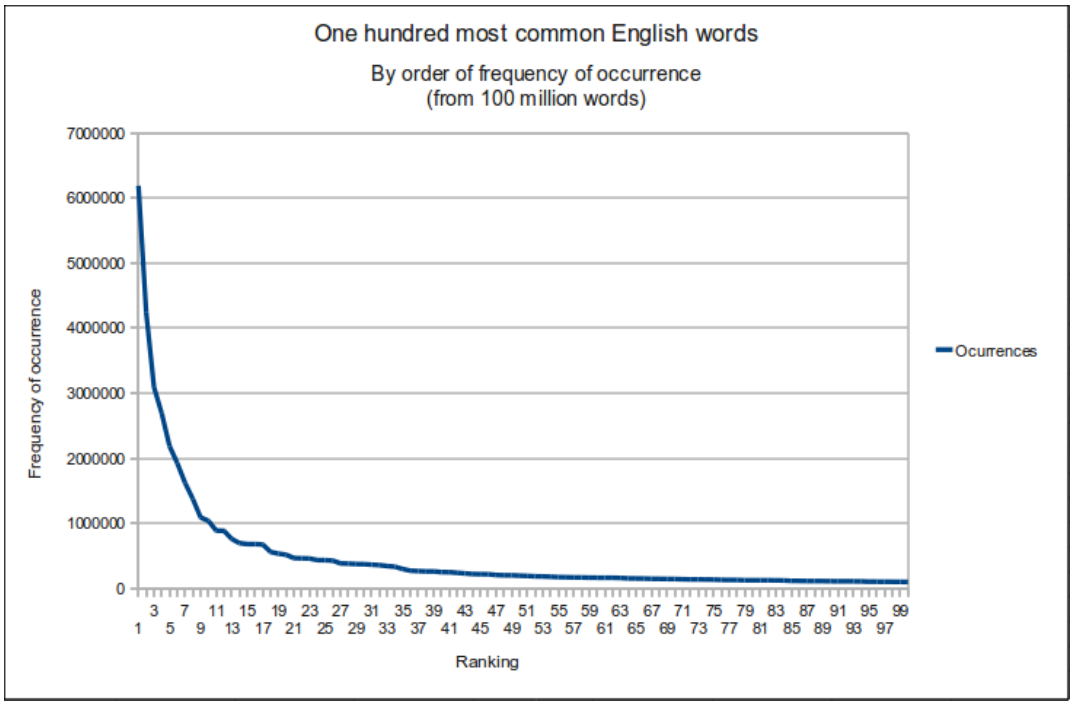

Figure 2.1.: The most common English words from the BNC, from data by Kilgarriff

surprisingly, Li (1992) showed that this effect is also to be observed in randomly created words, and is in fact a mathematical artifact related to the selection of ranking rather than word length as an independent variable. Nonetheless, this is still a useful observation.

Kilgarriff's lemmatised list is restricted to words which appear at least eight hundred times in his sample of one hundred million words; approximately 6,300 different words. Oxford Dictionaries (2010) estimate that English has approximately a quarter of a million words; Michel et al. (2011) estimate four times that. There are obviously a lot of words with very low occurrence. A first thought might be that these words would be randomly scattered throughout a text. However, Church (2000) showed that this was not the case; if a rare word is to be found in a text, the chances of it appearing more than once is greatly increased. De Roeck et al. (2004) point out that distribution of such rare words is non-homogeneous - if a rare word occurs, it is likely for it to occur again close to the first.

The implications of these observations are significant. Firstly, a lexicon to include all possible words (which even then would be limited to the words available at the time the lexicon was created) would be extremely large, with concomitant issues of storage and search speed. Secondly, a relatively small lexicon can be expected to include the majority of words in any given text in that language, but will almost certainly not contain all the words in a text. Thirdly, important words in a text will tend to appear more frequently and closer together than their ranking in a larger corpus would suggest. Where those words are neologisms or proper nouns (possibly fictional) they may not appear in the larger corpus at all.

The question arises of how large a lexicon must be to be useful for identification and correction. Damerau's (1964) corrector was limited by the available hardware to approximately 20,000 longer words. In 1978 McIlroy, discussed in Bentley (1985), used an ad-hoc approach to creating the lexicon by using a number of sources: starting with a dictionary as the main list, he then added many proper nouns – personal, company, place, and object names. The significant part of his approach was to observe correct words which had been flagged as non-words when his program was used and add them to the lexicon – an approach which these days would probably be rejected on grounds of privacy – but which eventually resulted in a lexicon of approximately 75,000 words. Kilgarriff's list suggests that the majority of his words would have appeared fewer than eight times per million.

Peterson (1980) briefly considered this issue and concluded that the lexicon should be restricted in size to reduce the probability of non-words being identified as rare words. In (Peterson, 1986) he investigated in more detail and proposed that the lexicon should be as small as possible, and ideally selected for each document or document class. He proposed a basic lexicon of between 20-40,000 words, based on the dictionaries intended for secretaries.

Damerau and Mays (1989) argue with this conclusion, pointing out that the errors which can occur are not necessarily those which do occur. They suggest that a

larger dictionary – 50-60,000 words – will reduce both false positives and negatives. However, there are issues within their methodology – specifically, they exclude proper nouns and possessives – which may render their conclusion inapplicable in the case of error detection in OCR'd fiction.

Ringlstetter et al. (2007) provide a long and extremely detailed analysis of the issues of lexicon size. They conclude that to be useful the lexicon must contain a high proportion of the words in the document, but note that for many domains the static nature of a lexicon means that "a significant part of the vocabulary is missed". They resolve this issue by searching for current web pages covering the subject domain under correction, and after filtering to remove pages with high spelling error rates, derive a lexicon specific to that domain - see Appendix D for a more detailed explanation. (En passant they suggest that Peterson's conclusions may have been influenced by the size of the lexicons he used; while wrong-words (false friends in Ringlstetter) increase monotonically as lexicon size increases to approximately 100,000 tokens, thereafter it remains stable.) They introduce the concept of a 'perfect dictionary' and compare it with conventional lexicons – showing significant improvements in all cases and particularly in the reduced numbers of false friends (though perhaps surprisingly there are still false friends present). In comparison with other methods, their crawled-web approach is shown to present results better than either large-lexicon or trigram methods, and to closely approach their perfect dictionary.

2.4. Optical character recognition processes

Current generalised optical character recognition (OCR) systems use a number of process stages in the path from a paper (or film) document to computer stored text. Although OCR itself is not the subject of this study (which deals with correction after the OCR process is nominally complete) it is helpful to consider a brief overview of the process to understand the sort of errors beyond those of simple spelling which can be introduced by it.

Takahashi et al. (1990) identify the major phases of the process as scanning (image acquisition), character segmentation, and character/word recognition. A final step is spelling correction. Nagy et al. (1992) document a similar process, including image deskew to ensure the text lines are parallel and also layout analysis to identify text areas. Taghva and Stofsky (2001) include layout analysis – 'zoning' in their terminology – but as a process which will often require human interaction in documents which are generally subject to zoning errors.

Schurmann et al. (1992) consider the overall concept as a process of knowledge acquisition. In their model not only the text itself but its position on the page is important; for them, therefore, the layout analysis is important. Strouthopoulos et al. (1997) discuss problems inherent in layout analysis, including issues such as the skew of the image, dealing with short lines, and identifying image areas. They demonstrate a robust algorithm which uses bounding boxes and a close analysis of small areas of the image to identify areas of the image as text, line drawings, or halftone graphics even when areas of each are non-rectangular. Klink and Jager (1999) attempt to resolve issues of faulty segmentation of text by merging the output of a number of commercial OCR engines and using a comprehensive voting mechanism to select the correct text; they report significant improvements in recognition accuracy.

One approach which has been used by a few workers is based on detecting the word – at the image stage – as a gestalt. Rather than building a word from segmented letters, the shape of the letter is used by Ho et al. (1992) to create a feature vector. The lexicon is also defined by feature vectors and those lexical entries most similar in shape to the word are considered the most likely candidates for it. However, the selection of the actual word requires further analysis.

Lu et al. (2008) evaluate a similar method for document retrieval; rather than perform OCR at all, they calculate a shape index for a word and compare it for closeness with the shape index of the word being searched for.

Kluzner et al. (2009) extend this to locating all similar word images in the original

(large) document. They manipulate the images such that they can identify all the cases where they have the same word; effectively they average the errors on any individual word and only then perform a recognition process on the word. In this way, they can recognise the word once but reliably apply that OCR result at all places in the document where the word shape has been detected.

The importance of identifying errors in the scanned output is considered by a number of workers in the field. Lopresti et al. (1995) consider the differences in error rates which can arise simply from different scanners, slightly misaligned pages, noise, or even just a slightly different procedure in testing, and conclude that while with clean input it takes relatively few pages to obtain repeatable OCR accuracy rates, where the input is less good – they cite photocopied originals – that it may take hundreds or thousands of pages to obtain reliable figures. Since figures of 99% or better are commonly claimed[1], fractions of a percent represent a significant improvement (or otherwise). Kanungo and Haralick (1999) extend this to the point of devising an automatic method of creating ground-truth documents which can then be scanned, processed, and compared with the original. Their method improves the obvious 'print it and OCR it' approach with a text file by including document metadata – the layout and position of the textual and other elements, in a page description language.

Spelling correction with specific regard to OCR output is not well covered in the literature. Takahashi et al. (1990) use an approach of approximate string matching claimed to be capable of correcting all non-words with up to a specified number of errors; Nylander (1999) used a phonotactical approach – tracking the pronunciation rules of the language – together with a number of rules designed to replace observed OCR errors with a corrected version.

Taghva and Stofsky (2001) point out that spelling checkers based upon human input make assumptions regarding the generated errors which are not generally valid in

[1] The exact protocol for defining the error rate is rarely if ever specified, but one might assume that this is with best-practice scans of clean images – or indeed, perhaps of computer generated images without the actual scan. Age of the printed document may also be a factor; certainly my own experience shows a much higher error rate with real-world scans from documents printed only twenty or thirty years ago compared with more recent printing.

OCR text – word isolation from the text stream can be difficult due to inserted spaces and/or punctuation, so broken or joined words are common; character mappings are not always 1:1 (e.g. 'm' may be transcribed 'iii'); and that OCR errors can vary between scanners, OCR software, and even the font in which a text is written. Their program 'OCRSpell' uses an approach which includes both statistical methods and an interactive learning mode whereby a user can correct a spelling and the associated confusion set will be updated with the change – altering the probability on future occasions in the same document. They also present a simple automated confusion set generator.

Reynaert (2008) presents an alternative method which, although intended for use with OCR'd text, takes little account of the actual errors introduced by the OCR system. Instead, a simple lexicon test seeks to generate candidate words within a defined Levenshtein distance, for each entry in the lexicon. The exhaustive list of candidates thus generated is based on a dictionary either specific to the domain, or generic to the language used. The approach is claimed to work largely language-independent, and is intended to work automatically with large scanned corpora.

2.5. The Research Question

The reasons for OCR transcription errors are well discussed in the literature (e.g. Esakov et al. (1994a), (1994b), Lopresti et al. (1995))) and only further research in the various fields of OCR technology will improve the initial fidelity of the conversion. It has been demonstrated (Zamora et al. (1981), Hull and Srihari (1982), Angell et al. (1983)) that non-words can be identified with high precision in the absence of a dictionary or lexicon, and by Peterson (1980) and Ringlstetter et al. (2007) that topic-specific lexicons minimise the number of undetected spelling errors.

Let us define a number of sets of discrete words:

- W_{OT} is the set of all discrete words in the correctly spelt original text.

- W_{ST} is the set of all discrete words in the scanned and OCR'd text.
- W_{DEF} is the set of discrete words in the default lexicon – a commonly available lexicon intended for Linux/Unix spelling checkers.

From these basic words, I derive two sets of non-words, from the sets of words in W_{OR} and W_{ST} which are not in the default lexicon W_{DEF}.

- $W_{ONW} = W_{OT} - W_{DEF}$
- $W_{SNW} = W_{ST} - W_{DEF}$

It is to be expected that in most cases, W_{ONW} will not be the empty set due to domain specific words, foreign language words, and proper names. It is also expected that many of the words in W_{NW} are present more than once – domain specific words and proper names in particular. These repeated words can provide an auxiliary local lexicon W_{OAL} which combined with W_{DEF} closely approaches Peterson's ideal small lexicon.

It should be possible using a number of methods to identify non-words in W_{SNW} which do not contain OCR artifacts and use them to create an equivalent of W_{OAL} which can be used in a similar way. I shall refer to this as W_{SAL}.

The primary research question is how similar it is possible to make W_{SAL} match W_{OAL}, with the intent of reducing the number of author-intended words which would otherwise appear as non-words in a spelling check of an OCR'd text. I will show that W_{OAL} can be easily generated, and will present an algorithm to generate W_{SAL}.

The process of correcting text *per se* is not a part of the research since it has been well-discussed in the literature.

Since we are attempting effectively to automate the acceptance of domain-specific words and proper names in an OCR'd text as being spelt correctly, there are a number of questions which must be answered before we can judge the effectiveness of the solution. How many discrete words from the correctly-spelt version of the corpus W_{OT} remain in W_{ONW}? What proportion of the discrete words in W_{OT} can

be identified as correct, using the method indicated above? Nominally, all the words in this text are correct but not all words not in W_{DEF} are present more than once in the text. Similarly, how many discrete words from the scanned and OCR'd version of the corpus W_{ST} remain in W_{SAL}? How many of these words can be identified as correct? How many non-words are incorrectly identified as correct? The effectiveness of the solution will be judged on the relative proportions of W_{SAL} and W_{ONW}, that is, how well the proposed algorithm(s) work to identify correctly spelt non-words. A secondary question will consider the effectiveness of the algorithm over shorter sources; the corpus contains a number of articles from under a thousand words to full-length novels.

2.6. Summary

This chapter has reviewed the literature in four major areas:

- The history of English spelling, why it continues to change over time, and why the regular introduction of new words and spellings prevents any single lexicon method being fully effective as a spelling checker.
- Statistical methods to describe the errors introduced in any writing system, and a number of methods to identify such errors in English text whether introduced by human or machine mistranscription. Specific consideration is given to error detection using the Viterbi algorithm and to the use of n-grams as a way of generating an index of peculiarity.
- Lexical methods to identify and correct non-words in text, with emphasis on the best size of such lexicons to use. The distribution of the most common words is also examined, and the perhaps non-intuitive observation that very rare domain-specific words can occur far more often in a particular document than their overall likelihood might suggest.

- An overview of the OCR process and of error correction specifically for OCR systems, but where such material exists it has been discussed. It is clearly identified that the errors caused by an OCR system are contextually different from those caused by a human typist, and that existing correction methods are insufficient to repair them.

Finally, the research question is formalised using basic set notation. By considering tokens which match a number of criteria (index of peculiarity, non-presence in a default lexicon, repeated instances etc.) an auxiliary lexicon is created of nominally correct tokens. This lexicon should ideally contain all the correct-in-context non-word tokens and no actual non-word tokens; the research is to discover how closely this ideal might be approached.

This chapter has surveyed a number of existing techniques. Some of these are useful in the proposed research and are further discussed in the next chapter, along with aspects of some errors which may be automatically repaired, and the rules by which a lexicon might be used to accept tokens as correct. The research methods are defined, demonstrated, and analysed.

3. Research methods

In this chapter, the logic behind the development of the process is discussed. The selection and processing of the data prior to formal analysis is detailed; a description of the hardware and software used in the initial data scanning and OCR processing is included. Finally, the detailed approach for measuring the errors contained within the data is considered.

The method is based on a number of observations, generally well discussed in the literature.

- conventional spelling checkers are rarely useful for OCR errors.
- rare tokens are unlikely to be in a lexicon, but are frequently clustered in a document.
- analysis of a token can provide a statistical measure of whether it is likely to belong to the language used in the text.
- errors introduced by the OCR process tend to be either statistically significantly different from English, and/or rare.
- given a suitable lexicon and knowledge of the particular OCR system, algorithms exist to correct OCR errors.

The proposed research is to discover whether identification of 'correct' tokens can be made sufficiently robust to derive a usable auxiliary lexicon. Their selection is based on (a) how frequently such tokens occur and (b) how closely they match, in statistical terms, other tokens in the domain. It involves a number of important phases:

- Identify source material for training and testing, and define the process by which the material will be scanned and OCR'd.
- Select a method to discriminate 'correct' words from non-words.
- Apply the method on the training text and on new material, and analyse the results.

3.1. Source material

The original need for this project is to assist in an on-going project involving scanning, OCR-ing, and proof-reading five hundred issues of a science fiction 'pulp' magazine, *Analog Science Fiction and Fact*.

Two sets of data are required to train the system and evaluate its performance on this reference data - the raw data from an OCR system, and the same data with all errors corrected with reference to the original print material.

The amount of material is chosen to give sufficient scope for statistical analysis, and to ensure that a range of print and scan quality is present in the sample. The training sample consists of all the issues from 1994, and the testing sample all from 1993. Each set is around eight hundred thousand tokens.

3.2. Pre-processing the text

The pages are scanned manually using an *Epson Perfection 2450 Photo* flat-bed scanner. For OCR work it has been found that the software maker's recommendation of 300 dpi works well. The OCR solution chosen is unfortunately no longer available - *Cuneiform V6.0* - but it is maintained for historical compatibility. This Windows program is able to batch scan selected files and produce an RTF format output which is saved as text using Microsoft's *Wordpad* application.

The scanned material still contains a number of unnecessary items. There are page headers and footers, additional editorial material, advertisements, and other

filler material. This is removed manually, without otherwise changing the text or its formatting.

What remains is the basic OCR text. This is duplicated and one copy manually corrected with reference to the original paper texts where required. This gives the two training 1994 reference sources. The same procedure is then applied to the 1993 scan to give the 1993 testing sources.

Although the default language of the corpus is US English, a number of other languages are referenced including UK English, Japanese, Hawaiian, and Italian as well as invented languages such as Martian and a number of phonetic dialects.

3.3. Hyphenation

It is unclear whether a hyphen should be considered an error if it appears in the output text of the OCR process. There are several cases where such a hyphen might appear:

'hyphen-ated' where a word has continued over two lines.

'hyphen- ated' where a line break has been incorrectly inserted after the hyphen.

'hyphenated-' e.g. in speech, where an m-dash has been transcribed as a hyphen.

'hyphenated-words' where a compound word has been created.

'hyph-nated' where a character has been mistranscribed as a hyphen.

Some OCR systems maintain the visual format of the document and require the hyphen to be retained. If the output is to be reformatted, as here, it is not required and can be removed.

As the source material is largely formatted in two fairly narrow columns, there are a large number of hyphenated words - around one percent; several thousand instances. A token which contains a hyphen is considered erroneous until proven otherwise.

Hyphenated words are processed automatically:

- if removing a hyphen results in a token which is present in a lexicon, e.g. *'bl-ue'*, then the hyphen is removed.
- if a token ends with a hyphen, then it is considered with the following token; such tokens are normally the result of incorrect insertion of a line break.
- in all other cases the token remains unchanged.

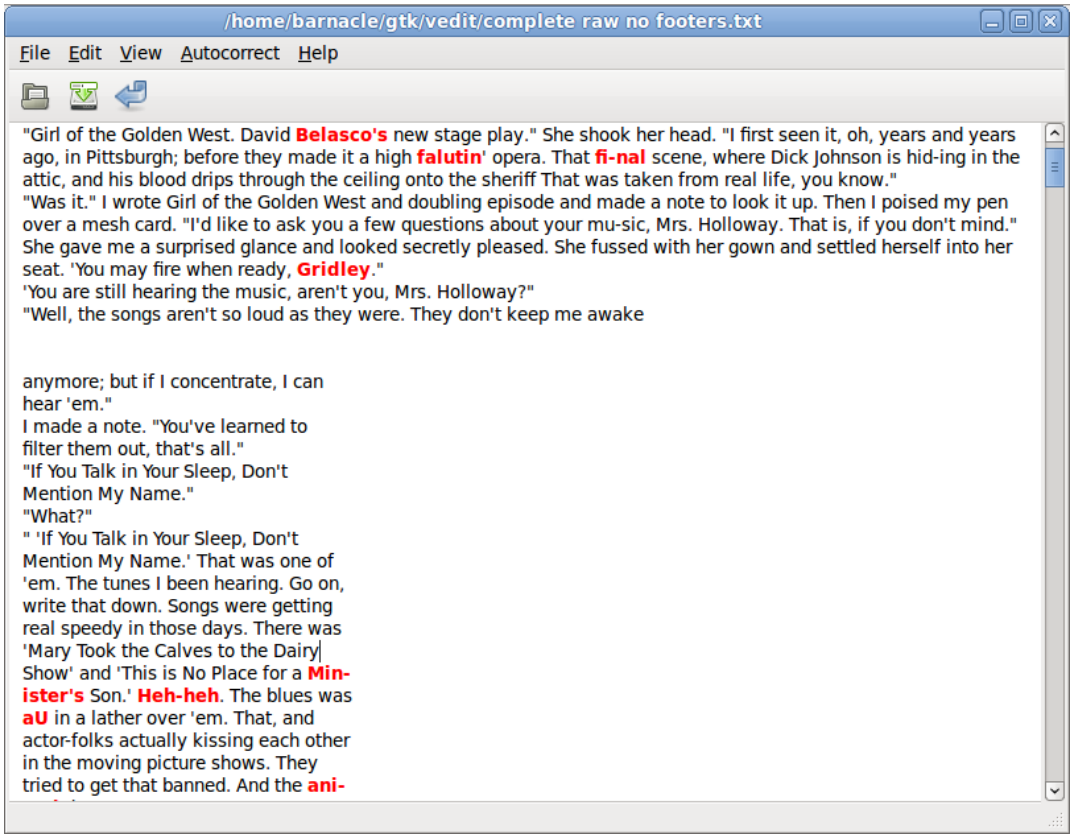

Figure 3.1.: Initial spell-check

Figure 3.1 shows some of the errors[1] due to hyphenation. Five tokens are incorrectly hyphenated; not all are shown as errors since the spelling checker examines each part independently. The de-hyphenation process deals correctly with such cases. Figure 3.2 illustrates the results after de-hyphenation: the corrected tokens are highlighted dark green.

[1] The lower section of the screen-shot shows a common effect, where line breaks have been wrongly introduced for an entire column of text.

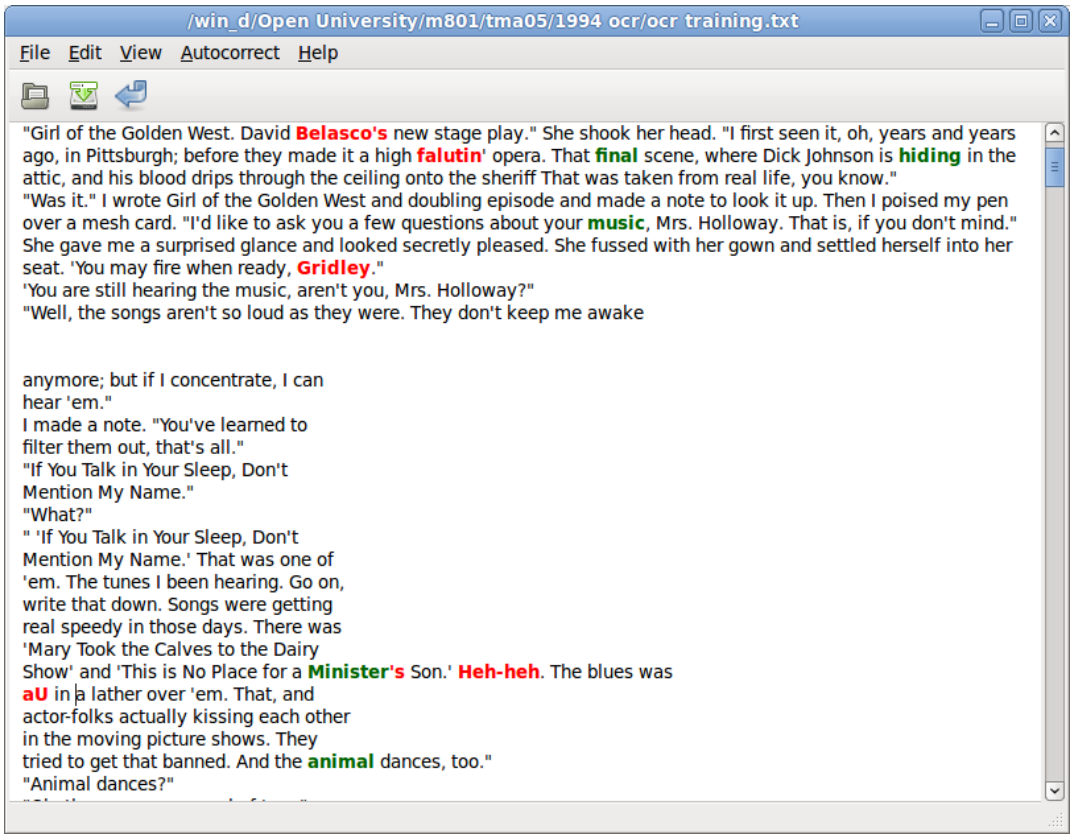

Figure 3.2.: After de-hyphenation

3.4. Spelling and the lexicon

Identifying a token as a correctly spelt word is not as simple as looking to see if it is in the lexicon. In prose, capitalisation is important (this is something which is rarely discussed in the literature); some letters in a word *may* be capitalised and some *must* be capitalised. Sometimes all letters are capitalised - acronyms, titles, or SHOUTING. The token may contain apostrophes or hyphens - e.g. *"Al-Haroun's"*

A number of rules are implemented in the look-up routine.

1. Words are stored in their default case in the lexicon; proper nouns have a leading capital, acronyms have all capitals, and words with multiple capitals are stored with them. They may contain hyphens, apostrophes, or accented characters.

2. A token under test is considered correct if:

a) it matches exactly a word in the lexicon; or

b) its leading capital is changed to lower case and it then matches exactly a word in the lexicon (such as the first word of a sentence); or

c) all letters are upper case, and it matches exactly if they are all changed to lower case; or

d) the token is a fully-parseable number, dollar currency, or a time (so *'42', '-10,000', '$1,000', '15:20', '10e-2'* are all correct words); or

e) the token contains one or more hyphens and all the individual segments are in the lexicon; or

f) removing one of the common suffixes *ed, 'd, ing, ish, 'ld, 'll, ly, 's, s, 've* leaves a word which is in the lexicon.

Figure 3.3 on the following page shows the result; the auxiliary lexicon has not yet been populated and so a number of alien names are indicated as incorrect.

3.5. Using n-grams to isolate 'correct' non-words

The majority of tokens within the text are in the default lexicon. Of approximately 800,000 tokens, around 15,000 tokens in the corrected text are absent from the lexicon. These are all wanted words by definition. Approximately twice this number remain in uncorrected OCR texts - the difficulty is to differentiate the wanted from the unwanted.

Using the relative probabilities of n-grams appearing within text to identify tokens as correct or otherwise is well discussed in the literature. Some methods use only the presence or not of a trigram in the training material (e.g. Hull and Srihari (1982)) while others (e.g. Zamora et al. (1981)) use a combination of sequential probabilities of n-grams. A more complex approach, used in the Unix TYPO program discussed by Peterson (1980), considers the relative probabilities of both bi- and trigrams within a token and compares those to a list generated in an initial scan of the document.

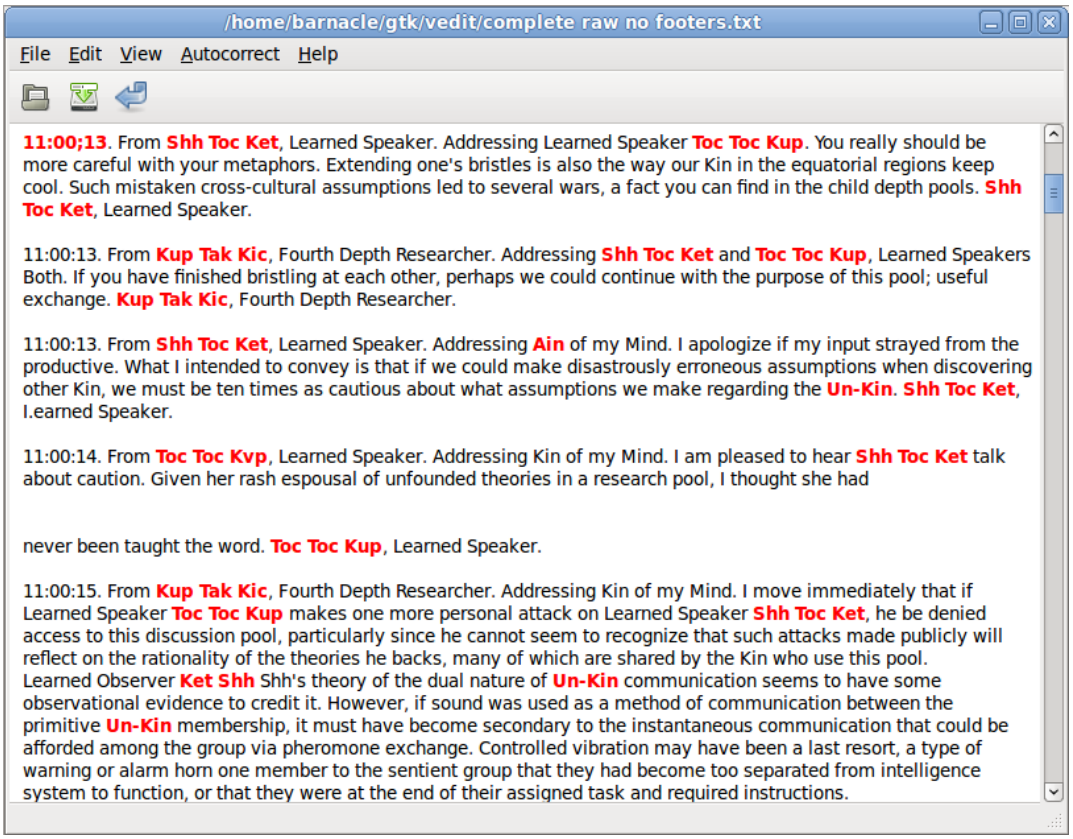

Figure 3.3.: Before the auxiliary lexicon is populated

N-gram probabilities are used to estimate a token as a word or otherwise. The training text is analysed for the relative numbers of trigrams, including case, non-alpha characters, and terminating spaces. Hull and Srihari report that bigrams provide insufficient discrimination, so I generate the index of peculiarity (IP) for a given token thus:

$$I_p = \frac{-log(P_{trigram[0]}) + ... + -log(P_{trigram[n]})}{n}$$

where n is the number of trigrams in the token and the probability of a particular trigram is $P_{trigram[n]}$. Each token is bounded by a space which is included in the calculation, and the letters' original case is maintained. By taking the negative logarithm of the probability, a positive number is generated which is larger for a more unlikely digram; by adding these logarithms together, the probabilities are

being multiplied. If a particular trigram does not exist in the training text it is given a default value of 10. The division by n normalises for token length.

This method is different from that of Zamora et al. (1981) in that it assumes a generalised IP for the entire token; their method observes that an error must affect a number of sequential n-grams and triggers a token as an error if sequential n-gram values are above a predetermined value. This does not help in this situation, since the aim of the IP analysis is different. Where Zamora et al. want to identify a token as definitively incorrect, the aim here is to define it as definitively correct; this requires further information. This is also why the binary approach of Hull and Srihari is not useful in this context; while the presence in the token of an n-gram which is not in a reference list definitively identifies a non-word, a token may contain only 'approved' n-grams and still be a non-word.

Evaluations are made of lexical and non-lexical tokens for each text, and the results arranged in order of peculiarity. They may be graphed to reveal a characteristic shape. Figure 3.4 on the next page demonstrates this for trigrams of all the discrete tokens in the texts.

Shown are four sets of values; one each for the lexicon word set of both the corrected (clean) and original texts, and one each for the non-words from each text. The x-scale merely represents a ranking IP position; the same word does not appear in the same place on each trace.

A large proportion of all the words - in each set - have similar IPs - somewhere between 3 and 4. There are a few very common words and a somewhat larger proportion of very uncommon words in all cases. None of the corrected words have an IP greater than 6; those of the original text rise in a few cases to 10. There are more discrete lexicon words in the uncorrected text than the corrected, largely due to tokens created by the OCR process inserting false line breaks at hyphenated words.

The same characteristic shape is apparent in all four curves. Each has a few instances of highly peculiar tokens, a large number of increasingly non-peculiar to-

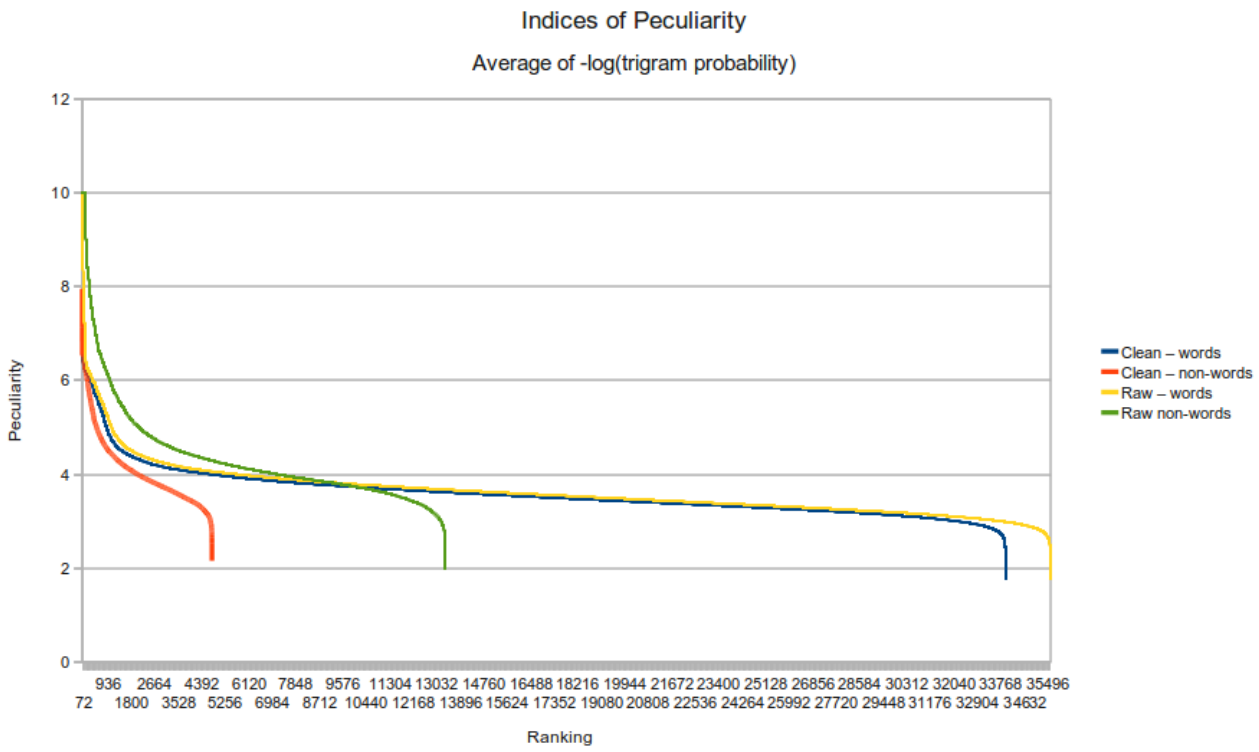

Figure 3.4.: Indices of peculiarity - trigrams

kens, and a few extremely non-peculiar tokens. It might be thought that these low-peculiarity tokens are in fact correct, and simply not in the lexicon, but this is not the case: for the raw tokens, they are extremely likely to be fragments of words resulting from incorrect segmentation of words in the OCR process.

An issue also relates to hyphenated words: in general a hyphenated word is not significantly different in its IP from correctly spelled words - many of them have a value of between 2 and 3. Within the non-words, they are closely clustered with the real but non-lexical words which need to be isolated. They are therefore removed automatically where possible using the approach discussed earlier in section 3.3.

3.6. High peculiarity tokens

High peculiarity values are highly correlated with non-words - in the highest fifteen hundred tokens only those in table 3.1 are 'correct' words.

Token	Instances	Ranking	Index of Peculiarity
MURPHY'S	1	131	8.977175
BRITISH	1	170	8.525378
SEOUL	1	178	8.443068
NANOTECHNOLOGY	1	203	8.225325
ATLANTA	1	674	6.416359
JERRY	1	733	6.32999
MARCUS	1	790	6.225883
ROBERT	1	812	6.190963
THOMPSON	1	828	6.163821
VICKI'S	1	837	6.151026
Mulcahy	1	1304	5.371757
Przbycki	3	1305	5.371479
Zelazny	1	1362	5.276803
Tetsuo	1	1458	5.142223

Table 3.1.: Highly peculiar 'correct' tokens

As the IP reduces, more and more real words begin to appear. An arbitrary point must be selected to define 'real' words; I have selected an IP of 4.5 as being a practical maximum for a 'correct' word based on examination of the lists.

An assumption is made that if any token appears more than once, it is correct. However, this is likely to include a number of non-words whose IP is sufficiently low that they fall in the range of acceptable words. Table 3.2 below lists the only tokens

in the resulting lexicon which are not to be found in the corrected auxiliary lexicon.

Token	Index of Peculiarity	Occurrences	Word	Comments
adair	3.819379	3	affair	
ai	4.261336	3		broken word
alford	3.586041	5	afford	
alraid	4.332112	4	afraid	
alt	3.390985	2	off	
altair	3.550056	2	affair	
anally	2.906311	2	finally	
anil	3.583926	2	and	
atraid	3.289022	2	afraid	
briedy	3.60455	5	briefly	
brietly	3.469609	2	briefly	
colfee	3.954391	2	coffee	
diferent	3.111904	18	different	
dificult	3.663412	3	difficult	
dilference	3.725739	3	difference	
dilferently	3.670228	2	differently	
ditferent	3.661818	2	difference	
ditterence	3.03601	3	difference	
ditterent	2.97973	2	different	
Dont	3.404069	29	front	

Token	Index of Peculiarity	Occurrences	Word	Comments
elfect	3.829185	2	effect	
elfort	3.596998	2	effort	
Erom	4.355966	13	from	
Eront	4.173204	3	front	
ewart	4.081521	8	effort	
fice	3.073152	2	free	
fiom	4.265288	4	from	
fo	4.431531	2		broken word
Gower	3.745029	3	flower	
hant	2.738633	3	front	
hee	3.093535	2	free	
helds	3.401771	4	fields	
hercely	3.134201	2	fiercely	
hiends	3.314295	2	friends	
hola	3.740616	3	holo	
hont	3.162578	4	front	
intrans	3.059657	2	introns	
Irom	4.259965	20	from	
irom	3.609864	11	from	
Iront	4.096404	2	front	
jake	4.186164	2		jake, Jake

Token	Index of Peculiarity	Occurrences	Word	Comments
kame	4.199409	4	frame	
kant	4.052463	4	front	
lang	2.928135	2	long	
le	2.804343	2		broken word
liat	3.197686	2	flat	
lier	3.035064	3	her	or Her
llie	4.196497	2	the	
lliere	3.675269	2	there	or There
lustration	3.121233	3	frustration	
mufled	3.91633	2	muffled	
ni	4.271045	2		broken word
ofhere	4.245723	2	of here	
oflice	3.717191	4	office	
oflicer	3.625185	2	officer	
oificer	3.881631	2	officer	
oisces	3.55997	2	offices	
olf	4.088715	20	off	
olfices	3.809145	2	offices	
om	4.002425	2		broken word
orat	4.002425	2	off	
overed	2.702974	14	offered	

Token	Index of Peculiarity	Occurrences	Word	Comments
Pribhal	4.218672	2		broken word - Pribhala
Rom	3.812939	141	from	
shuflled	4.477705	2	shuffled	
stalf	3.198238	2	staff	
stulf	3.93913	8	stuff	
swered	2.943668	2	suffered	
tarn	3.407649	13	torn	
Teah	4.037812	3	Yeah	
teil	3.932626	2	tell	
th	2.160235	2		broken word
Tou	3.826014	9	You	
Tou're	3.620015	6	You're	
traflic	3.892689	2	traffic	

Table 3.2.: Tokens present in OCR auxiliary lexicon but not in the corrected auxiliary lexicon

Inspection of the original text reveals that the majority of them have a single word as their source; the major exceptions being the two-letter non-words '*ai', 'fo*', '*le', 'ni*', *'om*', and '*th*' which are from fragmented words. '*Pribhal*' is a truncated word; '*tarn*' is rare in English (in Kilgarriff's BNC list (Kilgarriff (1998)) it appears only thirty-eight times in ten million words) and appears here in every case as a mistranscription of '*torn*'. The words '*jake', 'lier*', and '*lliere*' each appear as replacements for two words - capitalised or otherwise - with no apparent way to identify which is

intended. It will be noted how many of these non-words arise from mistranscription of the '*ff*' or '*fi*' ligature.

Damerau (1964) observed that the redundancy of the English language makes it difficult to correct tokens of fewer than five characters. It has been noted elsewhere in the literature that longer tokens are simpler to correct; presumably because they contain a greater proportion of correct characters. For this reason, tokens of fewer than three characters are not excluded from the auxiliary lexicon.

Each of the remaining seventy or so tokens is always mistranscribed from the same correct word. '*Rom*', for example, occurs 141 times to replace '*from*'. This provides an interesting opportunity; to use a table of known non-words and their correct replacements. Such a table could be used twice; the first time to exclude such words from the auxiliary lexicon, and secondly to provide a single replacement word option when correcting the text. This table is held in an external editable file to allow for later changes called 'usual_culprits.txt'.

Of the 820,000 tokens in the training text, this process identifies 1,259 different non-word tokens (10,316 in total) as being correct. Only four (nine instances) are incorrectly marked as words.

Inspection of the remaining non-word tokens reveals a number of other 'usual culprits', each of which either has a high IP score or is of short length, but all of which are common (often more common than those 'usual culprits' with lower IP scores) and derived from a specific incorrectly transcribed root word. Since the purpose of the exercise is to minimise the number of tokens which must be inspected after processing, adding these to the 'usual culprits' list is a reasonable optimisation. The following table lists these.

Token	Index of Peculiarity	Occurrences	Word
oE	10	249	off
&om	7.63093	125	from

Token	Index of Peculiarity	Occurrences	Word
&am	7.782343	61	from
aU	10	37	all
stuE	6.55956	31	stuff
coEee	7.114605	30	coffee
ofhce	6.985818	19	office
oEer	8.028563	18	offer
reaUy	7.07276	18	really
eEect	7.28906	16	effect
kom	5.236731	15	from
diEerent	5.476704	12	different
&owned	5.391016	12	frowned
aEord	7.297501	10	afford
FinaUy	6.948775	8	Finally
finaUy	6.60914	8	finally
staE	6.390294	8	staff
suEering	5.437038	8	suffering
caU	7.572431	7	call
eEort	7.26337	7	effort
5rom	6.471304	6	from
Qoor	5.630109	6	floor
aEair	7.290282	5	affair

Token	Index of Peculiarity	Occurrences	Word
smaU	6.776987	5	small
eEorts	6.610864	5	efforts
ofhcer	6.333707	5	officer
actuaUy	6.278497	5	actually
stiE	6.437328	4	stiff
dil5cult	6.067293	4	difficult
hgure	5.902968	4	figure
&action	4.902002	4	fraction
weU	7.551446	3	well
ol5ce	7.273524	3	office
cliE	6.751776	3	cliff
eEects	6.678249	3	effects
stiU	6.437328	3	still

Table 3.3.: 'Usual culprits' added by inspection

With this in mind, then, the final selection criteria for tokens to be included in the auxiliary lexicon are:

- token is not in default lexicon
- token appears more than once in original text
- token is not in usual_culprits.txt
- token has no hyphens
- token has three or more letters

- token has an IP of 4.5 or less

Having generated the auxiliary lexicon, automated replacement of hyphenated tokens is trivial, as is replacing tokens on the "usual culprits" list. After performing these steps, the remaining non-words and the auxiliary lexicon are output for analysis.

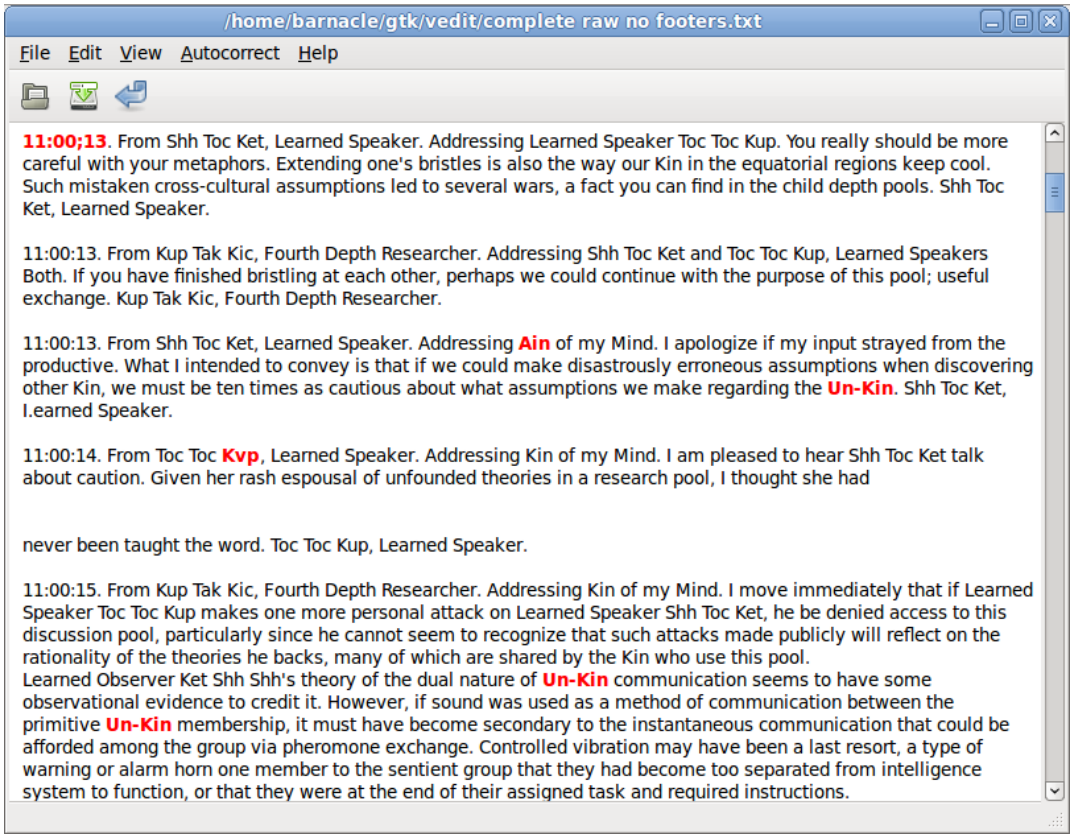

Figure 3.5.: After the auxiliary lexicon is populated

Figure 3.5 shows the effect of allowing the spelling checker to include the auxiliary lexicon. Compare with figure 3.3 on page 29 to see how the spelling errors - previously hidden in the clutter - are now more clearly visible. *'Un-Kin'* is correct in context, but is excluded from the auxiliary lexicon due to the included hyphen.

3.7. Summary

From the wide variety of approaches discussed in the literature, those best able to inform the research method are identified and discussed. A number of those approaches are combined to provide a robust method of locating non-word tokens which are correct in context. The problem of hyphenated tokens and their automated correction is discussed and a method defined, and the way in which a token must be compared against a lexicon entry is detailed.

An algorithm to calculate an Index of Peculiarity for a token is presented, and its use to define the likelihood of a token being part of a corpus is demonstrated graphically using the probabilities of trigrams across the whole corpus. A small number of problem words are noted which are reliably mistranscribed but whose mistranscription results in tokens with a low IP; a method to handle these cases is proposed. Finally, the algorithm by which a token will be defined as being a 'correct in context' word is defined.

It might be argued that the process is somewhat *ad hoc*, and to some extent this is true. It should be remembered however that the aim of the process is to remove indications of wanted non-words; where this is possible by a mechanical process it is an obvious optimisation. Where a guaranteed correction can be made - either by removing a hyphen or by direct replacement of a 'usual suspect' - then we are in effect modelling the noise channel of the complete printing-scanning-OCR process and we not only remove the visual clutter but replace a non-word with a correct word; again, simplifying the task of the proof-reader.

The decision whether a token is a valid word is more complex than a simple lexicon look-up. Most published literature on the subject ignores case and internal punctuation of a token, yet this is significant in any written text. Where OCR errors - which can introduce case changes and random punctuation - are under consideration, these are important issues. Major word processors can fail to recognise words damaged in this way as errors; yet they must be indicated before they can be corrected. It is

therefore critical that as good a decision as possible is made, even to the extent of including times and currencies. I would agree with a criticism based on the removal of suffixes; the algorithm is insufficiently robust in all cases and in production software should be improved (in that it can, for example, pass the non-word '*Rupertly*' though there are no cases in the test cases where this has occurred).

The next chapter will show the results of the process when applied to four corpuses: the training and the test text in both original OCR and corrected versions.

4. Results

This chapter presents the results of applying the method to the training and the test data and the differences between the two sets of data. A number of potential improvements to the method, in handling rare but important tokens, are discussed, as are notes on the importance of selection of 'usual culprits'. A brief discussion on future development in the field is presented.

4.1. Overall Statistics

Table 4.1 shows the overall statistics of the training and test texts, before and after processing. The uncorrected text has more tokens than the corrected text, due to word fragmentation.

Before processing, about two percent of the tokens in the corrected texts are indicated as non-words. For the uncorrected OCR texts, the proportion of non-words approximately doubles. It can be assumed that around half of the non-words indicated as errors in those texts are in fact correct.

Error counts for the OCR texts refer only to non-word tokens; formatting errors and wrong-words are ignored.

A reminder of the various sets of tokens being discussed:

- W_{OT} is the set of all discrete words in the corrected (ground truth) text
- W_{ST} is the set of all discrete tokens in the scanned and OCR'd text
- W_{DEF} is the set of words in the default lexicon

	Training material		Test material	
	Raw OCR	Corrected	Raw OCR	Corrected
Total tokens in text	820252	818236	802445	801960
Discrete tokens in text	49278	38722	49640	37448
Total non-words in text	33012	15492	36591	19710
Discrete non-words in text	14776	5021	15905	4935
		After processing		
Total hyphenated tokens	24869	4566	20473	3564
Corrected hyphenated words	19669	336	15589	0
'Usual culprits' replaced	1220	0	545	0
Discrete non-word tokens in text	5535	3657	7867	3686
Total non-word tokens in text	7306	5116	15151	10410
Discrete tokens in text	40661	38587	41916	37448
Total tokens in text	819120	818236	801960	800884
Discrete words in auxiliary lexicon	1134	1259	1055	1176
Total words in auxiliary lexicon	9867	10316	8730	9205
Unwanted tokens in auxiliary lexicon	4	0	39	0

Table 4.1.: Token statistics for the reference document

- $W_{ONW} = W_{DEF} - W_{OT}$ - the indicated non-word tokens in the corrected text, before processing
- $W_{SNW} = W_{DEF} - W_{ST}$ - the indicated non-word tokens in the OCR'd text, before processing
- W_{OAL} is the set of words in the auxiliary lexicon, after processing the corrected text
- W_{SAL} is the set of words in the auxiliary lexicon, after processing the OCR'd text

4.2. The training text

A three-step process - generation of the auxiliary lexicon, automated correction of hyphenated tokens, and automatic replacement of the 'usual suspects' tokens - is applied to the original OCR text and its corrected version, to enable comparison.

Unwanted tokens in W_{SAL} are those not present in W_{OAL}.

Appendix E provides a simple estimation of the number of tokens which might be sufficiently damaged by the OCR process and therefore excluded from W_{SAL}.

In the corrected text, 336 hyphens have been incorrectly removed; 19,699 hyphens are removed from the OCR text. A further 1,259 'usual_culprits' have been automatically replaced.

The values from the corrected text indicate the best results which might be expected from the OCR text. Figure 4.1 compares the individual instances of tokens in the auxiliary lexicons and remaining errors in the text as a whole; ninety percent of the possible 'correct' non-words have been identified in the OCR text. 2,190 discrete non-word tokens remain in the OCR text beyond those in the corrected text.

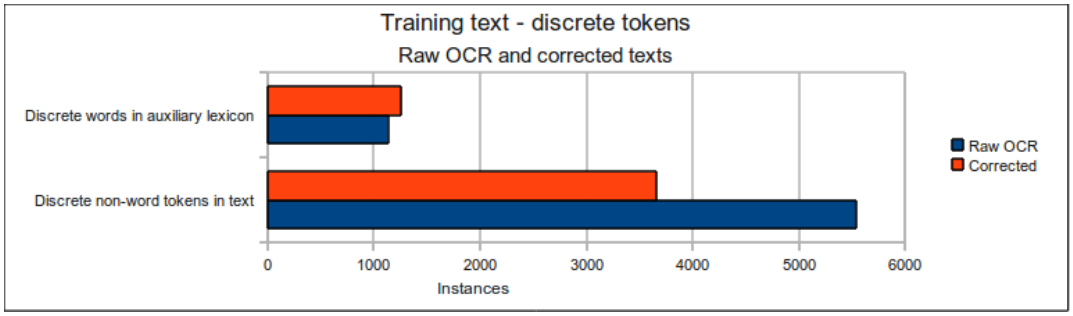

Figure 4.1.: Training text - discrete tokens

Figure 4.2 illustrates the proportions of total token instances and remaining non-word tokens in both of the training texts. W_{SAL} is slightly smaller than W_{OAL} as predicted in appendix E.

Perhaps the most revealing illustration demonstrates the main aim of the research: to reduce the number of displayed errors in the text. Figure 4.3 shows how effective

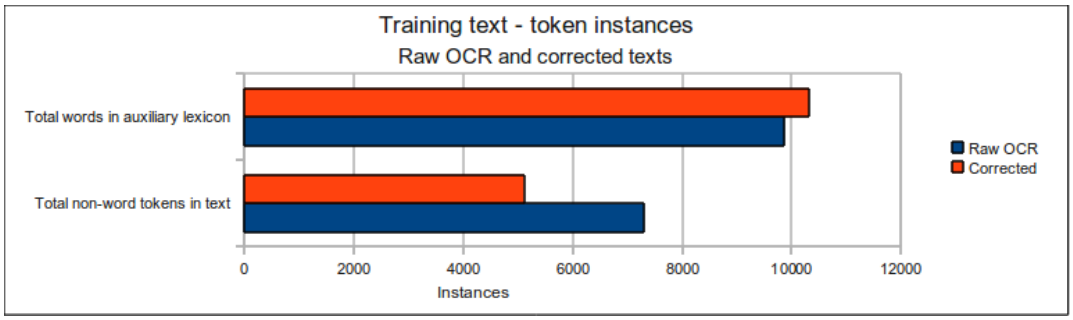

Figure 4.2.: Training text - token instances

this is - in the corrected text, 67% of the errors have been removed. For the OCR text, almost 80% of the errors are shown to be correct in context.

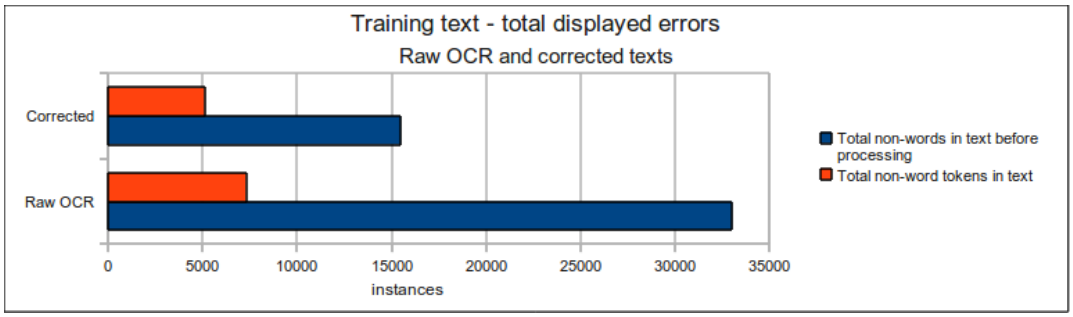

Figure 4.3.: Training text - total displayed errors

Finally, four tokens with nine instances - as discussed in 3.6 on page 31 - remain incorrectly marked as correct.

4.3. The test text

The same measurements for the test texts are shown in figures 4.4, 4.5, and 4.6. The auxiliary lexicons are similar both in size and proportion to those from the training text. While the number of discrete non-word tokens in the corrected text is similar to the training text, there are significantly more non-word tokens in the OCR text.

The total number of tokens in the auxiliary lexicons again are similar in scale and proportion to those of the training text. However, there are approximately twice as many instances of non-word tokens as the training texts (figure 4.5).

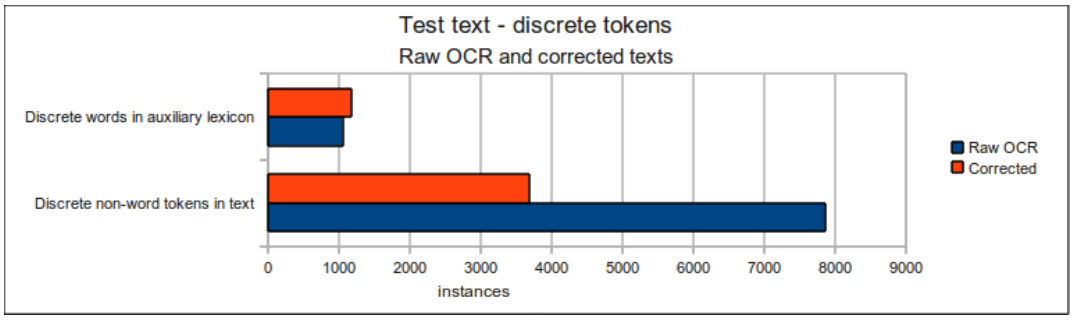

Figure 4.4.: Test text - discrete tokens

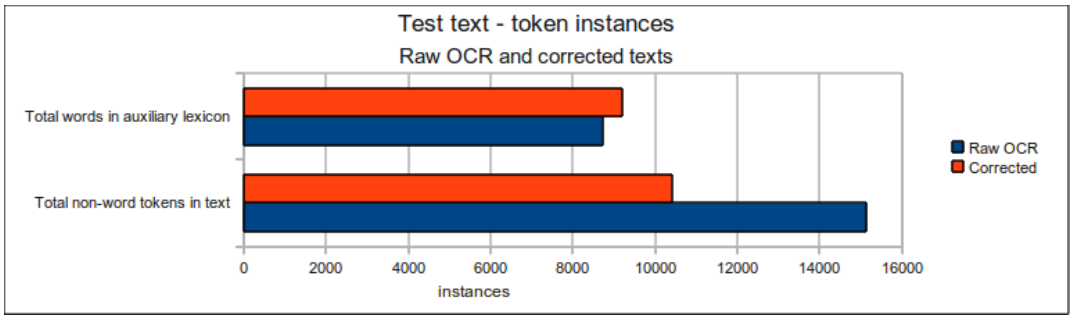

Figure 4.5.: Test text - token instances

For the corrected text, the displayed errors have been reduced by 47%; for the OCR text by 59%.

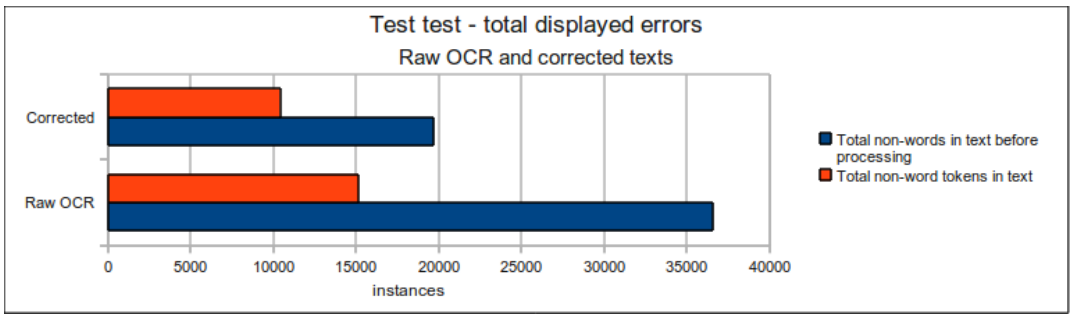

Figure 4.6.: Test text - total displayed errors

A total of 39 tokens are present in W_{SAL} which are not present in W_{OAL}. Some of these - table 4.2 - are consistent mistranscriptions and should be added to the 'Usual culprits' list. The remainder - table 4.3 - are in fact correct, and are missing from W_{OAL} either because a title has not been included in the corrected version,

or because a page footer has not been removed from the OCR version - an error in preparation. In one case, a word has been misspelt in the original print, but corrected in the corrected version. Together these tokens represent 80 instances of incorrect tokens which remain unmarked.

A single entry *'mme'* is in the list which cannot be associated with a particular source word.

Token	Index of Peculiarity	Instances	Word
ahout	3.57112	2	about
alld	3.502303	2	and
amund	3.532179	3	around
anct	3.227267	2	and
appearecl	4.125741	2	appeared
awn	3.647485	2	own
Ciovernor	4.447916	2	Governor
clidn't	3.36061	2	didn't
dients	3.134598	2	clients
Elis	3.665878	2	His
fust	3.329585	2	just
Gare	3.721336	2	flare
hact	3.493958	2	had
heing	2.695166	2	being
herc	3.643024	3	here
hest	2.830896	3	best

Token	Index of Peculiarity	Instances	Word
hetter	2.95022	3	better
hini	3.412687	2	him
keelady	3.682286	2	freelady
lan	2.962989	4	Ian
luve	4.289138	2	have
Melanic	3.769073	12	Melanie
Mern	4.012078	2	Mem
perseverence	3.02396	2	perseverance
smail	3.624044	2	small
stiflly	4.310506	2	stiffly
stilly	2.905347	2	stiffly
stulfed	4.047332	2	stuffed
vnderstand	4.295835	2	understand
vou	3.577536	2	you
womun	4.203861	4	woman

Table 4.2.: Incorrect tokens in the Auxiliary Lexicon

Token	Index of Peculiarity	Instances
Bing	3.630874	2
Conservancy	3.727365	2
goer	3.88365	2
inta	3.151713	2
isoceles	3.594019	2
mme	4.20101	2
Olla	4.480145	2
thar	2.658921	3

Table 4.3.: Other tokens in the Auxiliary Lexicon

4.4. Discussion and future investigation

4.4.1. Differences between the test and training data.

By applying the process to the corrected versions of the texts, it is shown that the process is effective in the absence of OCR noise. A large proportion of the indicated errors are deduced to be wanted non-words and placed in the auxiliary lexicon.

An issue is apparent with the corrected text for the training data: figure 4.7 shows a number of wanted hyphens which have been used by the author to indicate a fractured speech pattern.

The de-hyphenation routine has recognised many of these intended hyphens as items to be corrected and figure 4.8 on page 52 shows the result (auto-corrections highlighted in green): a total of 336 tokens have been 'corrected'. It is difficult to see how this can be avoided; it is a rare case which occurs only within one work in the training corpus and not at all in the test corpus. It is possible that a more sophisticated algorithm could take account of the normal typesetting practices regarding where a word might be split, but it remains likely that such an algorithm might fail to make wanted corrections. For the OCR training corpus, most or all of these 'corrections' will also be made.

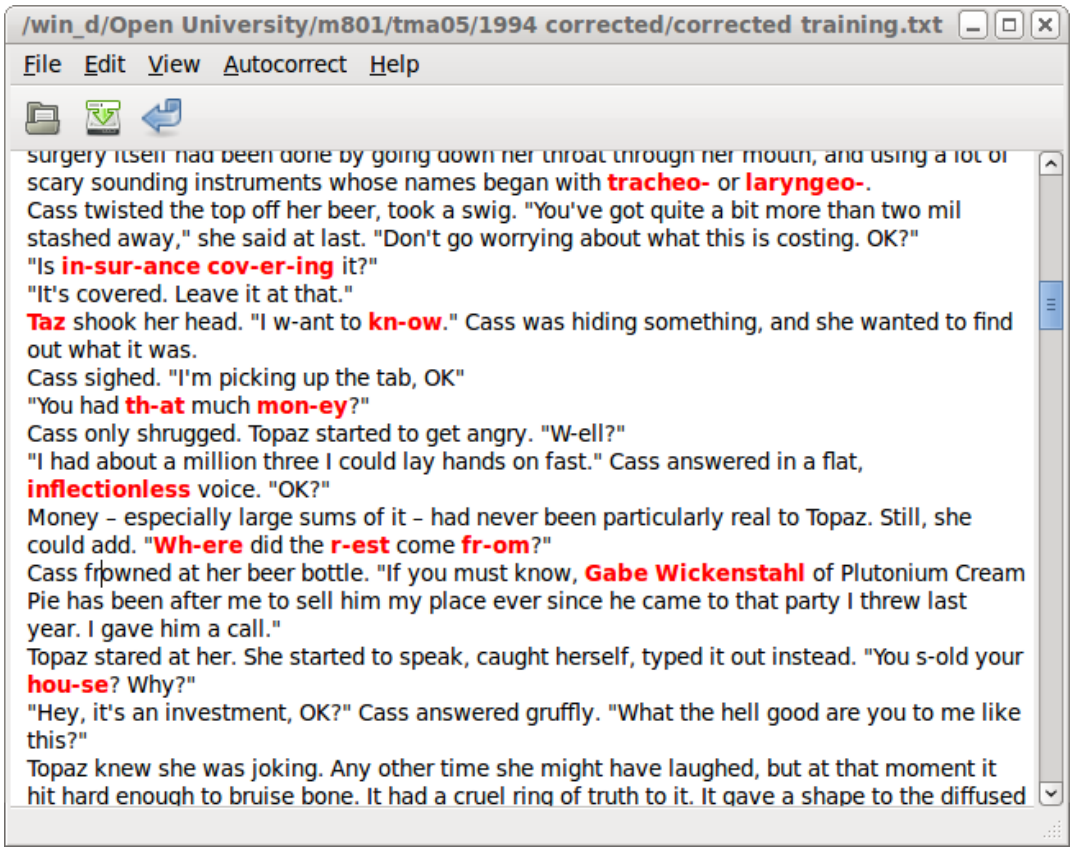

Figure 4.7.: Fractured speech - before hyphen correction

The texts are of similar size and contain material similar in context though different in detail; they each have approximately the same number of different non-words. After processing, both the numbers of words in the auxiliary lexicons and the numbers of wanted non-word tokens are similar.

There are almost twice as many remaining non-words in the test data as the training data after processing - 10,410 tokens against 5,116 - even though the auxiliary lexicons are similar in size. Zipf's law (recall the discussion in Section 2.3 and in particular Figure 2.1 on page 14) suggests that a small number of common tokens are likely to be responsible for the difference - and this is indeed the case; the most common sixty tokens account for almost five thousand instances. These sixty most common tokens are all character, species, or place names. Appendix C lists the most frequent non-word tokens; a comparison with the training text auxiliary lexicon in Appendix

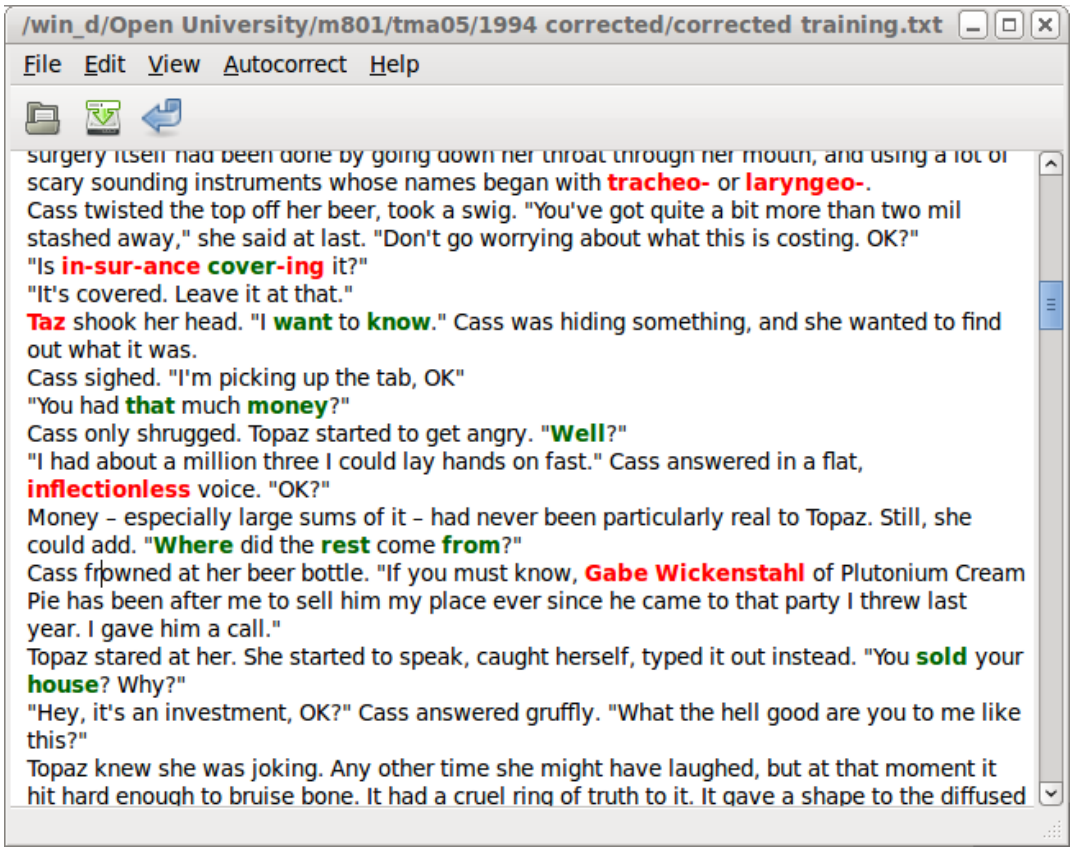

Figure 4.8.: Fractured speech - after hyphen correction

B will show that nouns are the most common entries in both and there are no obvious differences of style between them.

However, those sixty non-words have an average Index of Peculiarity of 5.8 (standard deviation 1.1) and as all are above the acceptance maximum of 4.5 are therefore excluded from the auxiliary lexicon. Since these tokens are excluded in the corrected test text, they are also excluded from the OCR text by the same reasoning. These tokens account for the majority of the difference in the numbers of non-word tokens between the test and the training OCR texts.

There are two other noise sources which significantly affect the quality of the converted output but which are not directly attributable to deficiencies in the OCR system itself.

Figure 4.9 demonstrates the resulting text when a page which has been printed

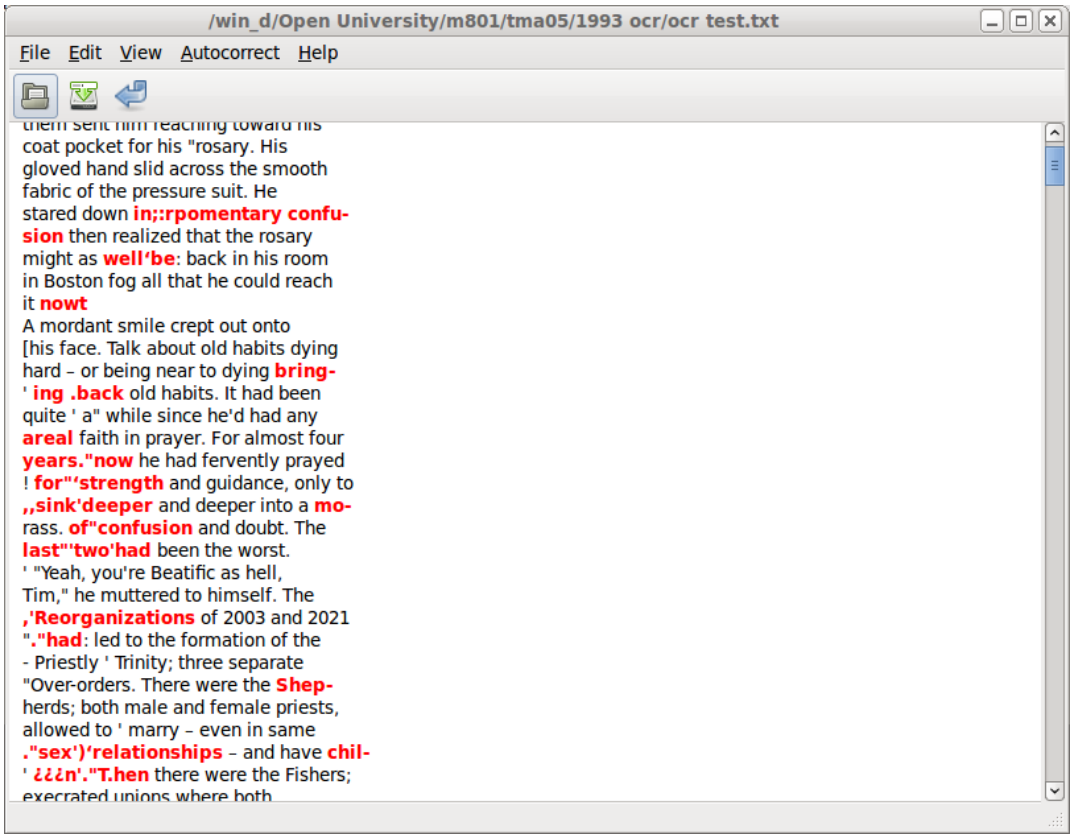

Figure 4.9.: Edge shading errors

too close to the binding is converted. The page is lifted slightly from the flat-bed scanner along that edge and as a result darkens; the result is usually a number of random marks which the OCR system interprets as additional punctuation, often joining tokens.

Figure 4.10 shows a different cause of errors: in this case the print on the page was particularly light and there has been insufficient contrast on the scan for the OCR system to cope. Errors caused in either mode - but particularly this latter - produce high volumes of errors which rarely repeat themselves. While the 'usual culprits' have a consistent mistranscription, these errors are random, of high IP, and frequent. Such a page may contain dozens of non-word tokens, against the one or two that a normal page would have.

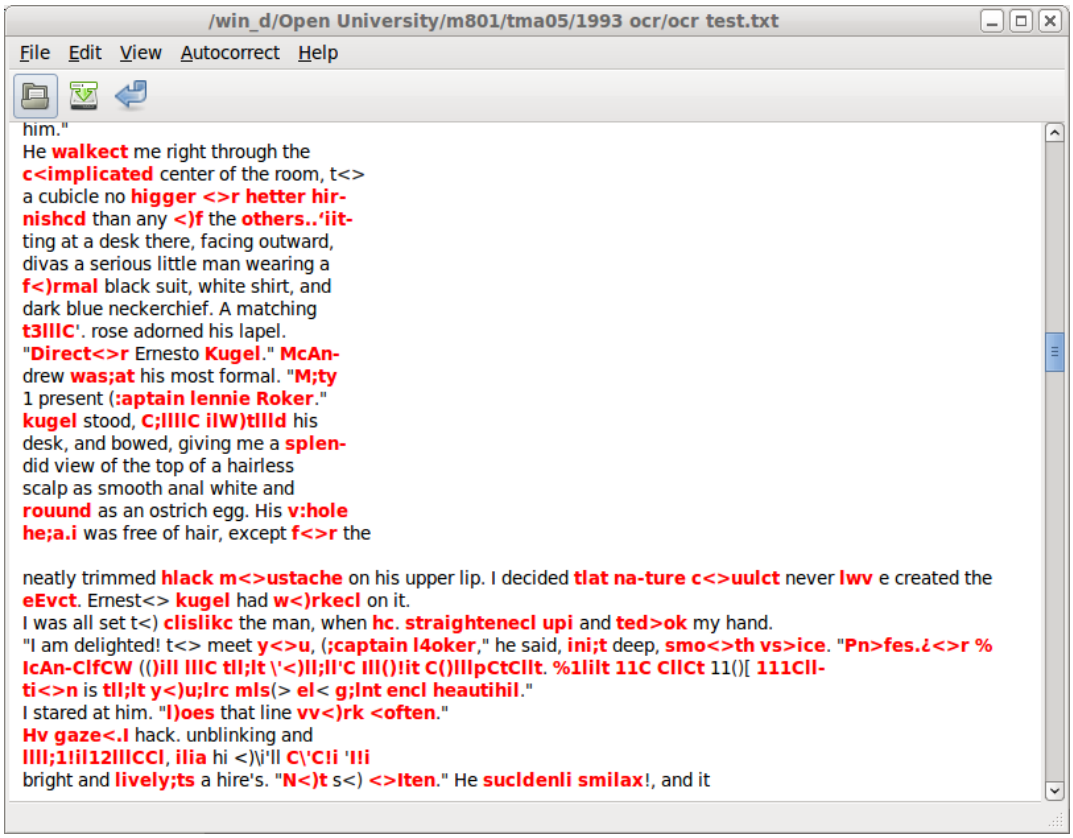

Figure 4.10.: Faint print errors

4.4.2. How might the results be improved?

There is no single IP value which separates words from non-words. Though a better IP algorithm might work it is not clear that this is the case, particularly as the words being identified are non-words by most definitions. Two approaches have been briefly explored but it should be noted that the results are at best tentative.

Raising the IP: The default acceptance level is selected on largely arbitrary grounds, from inspection of the training text. Raising it increases the numbers of accepted tokens, but also increases the number of unwanted non-words.

Figure 4.12 shows the effect on the training text. There is an increase in the number of acceptable tokens, but it is associated with an increase in the number of non-words - so while over nine hundred tokens could be removed from consideration this is at the cost of 416 instances of non-words silently included.

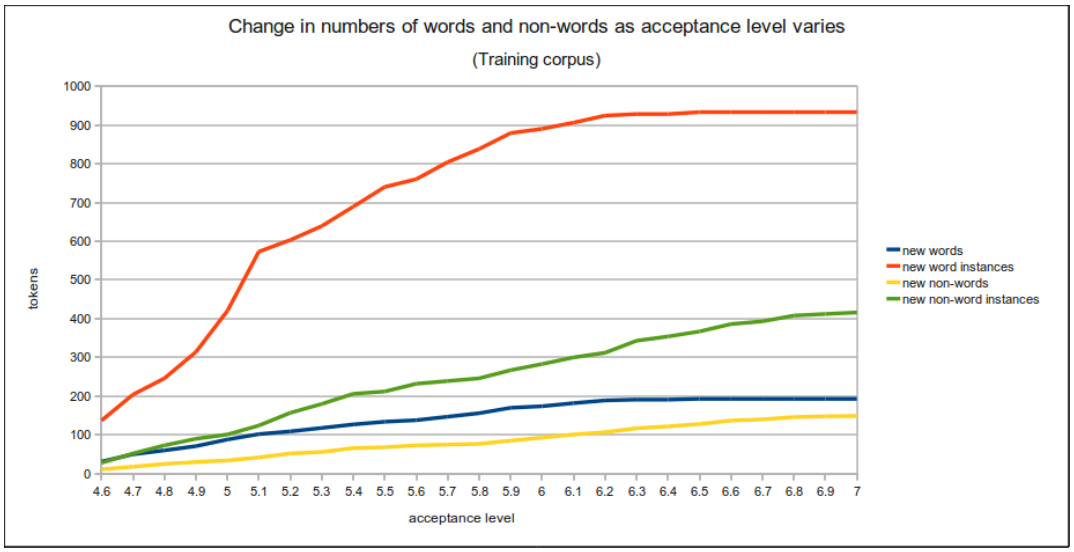

Figure 4.11.: Result of increasing acceptance level, training corpus

When this is applied to the training text, the results (figure 4.11) are dramatic.

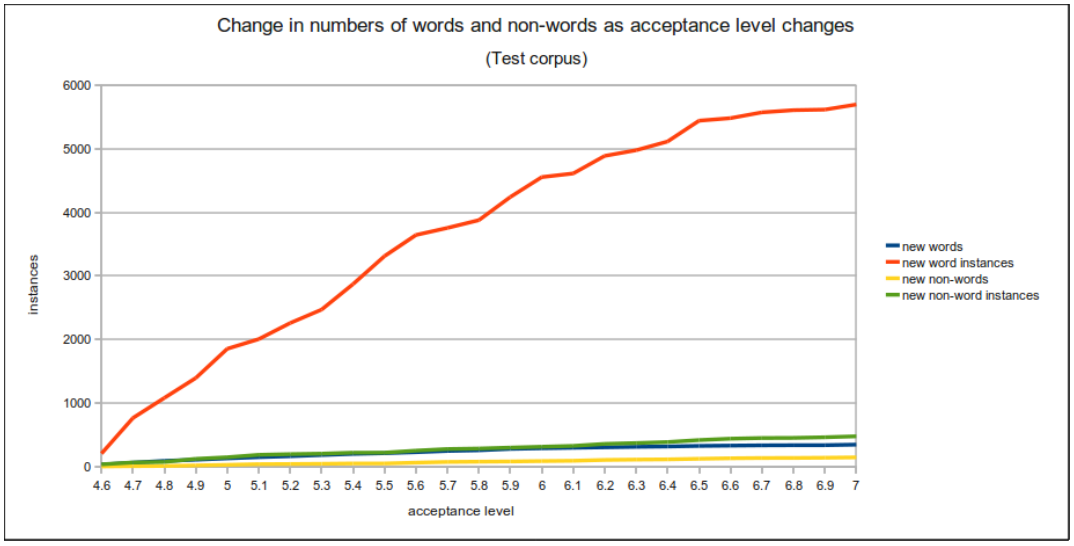

Figure 4.12.: Result of increasing acceptance level, test corpus

Better than five thousand tokens are available but the cost is 481 new non-word instances.

If the non-word instances are new 'usual culprits' then this could be an effective improvement; this is unknown at this stage but it seems unlikely. The original acceptance level of 4.5 minimised the number of non-words accepted; inclusion

of significant numbers of non-words is not helpful to a user.

A hybrid approach: The basic algorithm of the method assumes that desired non-words are merely low in IP and repeated, rather than using the frequency value. Zipf's power law relates frequency of occurrence to wanted non-words. Scaling with a power law suggests itself for high-IP wanted tokens; I divide the IP (calculated as previously) by log_{10} of the number of occurrences. This means that a high IP word with fewer than ten instances has an IP' which is raised, but lowered if it has more than ten instances.

For IPs of less than 4.5, the token is considered a word as before. If it has more occurrences than a trigger value and a calculated IP' of less than 4.5, it is also counted as a word. Table 4.4 shows how many high-IP tokens are actually words, how many are accepted, and how many new non-words occur when this is applied to the test OCR corpus. With a minimum of ten entries, we can identify every instance of these wanted words, with very few non-words included. At higher minima, the non-words are completely excluded, at a slightly reduced word acceptance rate.

Minimum instances required	10	15	20	25	30
Actual words (IP>4.5)	120	87	62	52	42
Actual word instances	5115	4746	4317	4088	3815
Accepted words	123	87	62	52	42
Accepted word instances	5146	4746	4317	4088	3815
Non-words	3	0	0	0	0
Non-word instances	31	0	0	0	0

Table 4.4.: Words, non-words, and accepted tokens

4.4.3. The 'usual culprits'

Entry of a token/word pair on the 'usual culprits' list needs careful thought. Common recognition errors on the OCR system affect many common words, particularly those

with ligatures, and commonly many misspelt variants of the same base word occur. However, there must be for any single entry an unambiguous base word or wrong-words will be inserted. See Section 3.6 for examples.

It is important also that document specific tokens are not added to the list since for a different document the replaced word might not be a replacement error - for example, character names should not be added. Only OCR transcription errors should be added.

It is likely that a practical system would have an ability both to add and remove 'usual culprits' on an ad-hoc basis. As more texts are processed, such additions are likely to reduce in frequency.

4.4.4. What is the next step?

Having reduced the number of putative errors, those remaining need correcting. Although not detailed in this research, observation reveals that the common commercial or open source spelling correctors are not particularly useful in the case of OCR errors. There are two reasons; first that many of the tokens in the text are considered non-words, and secondly that the correctors are based on a model of human input and the errors that human writers make - as first noted by Damerau (1964) and numerous workers since.

A human is unlikely to type a figure '*5*' instead of a letter '*s*', to replace the ligature '*ff*' with an '*E*' or to mistakenly use the wrong case in the middle of a word - but these are the types of errors that OCR systems make. In Damerau's model, 80% of typing errors are shown to be a single error - either a character added, deleted, two characters transposed, or an incorrect character substituted. Almost all errors from an OCR system, however, are the substitution of incorrect glyphs. Further, OCR error tokens are frequently more than one edit distance away from the correct word - that is, they may have more than one error in the token.

The noise channel model of the scanning and OCR system provides a mechanism to

improve the action of a spelling corrector. An analysis of the text provides a confusion set of incorrect glyphs and their corrections. With that statistical knowledge, and the observation that only substitution errors need be considered, the number of potential words generated by a non-word token is significantly reduced.

OCR generated token	Potential candidate words
hiU	friU
	fiiU
	biU
	eiU
	liU
	miU
	hfU
	hlU
	hrU
	hIU
	heU
	htU
	hhU
	hill
	hiu
	hifl
	hiLl

Table 4.5.: OCR system noise model word candidates

Table 4.5 on page 58 shows the effect of this model on the three-letter token '*hiU*', based on the confusion set generated from the test text and therefore including only observed errors. Only seventeen tokens need be tested at an edit distance of one, as opposed to the potential 52^3 (= 140608) possibilities that a simple transposition model would require. (As it happens, the correct word is present in this list and all other generated tokens are non-words. The editor used to write this dissertation offered thirty-eight candidates; none were correct.) This optimisation should make even quite high edit distance corrections manageable. A similar model - though for typing errors - is described in Brill and Moore (2000).

Further work should consider a complete proof-reading application, using the method described in chapter 3 to identify non-words and the statistical noise method to cor-

rect them.

4.4.5. Does the method work only for science fiction?

The method has been designed to solve a particular need to proof-read science fiction, specifically because of the prevalence of neologisms and 'strange' character/place names. As a sanity check, it has been informally tested with a number of texts, each with their own characteristics.

The King James Bible has a large number of Aramaic names, and is written in archaic English. *War and Peace* contains Russian names. *Moby Dick* is domain specific to whaling by sail-ship, and *Pickwick Papers* contains much phonetic dialogue. Finally, *Aviation in War and Peace* is a formal technical study.

It is observed that the method works in all cases, though to different levels. This is discussed further in the next chapter.

4.5. Conclusions

Domain specific words in a document can be a significant proportion of the actual tokens present and are unlikely to be found in a standard lexicon; as such, they are shown as errors when they are not, and a spelling corrector will not offer useful replacement candidates.

Such domain specific words tend to be proportionally common in the text, following Zipf's power law in terms of frequency distribution. Those most common will have many more instances than those less common.

They may be - in general - identified statistically based on the trigram statistics of a reference text with as few as two instances. Where a word occurs only once it cannot be so confirmed and must be classed as a potential error.

A method is proposed to provide a suitable statistical measure of a word's likely inclusion in the language of a training based on a few rules and a simple calculation; this is sufficient to identify the majority of such words, which are collated in a local

lexicon. Supplementary methods are briefly demonstrated which can deal even with highly unlikely words provided they are present in sufficient quantity.

A proof-reading application may use such a lexicon to reduce the number of indicated errors within a document, reducing the workload and increasing the accuracy of a human proof-reader.

OCR processing introduces a number of errors to a scanned text. The exact nature of those errors depends both on the noise model of the OCR system and on specific details of the printing or scanning process. Many of those errors will have only a single instance token but a significant number are repeatable and traceable to a single root word. Such repeated errors often have similar statistical properties to normal words in the language used. These can only be identified by inspection against a list of known token/word pairs, from which they may be automatically replaced with high accuracy.

A further OCR-induced error is the transcription of hyphenated words, particularly where narrow text-columns are used on the source material. These hyphenated words may be automatically corrected with reference to the created auxiliary lexicon and a default spelling lexicon.

It is shown that the method of identifying wanted non-word tokens is sufficiently robust to work in the presence of large numbers of OCR errors. A three-step process of lexicon building, replacement of known 'usual culprits', and automated removal of hyphens significantly reduces the number of indicated spelling mistakes.

Wallace (1987), Rose et al. (1994), and Jinkerson (1996) all agree that proofreading is difficult and that the help of technical aids can significantly improve accuracy. By reducing the number of instances where the proof-reader has to examine a token - particularly on 'difficult' domain-specific words - his workload is reduced. This method provides such assistance.

The development of a spelling corrector is outside the scope of this dissertation, but it is apparent that 'normal' word processors do not cope well with many domain-

specific errors - and cannot offer useful suggestions for 'made-up' words at all. The mechanisms by which such correctors work is often proprietary, but it is likely that a lexicon is used to offer candidate words. Obviously such a lexicon cannot offer words not in it. The distribution of words in a language is generally such that a very few words are very frequent (e.g. Zipf (1932)); implicit is that there are many words which are very rare, occurring sometimes less frequently than one case in a million. Church (2000) and De Roeck et al. (2004) have shown that the presence of such a word significantly increases the likelihood of it being present multiple times. It is therefore useful to be able to offer such words in a correction lexicon, both to indicate that the word is correct and to offer a useful replacement candidate. The process produces such a list; various ways in which it might be used are well-discussed in the literature.

5. Evaluation

This chapter evaluates a number of facets of the development of the method, the writing of the dissertation and the tools and methods used, and shows an example of the method used in a practical proof-reading tool.

5.1. The method

To demonstrate the utility of research, it is ideally tested against similar work in the field. There is however little in the literature which is directly relevant. While the work of Peterson (1980), Zamora et al. (1981), and Hull and Srihari (1982) is similar in that they use various statistical methods to identify tokens as words, I attempt to identify acceptable non-words.

The closest approach to this work I have seen is that of Ringlstetter et al. (2007) who conclude, after discussion of lexicon size, "a serious improvement of lexical correction techniques is only possible if dictionary construction is considered as an adaptive and *document-centric*[1] process". They discuss a way of creating a 'perfect dictionary' - see appendix D for details - which is essentially what I am trying to do. A comparison between their work and mine would indicate the relative efficiencies of using the external source or internal evidence to create the lexicon.

Unfortunately, they do not directly document the number of new entries to the lexicon. While they report improvements in the accuracy of the source material after automated correction, this is not a metric which can be applied to my work, since I

[1] Ringlstetter et al.'s emphasis

make no corrections. Their method seems likely to add tokens to the lexicon which are not present in the document. Finally, their selected domains - historical or scientific - are likely to be discussed on Internet-visible sources. Mine are not.

The research method therefore first demonstrates the process on a clean text, to discover how many nominally non-word tokens are present, and is then repeated on the same text as it is delivered by the OCR process. The auxiliary lexicons are then compared; first to discover the proportion of tokens present in the OCR text against the clean text, and secondly to discover any unwanted tokens in the OCR lexicon.

The method is repeatable by other workers, irrespective of the source material, and as such can be tested for alternate methods of isolating wanted non-words. It also clearly demonstrates changes in the auxiliary lexicons caused by the OCR process; since the aim of the research is to find a way to ignore 'correct' non-words this provides a simple and repeatable metric.

5.2. The results

The results, shown in section 4.1, clearly show that the process is effective in identifying 'correct' non-words in both ground-truth and OCR science fiction texts. To show that the process works with generic texts, I selected four well-known long pieces and one shorter technical article from 1920 from the Gutenberg Project. The number of non-word tokens before and after the creation of the auxiliary lexicon are compared and presented as a percentage reduction in figure 5.1.

The Gutenberg texts are tested only as ground-truth documents. A value of 30% indicates that 30% of the original indicated non-word instances have been identified as correct. All the Gutenberg texts show improvements; the King James Bible and The Pickwick Papers in particular contain many acceptable non-words.

Looking at these results I conclude that the process works; it is possible to identify significant proportions of acceptable non-word tokens in a corrected text. When the text is close to the training text, the results are significantly improved; as many as

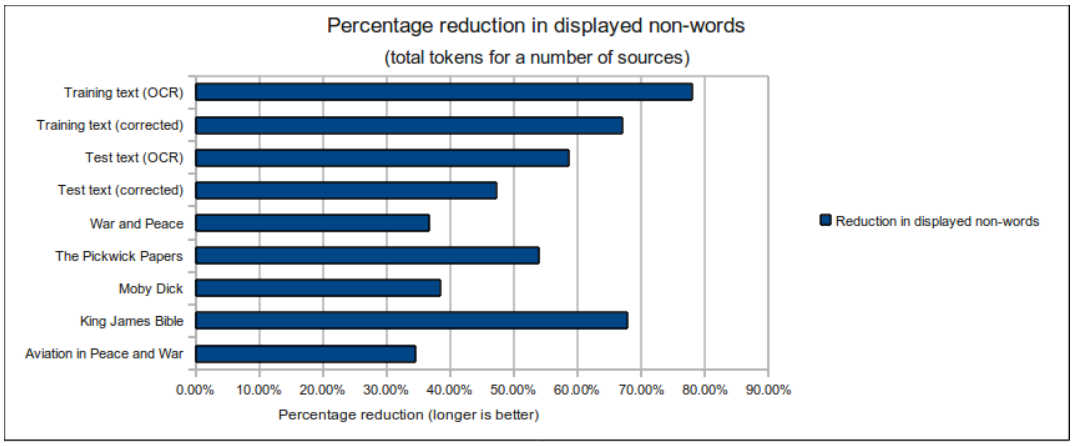

Figure 5.1.: Percentage reduction in non-word tokens; multiple texts

70% of the non-word tokens may be identified as correct.

I further conclude that the process works in the presence of significant OCR errors. Appendix E discusses a simple estimation of how many wanted non-words might be excluded from the auxiliary lexicon in an OCR text. As predicted, fewer words are included in the OCR lexicon, and in similar proportions for both test and training texts. Since these values are approximately double those estimated, it suggests an error in the estimation - but this cannot be confirmed without many more tests.

Together, automatically correcting inserted hyphenations (with reference to the auxiliary lexicon) and replacement of the 'usual culprit' tokens remove significant numbers of indicated spelling errors. The auxiliary lexicon identifies further tokens which can be removed.

It has been demonstrated that the majority of wanted non-word tokens can be extracted from a text in the presence of OCR errors. A very few unwanted tokens (fewer than 0.3%) remain within the token list; further work will be required to improve the discrimination of the process.

As discussed in Chapter 3, there were issues with both the use of a lexicon to differentiate word and non-word tokens and of de-hyphenation which needed careful consideration. While the methods I eventually chose may not be compatible in detail with other workers, I believe that the reasons I offer in the main text justify my

choices.

'Usual culprits' replacement and de-hyphenation do not directly contribute to the building of the auxiliary lexicon but do reduce the numbers of remaining errors. Since the two metrics are reported separately, and the aim of the research is to reduce the number of displayed errors when proof-reading OCR'd texts, I decided to include them in the main method; as those non-words are induced by the OCR process, it seems reasonable to remove them where possible.

5.3. Project management

I am a project manager by profession, with over thirty years' experience in major broadcasting projects. Initial uncertainty in the project aims is always a project management issue, and in this case resulted in some unnecessary investigative work: the generation of confusion tables and creation of a noisy channel model for the OCR system. That work will remain useful for future development of a spelling corrector.

Once more detailed objectives and aims were resolved, there were no significant issues with either timing or resourcing; the development of the necessary data (and the software to derive it) has proceeded within the tolerances built into the schedule.

Software project management has its own requirements, largely due to the often iterative nature of software development. By allocating time early in the project to develop a basic framework from which I could hang modules as required, I was able to develop individual short modules in isolation and in a short time as and when they were required; such agile development meant I had a working prototype at all times and if a new function was required - e.g. a method to output all non-words and their IPs - it was simple to add using existing functionality.

5.4. The educational process

Approximately one third of the previous post-grad diploma modules were purely technical and have not contributed to this dissertation other than by exposure to the academic process. The remaining modules are all directly relevant - Software Engineering, User Interface Design, and Systems Engineering.

Each of these deals not with the 'how' of the subject so much as the 'why'; lessons learnt from them ensure an efficient design process for the software and a robust testing regime. The development and test software user interface is simple, but similar enough in operation to a standard text processing application that it presents few challenges to a new user (informally tested with a small sample).

Systems Engineering teaches the importance of a global view for any project. In particular, it shows that stakeholders in any system are often more widely spread than might immediately be obvious and that satisfying them all might be impossible. It therefore becomes important both to identify them and prioritise them in the event that the work described here is incorporated into a distributed product.

The incremental nature of the TMAs required in this module was beneficial. I have found with previous modules having few TMAs that there can be issues ascertaining exactly what the university/tutor expects; here the TMAs build on previous ones and simultaneously allow confirmation of expected visual formatting.

5.5. Technical methods and tools

An early decision was the choice of the 'C' language to write the software, rather than one considered more suitable to text management, e.g. Perl. This was for two reasons; first because I was familiar with C and did not intend to learn a new language and second because I needed to handle UTF-8 text and non-letter characters, so I would need to develop text handling routines anyway.

The Ubuntu/Gnome desktop I use is based on the GTK+ toolkit. Using that base

provides a common 'look and feel' with other desktop applications, and its included widget set - in particular, a text-view widget - provides good functionality on which to base my application. Experiments with Mono/C# were ceased after the discovery of serious bugs in the GTK+ implementation.

A benefit of Linux systems is that excellent compilers and other development tools are freely available.

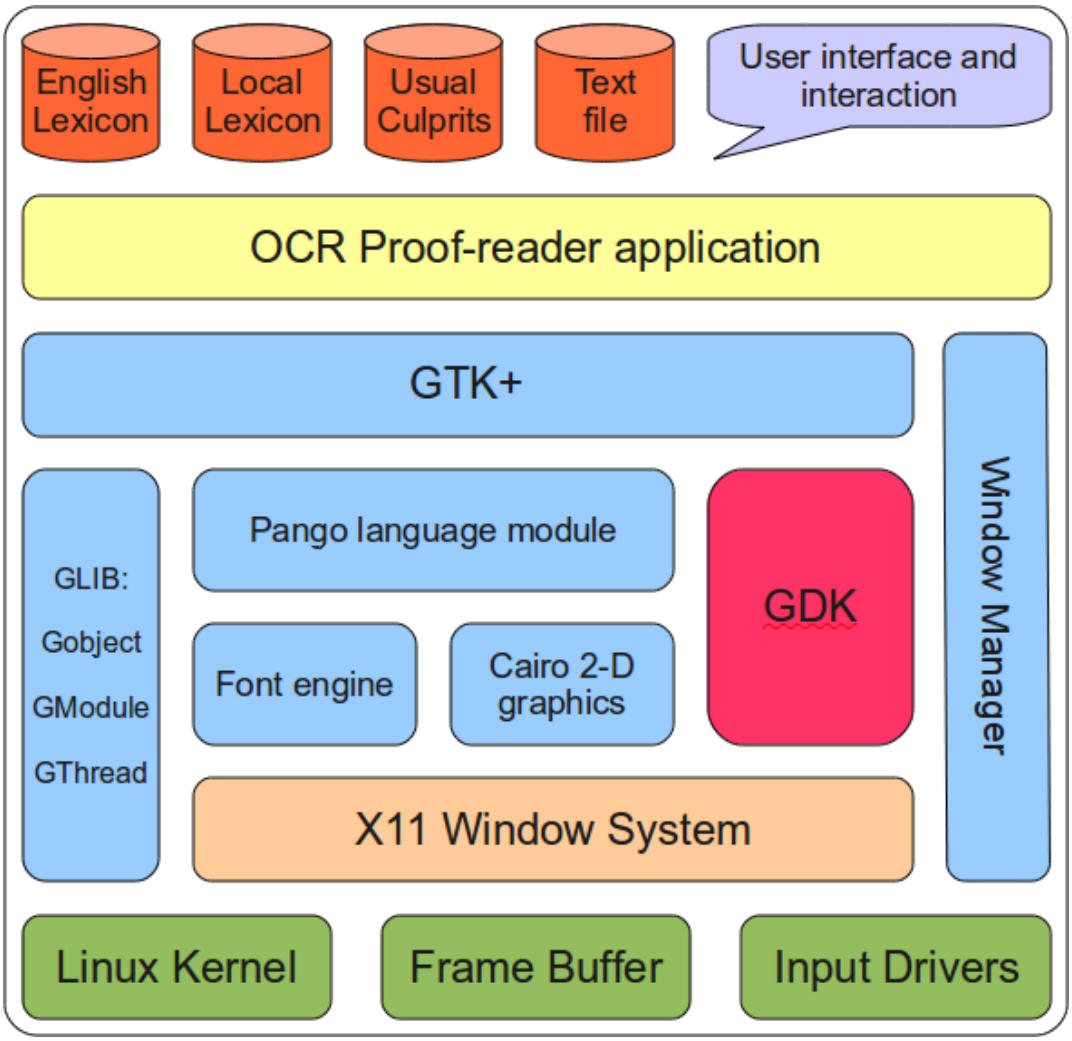

Figure 5.2.: The system and software framework

Figure 5.2 gives an overview of the system framework[2]; the application is at the top and the Linux system at the bottom. Individual components request services

[2]Inspired by a framework diagram at http://www.limofoundation.org/api/R1/aui/gtk/fnd/index.html

from components either lower or to one side of them. Detailed explanation is beyond the scope of this dissertation.

It is often the data structures in a program that define the operation of it, rather than the actual algorithms used. Here, the main issue is managing large lists of tokens; checking whether a token is present in a list must be done frequently and therefore as quickly as possible. Using a hash table provides an O(1) response to such searches and they are used in several places; to hold the main and local lexicons, for temporary storage when building the local lexicon, and to hold the 'usual culprit' token/word pairs. The GTK+ data storage is opaque to the programmer and uses documented accessor functions on the various classes used.

5.6. A practical example

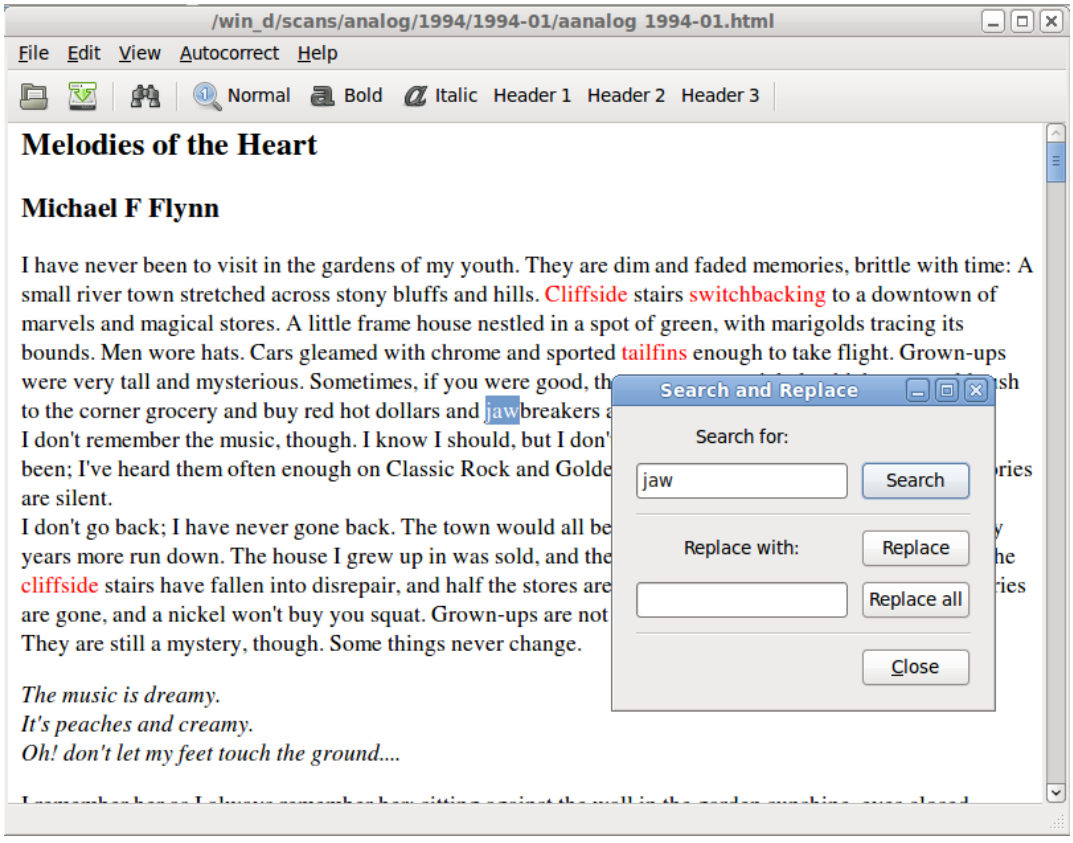

Figure 5.3.: OCR Proof-reader - search and replace

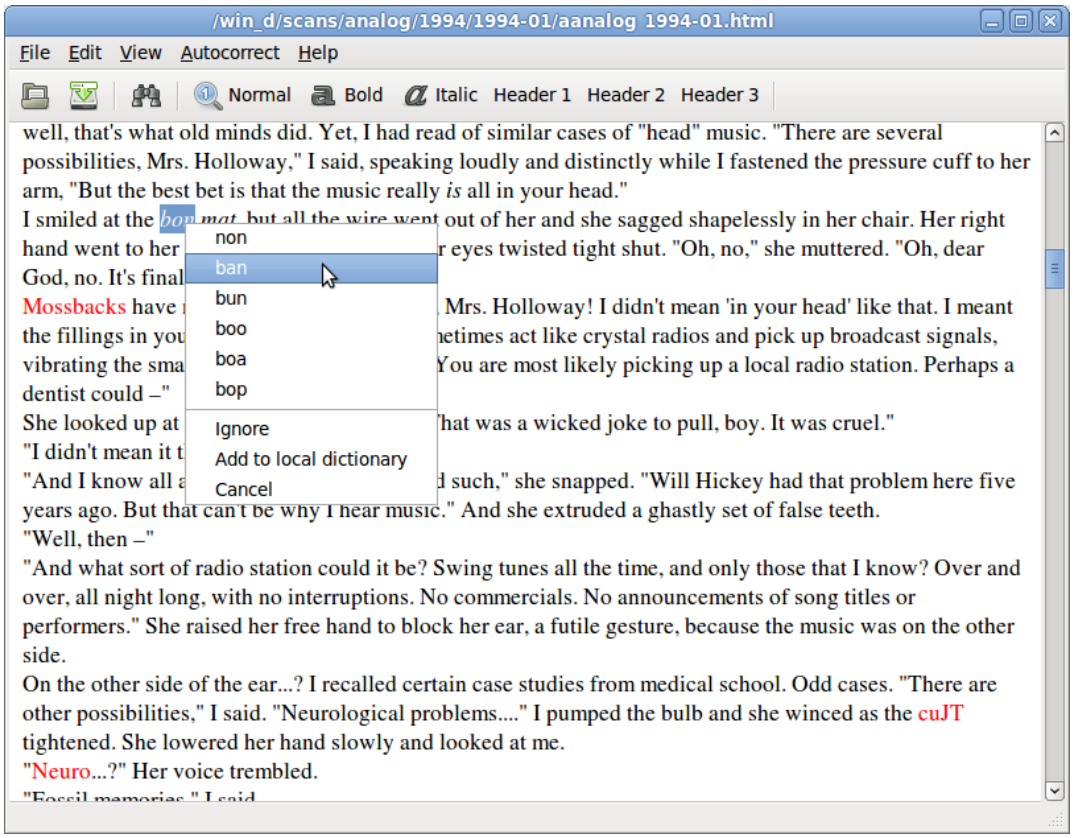

Figure 5.4.: OCR Proof-reader - spelling correction

Figures 5.3 and 5.4 show the process in a practical application. Although it uses the same code base as that of the dissertation, it uses the toolkit's facility to format text styles to present a more familiar view to a user (HTML is used as a simple input/output format compatible with most word processors). Other facilities such as search and replace have been added, and a spelling corrector using the methods proposed in section 4.4.4 is being investigated.

Development to this stage has taken perhaps twenty hours over a couple of weeks, simply building on the existing code base and structure.

5.7. In conclusion

At the end of this research, the initial objectives - to derive an algorithm to identify undamaged words in the OCR script and thus create an auxiliary lexicon, to evaluate

the effectiveness of that algorithm on the training text, and to evaluate it on text other than the training material - have been achieved on time and within the planned budget.

The method described here is effective both in identifying important domain specific words in text and in excluding words damaged by the OCR process, and has been used to create a working software application which demonstrates the utility of the method.

I have presented a method not previously discussed in the literature which has immediate application to manual proof-reading, but also to other fields where OCR text is used: one example might be for the automated creation of search terms for a large scanned database. While there is still work to do, particularly in better identification of very unusual but nonetheless wanted words, I believe I have made some small contribution to the field.

References

Angell, R., Freund, G., and Willett, P. (1983). Automatic spelling correction using a trigram similarity measure. *Information Processing & Management*, 19(4):255–261.

Banks, I. M. (1994). *Feersum Endjinn*. Orbit Books.

Bentley, J. (1985). Programming Pearls. *Communications of the ACM*, 28(5):452–458.

Bovill, J. (2009). The Spelling Society. *http://www.spellingsociety.org/*.

Brill, E. and Moore, R. (2000). An improved error model for noisy channel spelling correction. In *Proceedings of the 38th Annual Meeting on Association for Computational Linguistics*, pages 286–293, Morristown, NJ, USA. Association for Computational Linguistics.

Church, K. W. (2000). Empirical estimates of adaptation: the chance of two Noriegas is closer to $p/2$ than p^2. *Proceedings of the 18th conference on Computational Linguistics*, pages 180–186.

Damerau, F. J. (1964). A technique for computer detection and correction of spelling errors. *Communications of the ACM*, 7(3):171–176.

Damerau, F. J. and Mays, E. (1989). An examination of undetected typing errors. *Information Processing & Management*, 25(6):659–664.

De Roeck, A., Sarkar, A., and Garthwaite, P. H. (2004). Defeating the homogeneity

assumption. *Proceedings of The 7th International Conference on the Statistical Analysis of Textual Data (JADT 2004)*.

Esakov, J., Lopresti, D. P., and Sandberg, J. S. (1994a). Classification and Distribution of Optical Character Recognition Errors. In *Proceedings of the IS&T/SPIE International Symposium on Electronic Imaging*, pages 204–216. Spie.

Esakov, J., Lopresti, D. P., Sandberg, J. S., and Zhou, J. (1994b). Issues in Automatic OCR Error Classification. *Proceedings of the Third Annual Symposium on Document Analysis and Information Retrieval*, pages 401–412.

Fischer, S. R. (2005). *A History of Writing*. BCA/Reaktion Books, 2005 edition.

Forney, G. D. (1973). Viterbi algorithm. *Proceedings of the IEE*, 61(3):pp 268–278.

Franz, A. and Brants, T. (2006). All Our n-gram are Belong to You. *http://googleresearch.blogspot.com/2006/08/all-our-n-gram-are-belong-to-you.html*.

Golding, A. R. and Roth, D. (1999). A winnow-based approach to context-sensitive spelling correction. *Machine learning*, 34(1):107–130.

Golding, A. R. and Schabes, Y. (1996). Combining trigram-based and feature-based methods for context-sensitive spelling correction. In *Proceedings of the 34th annual meeting on Association for Computational Linguistics*, pages 71–78. Association for Computational Linguistics.

Ho, T. K., Hull, J. J., and Srihari, S. N. (1992). A word shape analysis approach to lexicon based word recognition. *Pattern Recognition Letters*, 13(11):821–826.

Howard-Hill, T. H. (2006). Early Modern Printers and the Standardization of English Spelling. *The Modern Language Review*, 101(1):16–29.

Hull, J. J. and Srihari, S. N. (1982). Experiments in text recognition with binary n-gram and viterbi algorithms. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, (5):520–530.

Islam, A. and Inkpen, D. (2009a). Managing the Google Web 1T 5-gram Data Set. In *Proceedings of the IEEE International Conference on Natural Language Processing and Knowledge Engineering (IEEE NLP-KE'09)*, Dalian.

Islam, A. and Inkpen, D. (2009b). Real-Word Spelling Correction using Google Web 1T n-gram with Backoff. In *Proceedings of the 18th ACM Conference on Information and Knowledge Management (CIKM 2009)*, pages 1689–1692, Hong Kong.

Jacquemont, S., Jacquenet, F., and Sebban, M. (2007). Correct your text with Google. In *Proceedings of the IEEE/WIC/ACM International Conference on Web Intelligence*, pages 170–176. IEEE Computer Society.

Jinkerson, L. A. (1996). Computer spelling checkers: an example cognitive tool. In *Proceedings of the 1996 international conference on Learning sciences*, page 432. International Society of the Learning Sciences.

Kanungo, T. and Haralick, R. M. (1999). An Automatic Closed-Loop Methodology for Generating Character Groundtruth for Scanned Documents. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 21(2):179–183.

Kernighan, M. D., Church, K. W., and Gale, W. A. (1990). A spelling correction program based on a noisy channel model. In *Proceedings of COLING 1990*, pages 205–210, Morristown, NJ, USA. Association for Computational Linguistics.

Kilgarriff, A. (1998). BNC database and word frequency lists. *http://www.kilgarriff.co.uk/bnc-readme.html*.

Klink, S. and Jager, T. (1999). MergeLayouts - Overcoming Faulty Segmentations by

a Comprehensive Voting of Commercial OCR Devices. In *Proceedings of the Fifth Internation Conference on Document Analysis and Recognition*, page 386.

Kluzner, V., Tzadok, A., Shimony, Y., Walach, E., and Antonacopoulos, A. (2009). Word-Based Adaptive OCR for Historical Books. In *2009 10th International Conference on Document Analysis and Recognition*, pages 501–505. IEEE.

Kukich, K. (1992). Techniques for automatically correcting words in text. *ACM Computing Surveys (CSUR)*, 24(4):439.

Levenshtein, V. I. (1966). Binary codes capable of correcting deletions, insertions, and reversals. *Soviet Physics Doklady*, 10(8):707–710.

Li, W. (1992). Random texts exhibit Zipf's-law-like word frequency distribution. *IEEE Transactions on Information Theory*, 38(6):1842–1845.

Liang, H. L. (2008). *Spell checkers and correctors: A unified treatment*. PhD thesis, University of Pretoria.

Lopresti, D. P., Tomkins, A., and Zhou, J. (1995). Systematic Bias in OCR Experiments. In *Document Recognition II (IS&T/SPIE Electronic Imaging)*. Citeseer.

Lu, S., Li, L., and Tan, C. (2008). Document image retrieval through word shape coding. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 30(11):1913–1918.

Michel, J.-B., Shen, Y. K., Aiden, A. P., Veres, A., Gray, M. K., Pickett, J. P., Hoiberg, D., Clancy, D., Norvig, P., Orwant, J., Pinker, S., Nowak, M. a., and Aiden, E. L. (2011). Quantitative analysis of culture using millions of digitized books. *Science (New York, N.Y.)*, 331(6014):176–82.

Mitton, R. (1996). *English spelling and the computer*. Longman.

Nagy, G., Seth, S., and Viswanathan, M. (1992). A prototype document image analysis system for technical journals. *Computer*, 25(7):10–22.

Nylander, S. (1999). Statistics and phonotactical rules in finding OCR errors. In *Proceedings of NODALIDA'99 (Nordiska Datorlingvistikdagarna)*.

Oxford Dictionaries (2010). How many words are there in the English language? *http://www.oxforddictionaries.com/page/howmanywords*.

Peterson, J. L. (1980). Computer programs for detecting and correcting spelling errors. *Communications of the ACM*, 23(12):676–687.

Peterson, J. L. (1986). A note on undetected typing errors. *Communications of the ACM*, 29(7):633–637.

Reinke, E. C. (1962). Classical Cryptography. *The Classical Journal*, 53(8):113–121.

Reynaert, M. (2008). Non-interactive OCR post-correction for giga-scale digitization projects. *Computational Linguistics and Intelligent Text Processing*, pages 617–630.

Ringlstetter, C., Schulz, K., and Mihov, S. (2007). Adaptive text correction with Web-crawled domain-dependent dictionaries. *ACM Transactions on Speech and Language Processing (TSLP)*, 4(4):9.1—-9.36.

Rose, T. G., Evett, L. J., and Lee, M. J. (1994). Contextual Analysis for Text Recognition: A Comparison with Human Performance. *AISB Quarterly*, 88:41–44.

Schurmann, J., Bartneck, N., Bayer, T., Franke, J., Mandler, E., and Oberlander, M. (1992). Document analysis-from pixels to contents. *Proceedings of the IEEE*, 80(7):1101–1119.

Shannon, C. (1948). A Mathematical Theory of Communication. *The Bell System Technical Journal*, 27:623–656.

Shannon, C. (1949). Communication In The Presence Of Noise. *Proceedings of the IRE*, 37(1):10–21.

Strouthopoulos, C., Papamarkos, N., and Chamzas, C. (1997). Identification of text-only areas in mixed-type documents. *Engineering Applications of Artificial Intelligence*, 10(4):387–402.

Taghva, K. and Stofsky, E. (2001). OCRSpell: an interactive spelling correction system for OCR errors in text. *International Journal on Document Analysis and*, 3(3):125–137.

Takahashi, H., Itoh, N., Amano, T., and Yamashita, A. (1990). A spelling correction method and its application to an OCR system. *Pattern Recognition*, 23(3-4):363–377.

van Delden, S., Bracewell, D., and Gomez, F. (2004). Supervised and unsupervised automatic spelling correction algorithms. In *IRI 2004. Proceedings of the 2004 IEEE International Conference on (0-7803-8819-4)*, Information Reuse and Integration, 2004., pages 530–535.

Wallace, B. (1987). Hypnotic Susceptibility and Proofreading Accuracy. *American Journal of Psychology*, 100(2):289–294.

Yannakoudakis, E. J. and Fawthrop, D. (1983). An intelligent spelling error corrector. *Information Processing & Management*, 19(2):101–108.

Zamora, E. M., Pollock, J. J., and Zamora, A. (1981). The use of trigram analysis for spelling error detection. *Information Processing & Management*, 17(6):305–316.

Zipf, G. (1932). *Selective Studies and the Principle of Relative Frequency in Language*. Harvard University Press, Cambridge, Mass.

Index

Glossary

Auxiliary lexicon: Throughout this text, the auxiliary lexicon refers to a lexicon used solely to gather and contain non-words which have been identified as correct.

British National Corpus (BNC): A large list - approximately one hundred million words - garnered from published and spoken texts. Kilgarriff has arranged the BNC in order of word frequency and established the relative frequency of the words by various criteria.

Corpus: A body of written or spoken work. In this study, a number of science fiction magazine articles. See also British National Corpus.

Default lexicon: The main lexicon used as a first check as to whether a word is correct; if it is not in the default lexicon then it is a non-word and requires further consideration. The lexicon is a standard linux file used for spelling checkers.

Dictionary: Similar to a lexicon, it is a list of words used as a reference. However it will usually contain some sort of metadata; this might be an explanation of the meaning of the word, a definition of its part of speech, or similar.

Distributed Proofreaders: Distributed Proofreaders is a project to assist Project Gutenberg; large numbers of volunteer proof-readers correct one page at a time - enabling fast correction of complete books.

dpi: Dots per inch - a measure of the resolution with which an image is scanned (though not necessarily the resolution actually available in the original printed image).

Edit distance: A measure of the number of changes (removal or insertion, replacement, or transposition of characters) required to convert a non-word into a correct word.

Error correction: The process of correcting an error which has previously been detected.

Error detection: The process of determining whether an error exists. For a spell-checker, this is most commonly done by comparing to a lexicon.

False friends: See wrong-word.

Glyph: A unit of written or printed text - either a single character or a group of characters. An entire word can be considered a glyph, but more commonly used to discuss a small number of letters in a group.

Ground-truth: The condition of a document from which all the errors have been removed; it is assumed to represent the text as originally intended by the author.

Index of Peculiarity (IP): A statistical value calculated for a word which indicates the likelihood that the word is a member of the words contained in the training text.

Lexicon: A lexicon contains only a list of words, with no metadata such as a dictionary would have. In the experimental software, lexicons are used both to define the default spelling for the majority of words, and to hold the non-words selected as being correct.

Ligature: A pair of letters which are typographically associated with each other and are represented as a single joined glyph - e.g. 'ff', 'fl', 'fi', or 'oe'.

Longest Common Substring (LCS) algorithm: Used to identify the number of symbols shared by two strings, in order. This can provide a measure of similarity

between two strings; it can also be used to identify missing or changed segments between two strings.

n-grams: groups of symbols of length 'n'. When 'n' is two, the groups are referred to as bi- or digrams; when three, trigrams. In a spelling correction context, the symbols almost always refer to characters.

Neologism: 1. a new word, meaning, usage, or phrase.
2. the introduction or use of new words or new senses of existing words.

Noise model: In a noisy channel model, the noise model defines the characteristics of the noise itself.

Noisy channel model: A way of describing a transmission channel for data which includes a noise source. The noise affects the data passing through the channel in a random but statistically reliable way to introduce errors.

Non-word: a token which is in neither the default nor an auxiliary lexicon. A token may be a non-word at the start of analysis and be promoted to a word after analysis is complete.

Non-words: Words which are incorrect in context; the context usually being defined by a lexicon or dictionary.

Optical Character Recognition (OCR): The process by which individual letters, words, and entire texts are extracted from a scanned image of a document.

Project Gutenberg: Project Gutenberg was founded by Michael Hart in 1971 as a free distribution system for public domain texts in electronic form.

Proper noun: the name of a specific person, place, or object; always indicated with a capital letter when written.

TMA: Tutor Marked Assessment - used by the Open University for continuing assessment of a student's progress through a course.

Token: a sequential group of characters not containing any spaces but which may contain any other characters including non-letter characters and punctuation.

Training document: A document (or set of documents) used to train the error correction system. In this software, two documents are used: one from which all the errors have been removed, which provides the statistical data used to identify words as likely to be correct, and a second which is corrected using that data to test the validity of the process.

Word: a token which is a correctly-spelt word in context. It may not generally be considered correct in normal usage, but its provenance as a word has been established either by its presence in the default lexicon, or by analysis as described in this dissertation (and therefore its presence in an auxiliary lexicon).

Wrong-word: A word which is nominally correct, by being present in a lexicon or some other definition, but which is wrong in context; for example, 'bat' instead of 'hat'. These words cannot be easily disambiguated. Also known as 'false friends'.

A. Extended Abstract

The Text Contains its Own Lexicon: Extracting a Spelling Reference in the Presence of OCR Errors.

David Neil Barnes
M3085208

Extended Abstract of Open University MSc Dissertation
Submitted 21/02/2011

Introduction

Increasingly, optical character recognition (OCR) systems are used to convert scanned material. Such texts will always require proof-reading - a difficult task. An OCR text contains three kinds of words: those correctly spelled, those with errors from the OCR process, and those which are correct in context but are nominally non-words (i.e. those not to be found in a reference lexicon). The result is that the displayed text is considered by most commercial word processors to have many more errors than are actually present - complicating the task of the proof-reader since he must consider all the marked errors.

Of the errors induced by the OCR process, a large number may be automatically corrected with no need for user intervention. If the remaining non-words can be classified as wanted or non-wanted - that is, identifying those which are correct in context - then two benefits accrue: firstly, they may be removed from those indicated errors, and secondly, they provide an extended lexicon which might provide more suitable words for spelling correction.

The majority of the wanted non-words will be specific to the subject domain of the OCR'd text. Recent research indicates that such words are uncommon in general usage, but where they occur once within a text, they are likely to occur frequently,

and usually in close proximity. Even where the words are invented - for example, places or names in science fiction - this relationship exists. If such a word can be differentiated from actual non-words, those containing errors caused by the OCR system, a robust method may be designed to classify it as correct-in-context and therefore wanted.

In this dissertation, I present a solution which combines a number of traditional methods - both lexical and statistical - to isolate the majority of these words from a text containing many OCR-induced errors. With such correct words removed from consideration, the task of the proof-reader is simplified.

Method

Three algorithms are applied to texts: creation of a lexicon containing wanted non-words, removal of common consistent errors, and correction of hyphenated words. The latter two do not contribute to the auxiliary lexicon but reduce the number of displayed errors; de-hyphenation requires both the default and the auxiliary lexicon for best effect.

Wanted non-words are identified by their frequency and by their similarity to a statistical model of the language used in the training material.

Hyphens within words are removed if and only if their removal results in a valid word - that is, a word which is present in a default lexicon or the auxiliary lexicon.

Common and consistent errors ('usual culprits') caused by the OCR process and the correct matching word are held in an external file (to allow future changes) and are replaced when found.

The method is applied to a number of corpuses:

- the corrected version of the training material, to give a 'best effort' baseline
- the training material itself, to provide a comparison
- other texts, to demonstrate the general utility of the method

The method is judged by the proportion of wanted non-words located in the OCR text and the number of non-words placed in the auxiliary lexicon.

Results

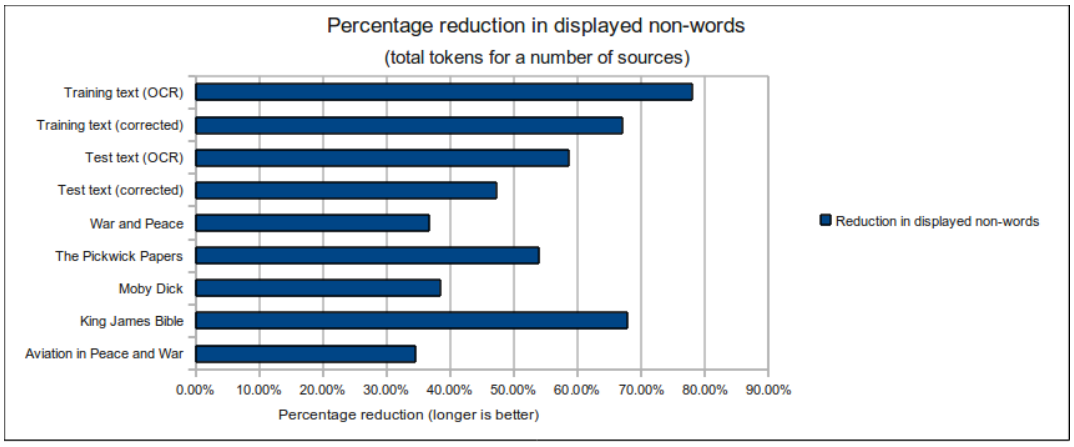

Figure A.1.: Reduction in indicated non-words on several texts

Figure A.1 shows the response of a number of texts to the process. In all cases the number of displayed errors are significantly reduced; by more than 70% in some cases.

The number of non-word tokens in a text is between 1.9% and 4.6% with an average of 3% (the higher values, as expected, on uncorrected OCR texts). The size of the auxiliary lexicon is surprisingly small - usually between 0.15% and 0.2% of the number of tokens in the original text.

From the OCR versions of the training and test documents, the auxiliary lexicon contained 4 and 39 non-words respectively. The OCR version of the auxiliary lexicons are approximately 10% smaller than the corrected versions.

Analysis

In an OCR'd text, there are four cases of indicated errors - hyphenated words, 'usual culprits', wanted non-words, and actual non-words, of which all but the wanted non-

words are introduced by the OCR process. The first two are corrected automatically and almost completely. Hyphenated forms of wanted non-words that occur only once in an un-hyphenated form will not be corrected, and 'usual culprits' which have not been seen in development or use and therefore not added to the list remain unreplaced.

Wanted non-words are located and collated provided that they are sufficiently common in the text; the most important words are always the most frequent. Tokens which appear only twice in the corpus are subject to a high probability of damage and fail to be accepted at a rate consistent with that of the declared error rate of the OCR system.

A number of other wanted words are not collected; either they occur only once in the corpus and so cannot be assumed to be wanted, or they are wanted but have a high IP - as might happen when an author uses foreign words or names (or in the case of science fiction, invented languages deliberately different from English orthography).

That there are stylistic differences between the training and the test corpus is apparent when it is observed that the test text has large numbers (over five thousand) of remaining wanted non-words after processing. Inspection reveals that these are caused by only 120 discrete tokens - all nouns or proper nouns - which occur within two stories in the corpus. These names have been selected to be highly 'foreign' by the authors and as such produce a high IP, which prohibits their inclusion as wanted words.

Discussion

Identifying a word as correct when it is in a lexicon is trivial – but many important words may not be in the lexicon. Domain-specific words – proper nouns in particular but also neologisms or spoken partial words – are correct in context but are identified as errors, and these nominally correct words may represent over half of all indicated errors.

Previous work shows that important words in a document are frequent, and that

it is possible to identify words which are statistically likely to be in the language and vocabulary of the document. Combining these two methods allows a significant number of words to be identified as correct in context - significantly reducing the effort required to proof-read the text. At the same time, very few wrong-words are so marked; ideally none would be.

It is clear that the method works; the number of displayed errors may be reduced by almost eighty percent, and very few words with OCR errors are so marked. However, there is room for improvement. A word may be important and common, but chosen by the author to have a spelling which is statistically very unlikely in the language of the text; it will be marked as an error. While any current spell-checking program would normally include a user function to ignore all instances of a particular word, ideally the word would be identified correctly before the user sees it.

Exploratory work suggests that a promising approach is to use the frequency of a word more aggressively in the word/non-word decision. Dividing the IP of a token by the logarithm of its occurrence count has the effect of increasing the IP for tokens with fewer than ten occurrences, and decreasing it for those with more. Thus an unusual token with many occurrences can be selected as correct, while one with only a few occurrences is rejected. This approach, if used on high-IP words previously rejected, appears capable of identifying most if not all instances with a very good rejection rate, and will be the subject of further research.

As it stands, there will almost always be words which are not in the lexicon and which this method cannot catch. They may be used only once; they may be foreign words or names; they may be dialect or words deliberately broken by the author. Such words will always appear - properly - as possible errors to which consideration must be given.

B. Auxiliary Lexicon Contents for the Corrected Training Document

This list - the complete contents of the auxiliary lexicon after processing - is presented as an example of the types of words which are present in the text and which would require attention from the proof-reader with a normal spelling checker.

It is sorted first by the number of occurrences and then alphabetically, so the most common tokens appear first. All tokens are shown with any associated non-alpha characters. All words before 'antihydrogen' - the first 220 - have ten or more occurrences and account for 6798 instances; 66% of the total occurrences.

All these words are correct in context but none are present in the default lexicon; most are considered as non-words by common spelling checkers. They do not include all the tokens indicated as non-words in the corrected document since tokens which have too high an index of peculiarity, which occur only once, or have too few characters are excluded by the process.

Drin Vick Jem Shimizu Arnie Santee Rabinowitz Toc Sneel Kelric Shh Reb Crisium Ordner Starmind Pellar elvirti Bantry Consuela Bubba Kiah camsofetu Taz Tessier Sarkar Marianna Kleth Magnus Reg Ket Trajendi Kup Schwenck Levexitor Stardancers thrusters Trimus Stardancer Cantner Messer Enceas Ravagger Taemon Provincetown Donelson remoras Hoy Magick Marmet Larvey Pinkertons Boegemann Aurum Do'utian Sanson Symbiote Laney Buchi graviton Hronk Kody Ferron Octahedron Stivra airlock eyewall Jenithar Jiang painlusters Skorin Colly's Ling shimp Soames Crocker Gori'allolub Gunn isosuit Arnie's Grombaugh Maynerd Shara shimps

Vrank Werner Cuthbert Francesco freefall Levexitor's Mariah Monna Sato Ehrmann goin Heatherington Marya Rucellai Shorty Trippy Wegner wristphone Chalnas Inman Kic Los perceptualization Tigernach Dammit Gunter Loagmeyer Stooka Angeles Cradar Devereaux Mamikonian Tch tech Vick's Zach Helmstreit Hobson Isobel Shimizu's starship Tak Tunman Fennell Horst Lith Lunarians nanotechnology remora suh Wivvel Drummond Sasaki Schuldman Skyler virophage wergild Whitechapel Danita Gadagal Granddame Innis Morgenstern painluster skyfall skystalk someone's Thorny's comm Dellor downdrafts L'uboleng mindlink Mistuh Pribhala Starnes Thorson Torrie Anwalt Clarkeville Davey longterm Lucena patient's Quintana rainwall veerspace afterwards Bodvian Cascadian dammit Diesha Dum Eston holosphere Jessa Salieri Schramm Shil signorina stasis talkboard Bantry's Besselton bot bro cronkites Droner DumDoom gauge Jenitharp Kor Lattyak Lunarian Marianna's milady Qataris Rild Santee's Searles Trancers uips Cantner's Chessie Fennell's Fredericksen holo Homo hon Kinergy knowbot Maarten Monura org Ottorino Remaldorixi Rockledge shaper Skyfall spacers stardrive antihydrogen Bateson Blount Coombs deskphone Entwhistle Giocondo gravitons Hardaway Hollilockes Lokelani magickian max Moonbase Oelievay oilbush Peeler peripump Rev'ren rogallo Sha spacer Stardance and/or crump DeadRinger dewars Dianara downdraft Elisabetta erst Glint's Guenther Hallowell honour hyperdrive Kumagai Magickian Maureena millibars resettlement Seavey Soundex starships telempath Theric Trimusian Armstead audiocom bein Belo Birla comsat comset Crisp's Do'utia Dzong earmouth gobuggy Horizonte ing Joelina Karlsson Kechar knowbots lightspeed Luellen Maastrichtian Marya's Noor oldster Paddington Pallas Rashid realtime Reverend's Roberts's Senor Signor sixpack Snelling Stookas Superdee swiftkillers telepath viewport Vinci volatiles Westford wordless Ambrose antinomy Auerbach beambox Bemis biomedical bots Brainerd chamberpot Choupette Cigno closeup Combuster commsats comparator Conservancy Consuela's cyberlink cybersystem del doci Dom faceplate fightin Gandaki Grissom haole helluva Highest's Holborn inhibitor Jasso Kiefer Kody's L'uboleng's Laneel Lucullus Mariah's Meleagros

*mockup nanofabs nothin pheromonal Philo Rosso Routledge Sarkar's Shaper Skyler's spaceborn Spacers spacetime Starhunters Tournet Tyrson unlit Waterworld aeroship Affonso Amin Barbry Braun caelestis coArtistic commsat Crump cyber dewar Dimitrios dirtside Do'utians doeskin doin Elvirti Emme Eustace facto Fell's flatscreen Gadagal's Garraty gettin goddam Gori'allolub's Greenberry Hallaj Hegerman introns Kelric's keypad Lancashire Larvey's Lector Loudon Lubchek's Mackey Magickal manmade Marian's missileers Mycroft Nana nonlinear oiler Oltion Ordner's paincult Painesville Penicillium primitivists rapturists ratiodrive Roper roquefortii Sanctorum seatbelt Senhor sentients Seticorp singlehandedly sleepsack Smythe spacefaring Sulke Tenshin Thecla til Trajendi's Trancedance Trumpton tryin Tyrson's underway unstrapped vips warbabies windowless antiStardancer anybody's Arosa backtime Bandello Bigney bio bloodwite Brava cafe cain't camsofetu's Caroline's Centauri comin comp connie could've coverup crewmember Crowe cybber darkies darkskinned des dialogue dopamine draught Driftglass Drummond's Earthside electively equerry Evinrude eyeward Feng fluidic focusers Froggies gauges gen'l geosynchronous git Gunter's handholds Hans's Haskel holorec inertialess inhouse intelligences intro Ittasta Jenkin's Jivin Kish laminar Lashi lookin magick Marmet's Mauna McClernand monofilament Nels nightvu onstage outboard Painlusters Paperclip Pavonis payback Presidente Pribhala's Prunebottom rad Readys Remaldor Remaldorix reticle Rosenfield sapiens Schuldman's Schwenck's seawater Shapers Shorty's Slidell somebody's somethin Starmind's stealthed supraluminal synaptic teddy Tessier's tetrahedral Tevverina thaumaturgical tranged trashrack trillionaires trove trusswork unimaginably uns upwellings uraninite versa voicebox vumail Walther Westlake Wha Wilshire woodlot workspace Zaubern *Atlas *That *What airflow Almroth anoxia antigravitons apportation Arrivederci behaviors Belter beltship blocker Boegemann's bonelessly brainwave Brazilian's Bubs Buenos Cafe carbonaceous cartwright Casebook catatonia Cenacolo Cerean ceresquake cetera Chalnas's Chatur childrens Chong chrono Clayborn cliche coca coDirector Colville commcent commlink Coores Cradar's cronkite*

*dearie der dismissively downside earmouths Earthborn ecosphere Edgington empathic enumerator Evenin everybody's eyestalk fibrosis Fickett FitzPatrick Fleance flutterbies forebrain Fuzzvoice Galactics Garand gassho Gillis Gioconda Glock gon gravities Grombaugh's Gundersen haid Haken hardwired Hari Hawkinsroshi Hobson's Holloways homespuns Hong Hronk's humaniform Imaro Immustim Injuns instinctual interferometer invectives irresolvable Jessamine Jura laundromat levelly lieto longstanding looker lowermost Maarten's Macalester Macedo madonna magickians Maldorit Malick manufactory Marchand Mata med messer millewettra mitral Moir Molson's Mondes mornin morningstar Mornington mudra Muhammed Myolock Nanotechnology neohorses neurowand NumberCrunch oldtimers opposable Orientator outworld paincults Pappy parasail Patapsaco Pellar's playgroup Postulants practicum presidente primitivist prob prog progeria protege pseudointellectual Rabinowitz's radiometer rainwalls rapturist Rashtapurdi's Ravagger's reinversion Remaldori retarget ridin riveras Rosenfield's runthrough sayin Schemuel Seavey's senor sequitur shapers sheba Shimon Signoria Signorina skinsuit snark sono spaceport spect spectrometer Spiegel Stapp starfarer starfarers Starminds sternal Stivra's subtext suer Sunny's Superdees suppertime sutliff swiftkiller Symes synth Taemon's Tarnation tectonic teleport teleportation Tenger Terrans tetched thinkin thymidine Tigernach's tol tonelessly Trancedancing Trancing trang trank Trev trillionaire Trippy's tugbots velcroed videophone virophages virtsuit watchfinger Whattaya Willem Witherspoon Woody's Worldlifter wristphones Yessir Youg *And actinic Aerospaceport afore afterbrain afterimages aimin airbag airlocks airtanks Alamos albergo Alemans alienness Alopecia amyotrophic annihilations anthropomorphize Ariana aroun Atherton autodoc autodrive backlit Bandello's Bangtailed Barrington beachmasters beardless bedchamber begat better'n bitty blocky Bodhisattvas bolus bombshelter botherin Bots bottega Bova's boyo brainscrubbers Braithwaite Braxton Brillig Brunner bushmaster Busker's cabman cafe's Californey cambots Camsofetu carabiner Carondelet carrozza carter Cascadian's Cass's Cathee cenacolo centerboard chappie Chien Ching chingaso chon-*

drite Choptank civilisation cliffside Columbian combuster cometary Command's Connelly conoscenza conspiratorially Coombs's Corrigan coulda couldn'ta coupla Crabbin crescendoed Crewson crosswinds cyborg cyclos Dade daresay darlin Dassau datastar deckhouse deepswimmer defibrillator deliverymen deponent deskpad detainees deus diddley Dikwan Dikwanic Diluminum distractedly Donelson's doppler doss doughnuts downriver draper draper's Dread's Dreamtime duena dystrophy Earthe Echinacea Ehrmann's elis encyb excellency Extraterritorial eyestalks favore Feffeti fella firebox fittin flatlined floormat Foote Fredericksen's Frederickson frigging fuzzies gaijin ganciclovir general's Generalist Geoffroy Gerkel Godalming goddammed Goodwife Grampa Grazie gridwork growler guard's guessin Gunderson Gunn's haircare Hallowells handbasket Hankster haoles Harcombe hardpan Haswell hauta hawklike hayrick heared heareth hearin Heatherington's Heinemann hellacious Helmstreit's Hephaestos highhanded highres Hilo hitman's Holo holomaps hovercar How'm hunnert hypnointerrogation ichneumon illogic imager Immiscible impellers inartistic incomprehension Indio individual's inexpressibly infrasonic inna insid Interlocutor interspecies ischemic Ishido isosuits Jabir Jasons Jenithar's Jeruve jes jillion Johansen Junoesque Kanemura Keeler Kenworth Kenworth's Kerm keypads killingly kinase kinesthetic Klien knobots Kuiper laborsaving Laney's Las laserburst Lassiter Lattyak's legalities legbands Lenclud loonie lounger loveseat Lunarian's machina Madre mage magickal magickian's Maheyna malachite malassadas Mamikonian's manta Marsport mastuh Mater Maynerd's McClernand's mechkan Med megabucks Merde Meurchong microwaldos midsection milord mindlinked miso missileer molto Moonbather's motherlode movin Murrays myocardial nacelles nada nanobombs nanobots nanotech nanotechnological nattering nebulosity Nellis nestmates neurasthenia neurocyberneticist Newcomb newsies nightdress nightstand noncommital Nothin nucleotides occipital Octahedron's older'n Olney oneway onsite Ordway orientator outgunned painblock Pappadopoulos parafoil parasails parrotted Paxton Paxton's Pedone Penfield Petalmouth Petroski phaged phys pickin Pigwhistle planetwide plannin pluribus Podleader's polyes-

terD5 positionin potser preprogrammed prominences proximal pseudoscience psychoactives psychoneurological quasidocs rab Rashta Ravenna Ready's rearview reattach Reboot recycler recyclers reengineering reflectance repressurizing rescript reticule retrovirus rivera sa've sailplane Samos Santees Sapiens sashimi Savannavong scareder scarp scavengefilters Scritch seatmate semistable senorita Senorita Shara's Shew shipful shiva sho Sho Shoot/Don't shootin shorthanded skeletally Sketchbook skrot skyfall's slidewalk slidewalks slipstream Slowdowsky Smartass sniffer soldi Somebody's spaceborne spacesuited Splendo Splo Springer stairstep Starseed Starship Starships startup starvoyager stat steepled stockbroking sua subsonics Suburban's suitbag Supercollider superconducting suppressors Susquehenna switchplate symbiote symbology tabletop takin talkin tele teleology teleoperated telepresent telomer Tenniel terabytes thermocline thinkable Thornhart Thorson's thruster thumbprint thunk Thurston touchpads Trancedances Tricity troodontids Tunman's Twonk twonk twonked twonky Umberto Unbelieving uncrush unfazed unpatentable unreality unreason unsinkable unum upthrust Valdoria Vanna Vazleeng vibrosaw vid villagemen vip voce vous wahine waldo waterjet waterwheels Waterworld's wesort Westford's When's whispery whoever's wholewheat windblown Windowless wristphone's yang Yas Yawara yin Youghiogheny

C. Most Common Wanted Non-word Tokens from the Test Document

The tokens in this list are the most common of those which are excluded from the auxiliary lexicon of the test text by reason of a high Index of Peculiarity, but which are still wanted words, ranked in order of occurrence and then alphabetically. There are about six hundred and forty entries, each of which appears at least twice, representing seven thousand four hundred instances in the text.

A further three thousand tokens appear only once in the test text and are not listed for reasons of space.

Rhys Dykstra Tsong vez Zee Urbanek Gyez Peggol Nikhil Hab Jynis Cygne Kabalevski Cuvey Bekwe Toglo Swayzey Brasn Dobbs Kuntz Lofosa zev McAndrew Vixa Bok Yoshi MacTavish Tson D.K Liem Dokhnor Kuntz's Kuskov gyrbird Pamdal Rhys's koprit Jarret Krobir Nocso Eltsac Kam Nakamura's Zugart Sadiq Thidkog Dykstra's kya Nakamura DDs Xena Dohlen FTL Mazra'ih Kleinschmidt Scoutkn Wulgo Al-Hariri Lil Sopsirk's Maprab MiningCorp RTRNRD Zosel Jeeger Kugel Ahthossio BSI Foglio Nikhil's gyrbirds Gyez's Kamorg Krepalgan L.tern Morgaf OROM Aneroi DD Fidela HCLU Pfister Pirelli Stryker Wojciech Bortav Cosba L.D Morgaffo yez Iyusha MGAD DI GM Hauptman Ri'C'ka U.S Abednego HQ Ish Stendahl's Zaffar Baha'is DA er J.J NIHT O.M SM bloke D.C Dokhnor's Hmm Kya Mbene Metz VR Bekwe's BeneCon CDROM CEO CO2 Cuvey's de FTI Heathkit Heathkits Hmmm Krepalga Linz McGonnigle Megtish's Mihdi PaLoana Reitz Stavro y'know FEL Foglio's Helmut Kalkt McLuan N.C OEV pc RTV Zaggisi #6 CLE Daoine EEG

EMT Gleege HiD J.G Koski Lipinski Morgaffos Nakahara Pirelli's Sidh T50 Ti8ner Tsadrat VNM VXN Wau Yoshi's Yubble Aames Armsby-Potter C02 DT etc gyrbird's Iyushan Jynis's Kabalevski's Kurbnesh Lompoc Mankiewicz Manlinkov MiningCorp's na NSF Ph.D Roker Tanaka Toshikawa TriV Tsuru VNMs Yulia 21st A.M AIB Aiyar Aw Cygne's Eckler Exoderm GOP Gorshkov's Halfax Karlburg Kleinschmidt's LCD McHenry Menk's mm Nachtegall's Nital NM nonEcom PC Pfister's Potos PR Quozel Shiites sss SwanCorp Tau Treevz Umeki uvvers 20th Al-Hariri's B-nath Baha'u'llah CD CPU D.K.'s d'you DAR DNR Dobbs's Dykdar en freeqing G-BOAG GM's Hazen Imbrium Jepson jerboa Kuskov's levtubes Lofosa's ol Orcas oryx Owada P.M PIs Raja's SDI Shi'ihs SIV Swayzeys Vixan Zebediah 3D Absaroka AIs AISOS Akka anti-Semitism bad'ha C'mon Ceti CPR da DC deja DNNR dollybird Dubuisson DV EC EEs Gadzang Gleeger Iaca II/West IQ IR Juhasz Kasmirski Korpi L.B Leclerc's Mihdi's MINERs Morgalfo Munira nav new-tech O'Dell Onikawa Pa-Lili's Pamdal's Peggol's Quozel's RAF Schwarzchild Shiite SLC2 ta Tartesh's Te Toglo's Uist UK upslope Urbanek's Uthgar volnoost vu WFA Wolfson WordForm Yah Zakorodsky 12th 429th 42nd 51/2 AB Adolfs biohazard BNDR Bubka bux CDs Charykova CO ColSec D.I Dahl Dalorz Davney DNCR Doktor eo-oak ET Euphrosyne Fa'a Faizi Fizgig Frau Fussom G-BOAC Gedde HDTV higgledy-piggledy Hmp hogtie IESC Iyushans Kahn Kappa Kawa Kkuuz Koprit Koprowski Kwathumba Kydak Lisbit-analog Lochboisdale LongDrink macaques MessyDos Mmm Moby Morgaffo's MSDOS munchkin Mungo Naw non-Einsteinian oryxes PackRat Pavlat pseudograv R&D RAE Riko Rockoon RT S.T.L.O Sopsirk t'is TAS TransSolar TriVid TV's Var Vazir vidphone Vite Voski VR-analog VRs Whaddya Yuhals zhburtz'ming Zucker Zzqolin #6's 1'd 112945.8G 14th 23rd 25A 40s 522nd 70s 90s a.c A.v.H AAAAARGH ABM AC ACB AFB Ahthossio's Aljira Aneroi's ANGELES AO AP AR Arecibo Armsby-Potter's ASAP Ashtawena AV AWOL B.C BA002 Bakhash BC Berdiaeff blokes BOAC Botvinnik BP Brax C.B C.E c'mon CALIFORNIA Cap'n Cassiiiieeee CDROMs CinderLand cm co Comp-Scan corn/soy Cosba's CP/M cps crabwise cul-de-sac cultch D.I.'s D.IC DA's DARs

DC10 DC3 DCO DD's De DEC DMV Dohlen's ECO EMP En-sam en-seem-oh EO etcetera EuroMart exozootic exozootics extra-Qushonian exurbs faerie faery FarCry Fitz Fizgig's Fogazzaro full-throated g's Gaia Gawd Gaxton Geb Geez GEO Git GPS gravlamps Gurrgh Gyrbirds half-witted helium3 Helium3 Hiro HIV HM hmm Hmph HP HPs HUD Husayn IRA IV/40th IV/West IVs Jaynsie Jeeger's Kamorg's Kaoriko Kloss Kraczyk Kraczyk's Kreekan L-fing's L.'s L.A L.D.'s L5 levstrips levtube li'l LOS LunaSat M.I.T Maprab's Mbene's McCrae MDOS Megalonyx MIDI MINER-8's Mirza Mmhm Nagamatsu nah Nah NAKAMURA Nakano Niven Nocso's non-Terrene Noooo ohgod ok op Orbidyne os Pa-Kai PaKai pc's PETROGRAD Pfui Phee Piblok-toq Piltdown PIO Plaatz PLISS Redbrae Ryuku S.O Sadiq's Safwat saltbush SAM satnav Seb shish-kebab SingerLabs SNAK@UNMRS.SPACENET Sopkirks ssss SST Stryker's SubLearn SysOp th Ti8nern too'c'ha't'ikoo Tsad Tsonvar TSR2 tsrial U.N umganu un Uranian USF USport USSSG Uthgar's V'schuntlik VCRs Vez vid-wall vid-xmit viewgraph VIP Vix Vixas's Vlahos Vogel Vorgrim's VTOL walkie-talkie wi wouldja Wulgo's Wytabit Xenobiological yah Ye've yoxen Yulia's Zhuitossne Zurich Zylecki Zylecki's

D. An Alternative Approach to Building a Lexicon

In Ringlstetter et al. (2007) a novel approach to building a lexicon is used. They identify the need for a lexicon tailored to a document - document-centric is their term - and they attempt to create a 'perfect dictionary' by using Internet resources and a search engine.

In both my method and Ringlstetter et al.'s, it is assumed that an OCR text will contain tokens in one of three classes: words (present in a local lexicon), non-words, and domain-specific tokens which may be either words or non-words. However, their method relies on generating the lexicon from an external reference rather than from the domain-specific tokens in the document itself. It is therefore able to work with a small text rather than requiring thousands of tokens.

To identify domain-specific words, Ringlstetter et al. follow a complex five-step process (they work in both English and German; for simplicity I discuss only the English steps here):

- A grammatical tagger is used to identify word bigrams of simple and compound nouns. Nouns from the default lexicon are accepted and referred to as 'query atoms'.
- The query atoms are grouped with others to create queries in such a way as to ensure that each atom is represented at least once in each query.
- Each query is sent to a search engine - in this case, 'AlltheWeb' - and as

many as the first thirty hits collated. Documents with a large numbers of orthographic errors are excluded. An error dictionary is maintained based on these orthographic errors and used to exclude tokens found within them (so a returned document with many errors is excluded; where only one or two errors are present, only those tokens are excluded).

- Formatting metadata is removed from the returned documents, and they are ordered in descending similarity to the original document. The lexicon is then constructed: if an ordered document contains at least one token which is not already present in the lexicon but which *is* in the original text, all the tokens in the document are added to the lexicon; otherwise, the lexicon remains as is and the next document is considered.

- The frequency of each token in the lexicon is collated.

In this way, a lexicon is created consisting of tokens found in documents essentially similar in subject matter to the original text, which can be reasonably assumed to be in the same domain. Word-trigrams are used to fingerprint the original text to ensure that the same document (or very close relatives) are not included in the returned documents and so do not bias the results.

Therefore, a domain-specific ordered list of tokens is created which can be assumed to match closely tokens which might be found in the original text and used both to identify non-words and to assist with their correction.

E. A Simple Estimation of the Probability of OCR Damage to a Wanted Non-word

While the errors caused by the OCR process are generally concentrated in areas where there are defects in either the printing or the scanning processes, with a large enough corpus they can be assumed to be randomly distributed throughout the text.

The corpus consists of a number of discrete texts of varying length; Church (2000) and de Roeck (2004) show that the wanted non-words will tend to be clustered and frequent within a text (rather than distributed smoothly throughout the corpus)[1] and on that assumption it becomes difficult to characterise the effect of the OCR errors. If a short story of only a few pages is part of the corpus, a single badly printed page can damage a significant number of instances of a wanted non-word. It is therefore assumed for the purpose of calculation that the non-words are smoothly distributed, and noted that the results may give a lower probability than is actually the case. To calculate this accurately would require further information on the length of individual works, on the number of wanted non-words within each work, and the number of badly-printed pages; these are beyond the scope of this discussion.

Working on a general OCR claim of 99% accuracy - another assumption: that this applies on a per character basis - and an average English word length of five

[1] This is intuitively correct; an unusually-named character in one work of fiction is rarely to be found in work by different authors.

characters, the chance of an error in a particular word is:

$$P_{error} = (1 - 0.99) * 5 = 0.05$$

To be included in the auxiliary lexicon - i.e. to be considered a 'wanted' non-word - a token not only has to be sufficiently similar to the English used in the training text, but must also occur at least twice in the corpus. If a wanted non-word is reduced to fewer than two instances by damage, it will not be included - with a consequent reduction of the effectiveness of the process.

For a particular token the probability of it being damaged to such an extent is obviously less the more frequent the token is:

$$P_{damage} = P_{error}^{(instances-1)}$$

and the number of tokens added to the auxiliary lexicon is reduced by the sum of the expected number of damaged tokens for each instance - the expected number of damaged tokens being:

$$T_{damaged} = P_{damage} * n_{tokens}$$

where n_{tokens} is the number of instances multiplied by the number of different tokens.

Table E.1 indicates that on the OCR text we can expect sixty-six tokens to be excluded from the auxiliary lexicon. For tokens represented four or more times, the number of expected exclusions is less than one, and ignored in the total.

Examination of the data - both test and training - indicates that the actual numbers of tokens excluded in the OCR data are approximately twice as great: 125 for the training text and 121 for the test text. This probably indicates that either the accuracy of the OCR system is not as high as assumed, or that the distribution of errors, as discussed above, is not uniform - probably both. As an example consider

Instances of token	Different tokens	P_{damage}	Excluded tokens
2	642	0.05	64
3	280	0.0025	2
total			66

Table E.1.: Predicted exclusions from the auxiliary lexicon due to OCR damage.

error density on Figure 4.10 on page 54 where '*Kugel*' appears three times, but two instances are damaged.