
Adaptation of Statistical Language Models for Automatic Speech Recognition

Philip R. Clarkson

Darwin College, University of Cambridge,
and
Cambridge University Engineering Department.



Dissertation submitted to the University of Cambridge
for the degree of Doctor of Philosophy

Abstract

Statistical language models encode linguistic information in such a way as to be useful to systems which process human language. Such systems include those for optical character recognition and machine translation. Currently, however, the most common application of language modelling is in automatic speech recognition, and it is this that forms the focus of this thesis.

Most current speech recognition systems are dedicated to one specific task (for example, the recognition of broadcast news), and thus use a language model which has been trained on text which is appropriate to that task. If, however, one wants to perform recognition on more general language, then creating an appropriate language model is far from straightforward. A task-specific language model will often perform very badly on language from a different domain, whereas a model trained on text from many diverse styles of language might perform better in general, but will not be especially well suited to any particular domain. Thus the idea of an *adaptive* language model whose parameters automatically adjust to the current style of language is an appealing one.

In this thesis, two adaptive language models are investigated. The first is a mixture-based model. The training text is partitioned according to the style of text, and a separate language model is constructed for each component. Each component is assigned a weighting according to its performance at modelling the observed text, and a final language model is constructed as the weighted sum of each of the mixture components.

The second approach is based on a cache of recent words. Previous work has shown that words that have occurred recently have a higher probability of occurring in the immediate future than would be predicted by a standard trigram language model (Kuhn and De Mori, 1990; Kuhn and De Mori, 1992). This thesis investigates the hypothesis that more recent words should be considered more significant within the cache by implementing a cache in which a word's recurrence probability decays exponentially over time.

The problem of how to predict the effect of a particular language model on speech recognition accuracy is also addressed in this thesis. The results presented here, as well as those of other recent research, suggest that perplexity, the most commonly used method of evaluating language models, is not as well correlated with word error rate as was once thought. This thesis investigates the connection between a language model's perplexity and its effect on speech recognition performance, and will describe the development of alternative measures of a language model's quality which are better correlated with word error

rate. Finally, it is shown how the recognition performance which is achieved using mixture-based language models can be improved by optimising the mixture weights with respect to these new measures.

Declaration

This thesis is the result of my own original work, and where it draws on the work of others, this is acknowledged at the appropriate points in the text. Some of this work has been published previously in conference proceedings (Clarkson and Robinson, 1997; Clarkson and Rosenfeld, 1997; Clarkson and Robinson, 1998). The length of this thesis, including appendices and footnotes is approximately 35,000 words.

Acknowledgements

I must begin by thanking my supervisor, Tony Robinson. His advice, support and enthusiasm have guided this research from the outset. It has been both a great privilege and enormous fun to have worked with him.

I also owe an enormous debt of gratitude to Roni Rosenfeld of Carnegie Mellon University. During my PhD years I have visited CMU twice – once for the summer of 1996, and later in the spring of 1998. The effect on my research of the knowledge and experience I gained during these visits cannot be overstated. I am immensely grateful to Roni for giving me these opportunities, for the faith he showed in me, and for continuing to provide advice and information over e-mail and the telephone. Many other people at CMU also contributed to making my time there a rewarding and enjoyable experience, especially Kristie Seymore and Rosie Jones.

The computing facilities within the speech, vision and robotics group in Cambridge are outstanding. Patrick Gosling does a magnificent job ensuring that these run smoothly, and that the resources are fairly allocated. I am very grateful to him. Thanks must also go to Mavis Barber for being immensely helpful and cheerful while cutting through the departmental red-tape.

This research would not have been possible without the financial support provided by the Engineering and Physical Sciences Research Council. Additional funding came from the Engineering Department, the European Language Resources Association and the Royal Academy of Engineering, and enabled me to attend several international conferences. I gratefully acknowledge the support of all these bodies.

My thanks also go to those within the speech group who have proof-read portions of this thesis – James Christie, Gary Cook, Jason Humphries, Thomas Niesler, Harriet Nock, Klaus Reinhart and Ed Whittaker all provided valuable feedback. I'd also like to thank them, along with Beth Logan, Parham Zolfaghari and Silke Witt for making the speech group a more enjoyable place to work.

I have been very lucky to have been at Darwin College during the last few years. It has provided a wonderful escape from work, especially during the last months of my PhD. Even while mired in thesis-related stress, I was still thoroughly enjoying my time in Cambridge. Thanks for this go to all my friends at Darwin. Particular thanks must go to Nic McNaughton for being a wonderful friend and house-mate.

The final thank you goes to my parents, for their constant love and support.

Notation

\mathcal{M}	A language model
\mathcal{E}	A probabilistic event (e.g. the occurrence of a unigram or trigram)
\mathcal{O}	Observed acoustic information
\mathcal{W}	A word string of arbitrary length
w_i^j	A word string consisting of the words w_i, w_{i+1}, \dots, w_j
PP	Perplexity
E	An equivalence class to which a word string may belong
\mathbf{V}	The set of words in a vocabulary
V	The vocabulary size
R	The number of words in a training corpus
$C(\mathcal{E})$	The number of occurrences of event \mathcal{E} in a sample
$C^*(\mathcal{E})$	The discounted count of event \mathcal{E} in a sample
$I_{\mathcal{E}}$	An indicator function, which equals 1 if \mathcal{E} occurred, and 0 otherwise
n_r	The number of events \mathcal{E} satisfying $C(\mathcal{E}) = r$
$P(\mathcal{E})$	The actual probability of event \mathcal{E}
$\hat{P}(\mathcal{E})$	An estimate of $P(\mathcal{E})$ according to some arbitrary model
$\hat{P}_{\mathcal{M}}(\mathcal{E})$	An estimate of $P(\mathcal{E})$ according to model \mathcal{M}
d_k	The discounting coefficient applied to counts of k
$\alpha(w_i^j)$	The back-off weight for the context w_i^j
r	The Pearson product-moment correlation coefficient
r_s	The Spearman rank-order correlation coefficient
T	The Kendall rank-order correlation coefficient

Contents

1	Introduction	10
1.1	Motivation for language modelling	10
1.2	Language modelling for speech recognition	11
1.3	Rule-based and statistical language models	12
1.4	Problems addressed by this thesis	13
1.5	Thesis organisation	13
1.6	Summary	14
2	Language Modelling Theory	15
2.1	Measuring the quality of a language model – Perplexity	15
2.2	Word history equivalence classes	18
2.2.1	<i>N</i> -grams	18
2.2.2	Alternatives to <i>N</i> -grams	19
2.3	Smoothing	23
2.3.1	Linear interpolation	24
2.3.2	Discounting	24
2.3.3	Backing-off	26
2.4	Alternatives to perplexity	26
2.5	Language model adaptation	29
2.5.1	Cache-based language models	29
2.5.2	Mixture-based language models	31
2.6	Methods of combining information sources	32
2.6.1	Linear interpolation	33
2.6.2	Backing-off	33
2.6.3	Maximum entropy	34
2.6.4	Log-linear interpolation	35

2.7	Decoding, <i>N</i> -best and lattice rescoring	35
2.7.1	The decoding process	36
2.7.2	<i>N</i> -best rescoring	38
2.7.3	Lattice rescoring	39
2.8	Summary	39
3	Construction of Language Models	40
3.1	Background	40
3.2	Preprocessing tools	41
3.2.1	Vocabulary generation	41
3.2.2	Open and closed vocabularies	41
3.2.3	Context cues	42
3.2.4	<i>N</i> -gram files	42
3.3	Language model construction	42
3.3.1	<i>N</i> -gram storage and cutoffs	42
3.3.2	Calculating discounting coefficients	44
3.3.3	Calculating back-off weights	44
3.4	Summary	45
4	Experimental Paradigm	46
4.1	Introduction	46
4.2	The broadcast news corpus	47
4.3	The British national corpus	47
4.4	Baseline language models	48
4.5	Speech recognition experiments	48
4.6	Summary	49
5	Mixture-Based Language Models	50
5.1	Introduction	50
5.2	Clustering algorithm	51
5.3	Component language model construction	53
5.4	The topic homogeneity of the components	54
5.5	Perplexity experiments	55
5.6	Speech recognition experiments	59
5.7	Summary	60
6	Cache-Based Language Models	61
6.1	Introduction	61
6.2	Perplexity experiments	62
6.2.1	Regular cache	62

6.2.2	Decaying history	64
6.3	Speech recognition experiments	70
6.3.1	Regular cache	70
6.3.2	Forward-backward cache	70
6.4	Summary	71
7	Perplexity and Word Error Rate	73
7.1	The effect of an errorful initial transcription	73
7.1.1	Supervised adaptation	74
7.1.2	Corrupting the adaptation text	75
7.1.3	Perplexity computed on the reference transcription	76
7.2	The use of a more general language model	77
7.3	Same perplexity language models	78
7.3.1	Construction of same perplexity language models	78
7.3.2	Estimating the number of words correct	79
7.3.3	Estimating the functions f and g	80
7.3.4	Results	83
7.3.5	Discussion	84
7.4	Summary	85
8	Alternative Language Model Evaluation Schemes	86
8.1	Introduction	87
8.1.1	The use of the whole distribution	87
8.1.2	Measures of correlation	88
8.2	Language model test set	89
8.3	Proposed language model evaluation measures	92
8.3.1	Perplexity	92
8.3.2	Rank-based measures	95
8.3.3	Entropy	96
8.3.4	Low probability estimates	96
8.4	Combining measures	97
8.4.1	Correlation of features	97
8.4.2	Combination of log probability and entropy	98
8.5	Application to language model development	99
8.6	Summary	101
9	Conclusions and Future Work	103
9.1	Review of original contribution	104
9.1.1	Adaptive language modelling	104

9.1.2	Perplexity and word error rate	105
9.2	Future directions	106
9.2.1	Mixture-based language models	106
9.2.2	Cache-based language models	107
9.2.3	Measures of language model quality	107
9.3	Outlook	108
A	The Matched Pairs Sentence Segment Word Error Test	109

Introduction

1.1 Motivation for language modelling

Ever since humans have been interacting with computers there has been a drive to allow communication between the two to occur in a more convenient way. Over the last thirty years, keyboards and mice have replaced punch-card readers as the means of entering data and commands into computers. Yet we are still communicating with the computer on its own digital terms.

In the late 1990s it is becoming realistic to expect to be able to interact with computers in a more human-like way. Users would like computers to be able to recognise their speech and handwriting, and to understand their language. However, none of this can be achieved with any real degree of success unless the computer is equipped with a good deal of knowledge about human language. Human language is vastly more complex than any of the digital codes which have been used to communicate with computers in the past.

Language modelling is the attempt to encode linguistic knowledge in a way which is useful for computer systems which deal with human language.

This thesis is about the use of language modelling for automatic speech recognition. Speech recognition is concerned with the process of converting an acoustic signal containing speech data into the appropriate text transcription (see Figure 1.1). Human-like speech recognition performance cannot be achieved by considering the acoustic signal alone – some form of linguistic knowledge is essential. Moreover, language modelling is not merely useful in order to improve the performance of a speech recognition system. It is necessary in order to be able to distinguish between homophones such as MEET and MEAT, a distinction which would be impossible using only the information contained within the acoustic signal.

Speech recognition is currently the most common application for language

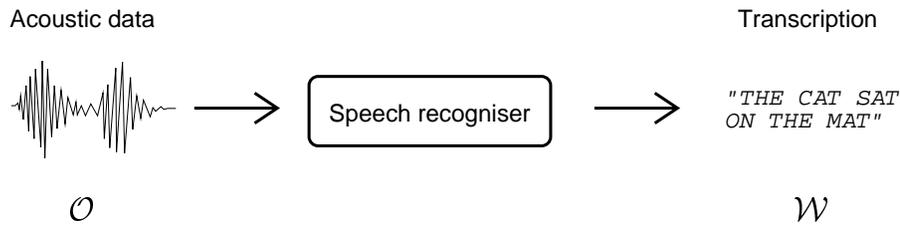


Figure 1.1 *The speech recognition process*

modelling, but it is far from the only one. Systems for optical character recognition, machine translation and spelling correction can all benefit from the application of linguistic knowledge.

1.2 Language modelling for speech recognition

An automatic speech recognition system receives a speech signal as input, and provides as output the best string of words which it can find to match this input. This is illustrated in Figure 1.1.

Put more formally, given observed acoustic data \mathcal{O} , it is the task of the recogniser to find the most likely word string¹, that is the word string \mathcal{W}' such that

$$\mathcal{W}' = \underset{\mathcal{W}}{\operatorname{argmax}} P(\mathcal{W} | \mathcal{O}). \quad (1.1)$$

Now, by Bayes' law, $P(\mathcal{W} | \mathcal{O})$ can be written as

$$P(\mathcal{W} | \mathcal{O}) = \frac{P(\mathcal{O} | \mathcal{W})P(\mathcal{W})}{P(\mathcal{O})}. \quad (1.2)$$

Since $P(\mathcal{O})$ is independent of the word string that is selected, it can be disregarded, and the problem is reduced to finding \mathcal{W}' such that

$$\mathcal{W}' = \underset{\mathcal{W}}{\operatorname{argmax}} P(\mathcal{W} | \mathcal{O}) = \underset{\mathcal{W}}{\operatorname{argmax}} P(\mathcal{O} | \mathcal{W})P(\mathcal{W}). \quad (1.3)$$

It is the role of the acoustic model component of the recogniser to generate an estimate for $P(\mathcal{O} | \mathcal{W})$, whereas it is the language model's responsibility to generate an estimate for $P(\mathcal{W})$.

¹Note that this seeks to minimise the *sentence* error rate, rather than the *word* error rate which is the measure of recognition accuracy used in this thesis. In many cases, however, the two are strongly correlated (Stolcke et al., 1997).

1.3 Rule-based and statistical language models

This thesis is concerned with *statistical* language models; models which can provide an estimate for $P(\mathcal{W})$. These models are constructed by examining large corpora of text for patterns and regularities, in a process known as *training*. This approach frequently horrifies those who lean more towards linguistics than engineering. Their argument is that language has structure and grammatical rules which can't be described using numbers and probabilities, and that one can't learn about language by simply counting words and phrases.

The alternatives which are often proposed are *rule-based* models which are generated by linguistic experts. Suppose that linguists produced a model which encoded a complete grammar of English, which contained all the words in the language and rules indicating the ways in which these words can legally be combined. Such a model could receive a sentence as input, and provide a "yes/no" answer about its validity². Even if such complete linguistic knowledge were available (linguists are, in fact, some way from providing a complete grammar of English), such information is clearly not useful in the context of providing an estimate of $P(\mathcal{W})$, as required by equation (1.2). This does not mean, however, that rule-based models should be rejected immediately. For example, a recognition system might use a statistical language model to generate several hypotheses about what was said, and as the next step all of the hypotheses that a rule-based model deems ungrammatical could be rejected.

It is not clear that ungrammatical sentences should always be rejected, however. Much use of natural language is ungrammatical, particularly in certain contexts. For example, while a speech recognition system designed to transcribe dictated business letters might usefully reject ungrammatical sentences, a system which used speech recognition to provide subtitles for television programmes will certainly be required to process ungrammatical input.

The basic criticism of statistical language models still holds, however. Language does indeed have far more structure than is encoded in the language models which will be described in this thesis. But this is a criticism of the statistical language models which are currently used, rather than of statistical language modelling in general. Who knows how much more structure we will be able to encode in statistical language models in the future?

²In reality, the distinction between rule-based and statistical language models is not this clear cut. Models exist, for example, where the rules as to the way in which linguistic elements can be combined are hand-coded, but their probabilities of execution are estimated from text corpora. See Section 2.2.2.3 for further details.

1.4 Problems addressed by this thesis

Current speech recognition systems tend to be heavily targeted towards one particular task. For example, the work in this thesis focuses on the recognition of broadcast news. The language models used in such systems are specific to the style of language used in the task. As such, when presented with language from another domain, performance can frequently degrade very badly. In order to perform speech recognition on language from arbitrary domains, one can either generate a model which is trained on material from a specific domain, with the result that the model's performance will be good for text from the same domain, but poor for more general text, or one can train a model on text from many diverse sources, which will perform better on general text, but will not be especially well suited for any particular domain. Clearly, the ideal would be a general language model whose parameters could be automatically tuned according to the style of text it is attempting to model. It is these *adaptive language models* which are the primary focus of this thesis.

This thesis also addresses the problem of how to predict the effect of a language model on speech recognition accuracy. Complete recognition experiments are computationally costly, and thus language models are often evaluated in terms of a measure called *perplexity*, which is based on the likelihood that the model assigns to some hitherto unseen text, and which is much less computationally demanding to compute. However, the results presented in this thesis, as well as in other recent research, suggest that perplexity is not as well correlated with recognition accuracy as was once thought. This thesis investigates the discrepancy between recognition performance and perplexity, and describes the development of new evaluation measures which are better correlated with recognition accuracy.

1.5 Thesis organisation

In Chapter 2, an overview of current language modelling theory is presented. Chapter 3 describes the techniques used to construct N -gram language models (by far the most commonly used language model in large vocabulary speech recognition), in the context of software which was developed to facilitate this research. The next chapters describe the experimental work carried out for this thesis. Chapter 4 sets the scene by describing the structure of the baseline language models, and the corpora used to train and test them. Chapters 5 and 6 describe experiments with two adaptive language models, namely mixture- and cache-based language models. In Chapter 7 the lack of correlation between

perplexity and speech recognition performance is discussed, and in Chapter 8 alternative evaluation measures for language models are developed. Finally, in Chapter 9 the conclusions and directions for future work are presented.

1.6 Summary

This chapter has introduced the ideas which are central to language modelling and has detailed language modelling's relation to automatic speech recognition. It has also introduced the problems which will be addressed by this work, and outlined the structure of the remainder of this thesis.

Language Modelling Theory

The previous chapter introduced the basic concepts of language modelling and explained the role of language models in large vocabulary speech recognition. This chapter will discuss the theory of language modelling in more detail, outlining the traditional problems inherent in statistical language modelling and the techniques commonly used to overcome them.

2.1 Measuring the quality of a language model – Perplexity

The utility of a language model is intrinsically linked with the effect which it has on the accuracy of a speech recogniser. This accuracy is generally measured in terms of *word error rate*, which is defined as the total number of errors (word insertions, deletions and substitutions) divided by the total number of words actually said. A related measure is *words correct*, which records the proportion of words which were correctly recognised, and therefore ignores insertion errors.

Recognition experiments are computationally costly, however, and so other methods of evaluating a language model's quality are typically used. This section will describe *perplexity*, the most common of these, present its theoretical foundation and discuss its drawbacks.

Perplexity is a measure of language model quality based on the concepts of information theory, and on the concept of entropy in particular. A brief review of the concepts is given below; for more details see (Cover and Thomas, 1991).

Within the framework of information theory, an information source is viewed as being a device which emits symbols chosen from a finite set \mathbf{V} . For example, the symbols might be words, and the set \mathbf{V} the vocabulary. The information

provided by this source is measured by its *entropy*. If the information source emits symbols x with probability $P(x)$, and these symbols are emitted independently of one another, then its *entropy* is given by:

$$H = - \sum_x P(x) \log_2 P(x). \quad (2.1)$$

Since the logarithm in equation (2.1) is taken to the base 2, the information provided by a source is measured in bits. The fundamental coding theorem of information theory states that on average H bits are required in order to represent a symbol emitted from a source of entropy H (Shannon and Weaver, 1949).

H is maximised when all symbols are emitted by the source with equal probability, that is $P(x) = 1/|\mathbf{V}|$ for all $x \in \mathbf{V}$. Thus a source with entropy H contains as much information as one which emits symbols with equal probability from a set of size 2^H .

General sources (such as language) will not output their symbols independently. If a general source emits symbols x_1, x_2, \dots, x_n , then its entropy is given by the following generalisation of equation (2.1):

$$H = - \lim_{n \rightarrow \infty} \frac{1}{n} \sum P(x_1, x_2, \dots, x_n) \log_2 P(x_1, x_2, \dots, x_n) \quad (2.2)$$

where the sum is taken over all possible sequences x_1, x_2, \dots, x_n .

If the source is ergodic (that is, sufficiently long sequences allow one to draw conclusions about its statistical properties), then equation (2.2) is equivalent to

$$H = - \lim_{n \rightarrow \infty} \frac{1}{n} \log_2 P(x_1, x_2, \dots, x_n) \quad (2.3)$$

and if n is sufficiently large, this can be approximated by

$$H \approx - \frac{1}{n} \log_2 P(x_1, x_2, \dots, x_n). \quad (2.4)$$

If the information source is one which emits words from a vocabulary, then H is the number of bits required to specify a word. Thus entropy is a measure of the difficulty of performing speech recognition on this language, since on average H bits of data must be extracted from the acoustic data in order to determine a specific word.

The exact probability of the word sequence cannot be known, but as was shown in Section 1.2, a statistical language model assigns a probability of $\hat{P}(\mathcal{W})$ to a word string $\mathcal{W} = w_1^n$. If w_1^n is a hitherto unseen word string (referred to as the *test text*) and n is sufficiently large to allow the test text to be representative of the language, then as far as the speech recognition system is concerned, a measure of the difficulty of the recognition process is given by

$$\hat{H} = -\frac{1}{n} \log_2 \hat{P}(w_1^n). \quad (2.5)$$

If n is sufficiently large, and the language source is ergodic, then it will always be the case that $\hat{H} \geq H$. Thus the lowest possible value of \hat{H} can only be achieved using a perfect model of the language. Therefore, a good model is one which has a low entropy, and hence assigns a high probability to the test text.

The quality of a language model is generally measured in terms of its perplexity (PP) (Bahl et al., 1983), which is related to \hat{H} as follows:

$$PP = 2^{\hat{H}} = \hat{P}(w_1^n)^{-1/n}. \quad (2.6)$$

Therefore a low value for perplexity corresponds to a language model that is in some sense “good”. However, this does not mean that it is good in the sense of leading to accurate speech recognition systems. This is the key weakness of perplexity as a measure of a language model’s utility. In seeking to minimise perplexity, all that is achieved is the generation of a language model that in some very general sense *models language* better. If one is concerned with reducing the number of errors made by speech recognition systems then one should also consider the acoustic similarities between words: a language model which was capable of discriminating between acoustically similar words would be more useful than one which was not. Similarly, those concerned with language modelling for optical character recognition should take account of the visual similarities between words and characters. Furthermore, the accuracy of a speech recognition system (or many other applications of language modelling) will depend not only on the probabilities assigned to the correct hypothesis, but also on the probabilities assigned to all other candidate hypotheses which will be competing with it. These probabilities are ignored by the perplexity measure.

Section 2.4 will describe previous work on alternatives to perplexity which attempt to overcome these drawbacks. However, in order to provide some context for that, it is first necessary to introduce some further language modelling concepts.

2.2 Word history equivalence classes

In Section 1.2 it was shown that the role of the language model is to provide an estimate of $P(\mathcal{W})$. If the word string \mathcal{W} is w_1^n , then its probability may be decomposed as follows:

$$P(w_1^n) = \prod_{i=1}^n P(w_i | w_1^{i-1}) \quad (2.7)$$

and so the problem of assigning a probability to a word string can be reduced to that of assigning a conditional probability to each word in the word string given its *word history*.

Plainly, however, one cannot hope to enumerate all possible word histories of any reasonable length. For this reason, word histories are partitioned into *equivalence classes* E_1, E_2, \dots, E_m such that each possible word string w_1^n belongs to one and only one equivalence class. Then, if $w_1^{i-1} \in E_j$, we have

$$\hat{P}(w_i | w_1^{i-1}) = \hat{P}(w_i | E_j). \quad (2.8)$$

The most common method of partitioning the word histories is by the use of N -grams, where the histories are partitioned according to their final $N - 1$ words. This section will describe this approach in detail, discuss its major advantages and drawbacks, and describe some of the alternative methods of partitioning word histories into equivalence classes which have been proposed.

2.2.1 N -grams

An N -gram language model partitions word histories according to their last $N - 1$ words. So, if for example $N = 3$ then the two word histories u_1^n and v_1^m will be in the same equivalence class if and only if $u_{n-1}^n = v_{m-1}^m$. Thus the word probabilities are calculated as follows:

$$\hat{P}(w_i | w_1^{i-1}) = \hat{P}(w_i | w_{i+1-N}^{i-1}). \quad (2.9)$$

The choice of N is based on a trade-off between detail and reliability, and will be dependent on the quantity of training data available. A *bigram* ($N = 2$) language model will have larger equivalence classes, and hence fewer parameters than a *trigram* ($N = 3$) model. A bigram model's parameters will therefore be more reliably estimated, while a trigram model will be more precise, and

will therefore be a more accurate model, provided there is sufficient training data.

For the quantities of language model training data typically available at present, trigram models strike the best balance between precision and robustness, although interest is growing in moving to 4-gram models and beyond.

The simplicity of the N -gram model is both its greatest strength and weakness. On the positive side, the model's simplicity means that the models are easy and efficient to train, even when corpora of hundreds of millions of words are used. The models can therefore be based on a large amount of real data, and simultaneously encode syntax and semantics. In addition, the models can be easily included in the decoder of a speech recogniser, something which is not true of more complex models. The negative side, however, is that the current word is clearly dependent on much more than the previous one or two words. It is easy to construct a sentence which is entirely nonsensical or at least ungrammatical, but which consists entirely of plausible trigrams. This can lead to recognition errors; an example given in (Chase, 1997) shows the words HE IS A BRILLIANT FINANCIAL ANALYST recognised as HE IS A BRILLIANT FINANCIAL ANALYSTS – a clearly ungrammatical sentence which one would hope a more sophisticated language model would cause to be rejected. It is the inability of N -gram models to consider these *long-range dependencies* that is their key disadvantage. Furthermore, the N -gram approach can fragment the data unnecessarily. There is clearly something wrong with a model which considers the two histories GOLD PRICES FELL TO and GOLD PRICES FELL YESTERDAY TO as unrelated (Rosenfeld, 1994), as a trigram model would.

2.2.2 Alternatives to N -grams

2.2.2.1 Class-based N -grams

Class-based N -gram models differ from the word-based models described above by defining a mapping of the vocabulary words into a smaller number of *classes*. The N -grams are then based on these classes, rather than the words themselves. The classes are sometimes linguistically motivated, and correspond to the word's part-of-speech, or are sometimes automatically derived from the language model training data.

If the assumption is made that the mapping of words to classes is one-to-one, i.e. each word w belongs to exactly one class $g(w)$, then the class-based N -gram can take one of many forms. For example, when dealing with a trigram, the class-based model could define any of the following:

$$\hat{P}(w_i | w_1^{i-1}) = \hat{P}(w_i | g(w_{i-2}), g(w_{i-1})) \quad (2.10)$$

$$\hat{P}(w_i | w_1^{i-1}) = \hat{P}(w_i | g(w_{i-2}), w_{i-1}) \quad (2.11)$$

$$\hat{P}(w_i | w_1^{i-1}) = \hat{P}(g(w_i) | g(w_{i-2}), g(w_{i-1}))\hat{P}(w_i | g(w_i)) \quad (2.12)$$

In practice, it is the last of these options which is the most commonly used.

Some class-based models have allowed words to belong to more than one class. The reasoning behind this is that words (especially in English) can belong to many linguistic groups. For example, the word LIGHT can be a verb, noun, adjective or adverb.

If $G(w)$ is the set of classes to which the word w can belong, then the equivalent of equation (2.12) is

$$\hat{P}(w_i | w_1^{i-1}) = \sum_{\substack{g_i \in \\ G(w_i)}} \sum_{\substack{g_{i-1} \in \\ G(w_{i-1})}} \sum_{\substack{g_{i-2} \in \\ G(w_{i-2})}} \left[\hat{P}(g_{i-2} | w_{i-2}) \hat{P}(g_{i-1} | w_{i-1}) \right. \\ \left. \hat{P}(g_i | g_{i-2}g_{i-1}) \hat{P}(w_i | g_i) \right] \quad (2.13)$$

It is clear that the additional flexibility which results from allowing words to belong to multiple classes is offset by a large increase in computational complexity.

There are many different approaches to the problem of assigning words to classes. Firstly, one can use a corpus in which all of the words have been manually tagged according to their part-of-speech or linguistic function. Such corpora include the Lancaster-Oslo-Bergen corpus (Johansson et al., 1986), or the British national corpus (see Section 4.3). Alternatively, one can manually group together all words which one believes will behave in a similar fashion. Examples of such classes might be days of the week or numbers. An obvious situation in which this might be applied is the ATIS task, which involves the recognition of travel queries. Classes consisting of airline names or cities could be defined (Rosenfeld, 1994). Finally, one can automatically generate classes by observing patterns in the training data and clustering words appropriately. Many automatic clustering approaches have been investigated, for example (Brown et al., 1992) and (Ney et al., 1994).

Class-based models have several advantages over the word-based equivalents. Firstly, since the possible number of histories for the model is greatly reduced, it becomes much more compact. Viewed another way, the model could now be expanded to use more context. For example, a class-based 4-gram model might be approximately the same size as a word-based trigram.

An example of this is described in (Niesler and Woodland, 1996), in which the class-based N -grams are allowed to be of arbitrary length, the context length being increased until no further predictive power is gained. For the Wall Street Journal corpus, this technique results in the context being extended sufficiently far that some 7-grams are retained in the model. The resulting models are considerably more compact than word-based models, and offer comparable performance.

An additional benefit of the reduced number of contexts is a reduction in the problem of data sparsity. Since there are generally many fewer classes than words in the vocabulary, the number of potential N -grams in the model is greatly reduced. Furthermore, class-based models are able to make much more reliable probability estimates for events which were not seen in the training data, since even if a word N -gram is unseen, the equivalent class N -gram is likely to have been seen, and thus be accurately predicted by the model. For this reason, class-based models have been particularly successful in situations where limited quantities of training data were available.

The obvious disadvantage of class-based models is that they lose some of the semantic information which makes the word-based model so powerful. This is, of course, another example of the “generality vs. specificity” tradeoff that is a recurring theme in language modelling. In this case, however, the problem can be partially overcome by constructing language models which combine information from word-based and class-based models. A more critical problem is that class-based models still fail to overcome the key disadvantages of word-based models. While they are able to use somewhat more context than word-based models, the problems of long-term dependencies (whose effects can span an arbitrary number of words) are still not addressed.

2.2.2.2 Tree-based language models

In (Bahl et al., 1989) a language model is presented which adopts a radically different approach from the N -gram model. The model is based on a binary decision tree. At each node of the tree, a “yes/no” question about the word history is asked, and the answer to these questions determines the path taken down the tree from the root node to a leaf. Thus the word histories are partitioned into equivalence classes by the decision tree. At each of the tree’s leaves, a probability distribution is defined over all the words in the vocabulary. The model that is proposed achieves a moderate improvement in perplexity over a trigram model trained on the same data.

The key difficulty with this approach is that of training a “good” tree from the language modelling training data. The list of potential questions one could

ask at each node is so vast (“Was the previous word a verb?”, “Was there any computer jargon in the previous fifty words?”, etc.) that searching through the space of potential trees requires a vast amount of computational effort. Clearly, some reduction of the set of possible questions is necessary, and it is in this simplification that the tree-based models lose much of their power.

It may be the case that better trees can be developed, and will result in useful language models. So far, however, the work which has attempted to extend that of (Bahl et al., 1989) has met with limited success (Rosenfeld, 1997).

2.2.2.3 Probabilistic context-free grammars

A context-free grammar (CFG) consists of

- A set of terminal symbols, which generally consist of the words of the language (e.g. DOG, ATE)
- A set of non-terminal symbols, which correspond to grammatical objects (e.g. NP, VP, NOUN)
- Rewrite rules, which specify how these symbols can be related (e.g. $VP \rightarrow VERB NP$, $NP \rightarrow DET NOUN$, $NOUN \rightarrow DOG$)

A CFG assigns one or more structures to any sentence which is valid according to the grammar’s rules. These structures correspond to the syntactic parses of the sentence. An example is shown in Figure 2.1.

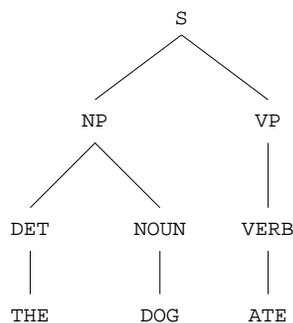


Figure 2.1 A syntactic parse of the sentence *THE DOG ATE*

A probabilistic context-free grammar (PCFG) assigns probabilities to each of the rewrite rules. Thus each parse of a sentence has a probability associated with it, and the sum of these can be considered to be the probability of the

sentence¹. In this way, a PCFG can be used as a language model.

Algorithms exist to train PCFGs by estimating appropriate rewrite rule probabilities. This is achieved by seeking to select probabilities which maximise the likelihood of the training corpus. At present, however, these algorithms are very computationally demanding, and have not seen much use except on very small training corpora².

PCFGs are a syntactic approach to modelling language. As such, they suffer from the same problem as class-based N -gram models, namely that they lose much of the useful semantic information that word-based N -gram models encode. This, in combination with the computational effort required to both train and use the models have limited their use in large vocabulary tasks. However, PCFGs have the advantage that they are linguistically motivated, and hence capture the long-range dependencies that are ignored by N -gram models. They have been used successfully in some smaller vocabulary tasks (see, for example (Wright et al., 1992)).

A more detailed discussion on the training and use of probabilistic context free grammars can be found in (Charniak, 1993).

2.3 Smoothing

The key difficulty with using N -gram language models, as well as many of the alternatives, is that of *data sparsity*. One can never have enough training data to estimate all of the model's parameters reliably.

The maximum likelihood estimate for the probability of an event \mathcal{E} which occurred $C(\mathcal{E})$ times out of a possible R is

$$P(\mathcal{E}) = \frac{C(\mathcal{E})}{R}. \quad (2.14)$$

In a sparse sample, however, the maximum likelihood estimate is biased high for observed events and biased low for unobserved ones. In an N -gram language model it is impossible in practice to avoid the problem of unseen events. For example, if a vocabulary of 64,000 words is used, then there will be 2.62×10^{14} possible trigrams. Even a training corpus of 100 million words can contain

¹In practice, the number of possible parses can grow exponentially with the length of the sentence, and so summing their probabilities becomes very inefficient. Efficient algorithms for finding the most probable parse exist, which partially circumvent this problem.

²Note, however, that if human knowledge is used to limit the set of rewrite rules which can be applied, then larger corpora can be used to estimate the rewrite rule probabilities (Charniak, 1993)

at most 0.000038% of these trigrams. If a trigram never occurs in the training text, then the method of maximum likelihood estimation will assign any string which contains the trigram a probability of zero, and it will not be correctly transcribed by the speech recognition system. Furthermore, the perplexity of the model with respect to any text which contains this string will be infinite (see equation (2.6)).

This section will discuss techniques which are used to *smooth* the data to correct the bias of the maximum likelihood estimate, and to ensure that no word strings are assigned zero probabilities.

2.3.1 Linear interpolation

One of the most simple ways of avoiding zero probabilities is to define the N -gram probabilities to be a linear combination of the 1-gram, 2-gram, . . . , N -gram maximum likelihood probability estimates. So if one is dealing with a trigram model the trigram probability is a linear combination of the unigram, bigram and trigram maximum likelihood probability estimates:

$$\hat{P}(w_i | w_{i-2}^{i-1}) = \lambda_1 \frac{C(w_i)}{R} + \lambda_2 \frac{C(w_{i-1}^i)}{C(w_{i-1})} + \lambda_3 \frac{C(w_{i-2}^i)}{C(w_{i-2}^{i-1})} \quad (2.15)$$

where $\sum_i \lambda_i = 1$, R is the number of words in the training text, and λ_1 , λ_2 and λ_3 are chosen to maximise the likelihood of some held-out text (Jelinek, 1990).

2.3.2 Discounting

Discounting is a more principled method of correcting the bias towards observed events of maximum likelihood probability estimates. An event's count is *discounted* by multiplying it by a discount coefficient $d_{C(\mathcal{E})}$, where $0 \leq d_{C(\mathcal{E})} \leq 1$ for all $C(\mathcal{E}) \geq 1$. So the discounted count is

$$C^*(\mathcal{E}) = d_{C(\mathcal{E})} C(\mathcal{E}). \quad (2.16)$$

The discounted probability (cf. equation (2.14)) is

$$P(\mathcal{E}) = \frac{d_{C(\mathcal{E})} C(\mathcal{E})}{R} \quad (2.17)$$

and the remaining probability mass is distributed among unseen events.

2.3.2.1 Good-Turing discounting

The most commonly used discounting scheme in language modelling is based on Good-Turing discounting (Good, 1953), and was first applied to language modelling by Katz (Katz, 1987) who used it in conjunction with backing-off (Section 2.3.3). The modification of Good-Turing discounting proposed by Katz defines

$$d_r = \frac{\frac{(r+1)n_{r+1}}{rn_r} - \frac{(k+1)n_{k+1}}{n_1}}{1 - \frac{(k+1)n_{k+1}}{n_1}} \quad (2.18)$$

(where n_r is the number of events occurring exactly r times) for $r < k$ (where typically $k \approx 7$) and $d_r = 1$ for higher counts, the belief being that events occurring more than k times are well estimated by maximum likelihood.

This approach does have some disadvantages, however. For example, it is necessary that $d_r > 0$ for all r , and this puts some constraints on the relative values of n_1, n_2, \dots, n_{k+1} . In general, these constraints will be satisfied by naturally occurring data but may not be if one has doctored the data in some way (for example by boosting the counts of some subset of the N -grams).

2.3.2.2 Witten-Bell discounting

In (Witten and Bell, 1991) a discounting scheme is described in which the discounting coefficient is dependent not on the event's count, but on t , the number of distinct events which followed the particular context. So, for the bigram "A B", t is the number of distinct bigrams of the form "A *" which occur in the training data. This was first applied to language modelling by (Placeway et al., 1993). It sets

$$d_r(t) = \frac{R}{R + t} \quad (2.19)$$

This is motivated by a desire to assign probability estimates to unseen events which reflect how many times novel events have been seen in the past. t is viewed as being the number of times that a novel event has previously been observed following a particular context.

2.3.2.3 Absolute discounting

Absolute discounting involves subtracting a constant b from each of the counts. It therefore defines

$$d_r = \frac{r - b}{r}. \quad (2.20)$$

It has been shown (Ney et al., 1994) that setting $b = \frac{n_1}{n_1 + 2n_2}$ is approximately optimal in terms of maximising the leaving-one-out likelihood.

2.3.2.4 Linear discounting

In linear discounting (Ney et al., 1994) a quantity proportional to each count is subtracted from the count itself. That is, d_r is set to be a constant value for all r . If, for example, one wanted to select the constant to be such that linear discounting assigns the same probability to unseen events as Good-Turing discounting, one would define

$$d_r = 1 - \frac{n_1}{R}. \quad (2.21)$$

2.3.3 Backing-off

The principle behind *backing-off* (Katz, 1987) is that if one has insufficient data in a language model to accurately estimate a word probability, then one should *back-off* to a less specific language model. One could, for example, back-off from a word-based trigram to a class-based trigram (Niesler, 1997). In practice, however, it is more usual to back-off from an N -gram to an $(N - 1)$ -gram model.

Typically, the conditions for backing-off are that the N -gram whose probability is sought does not occur in the model. So, if one has a trigram language model, then

$$\hat{P}(w_i | w_{i-2}^{i-1}) = \begin{cases} \frac{C^*(w_{i-2}^i)}{C(w_{i-2}^{i-1})} & \text{if } C(w_{i-1}^i) \geq 1 \\ \alpha(w_{i-2}^{i-1}) \hat{P}(w_i | w_{i-1}) & \text{otherwise} \end{cases} \quad (2.22)$$

where $\alpha(w_{i-2}^{i-1})$ is the *back-off weight*, and is chosen to ensure that $\sum_{w \in \mathbf{V}} \hat{P}(w | w_{i-2}^{i-1}) = 1$.

2.4 Alternatives to perplexity

Section 2.1 introduced *perplexity*, a widely-used measure of language model quality. It was shown that while it is theoretically well-founded, it has a number

of drawbacks. Several techniques have been proposed to try to overcome these drawbacks, and these will be described in this section.

(Ferretti et al., 1989) proposes a measure called *speech decoder entropy*, which attempts to jointly estimate the performance of the language model and acoustic model, while taking account of the interactions between the two. While entropy (and hence perplexity) is concerned with the information provided by the language model, speech decoder entropy is calculated from the amount of information provided jointly by the acoustic information and the language model. The approach has significant drawbacks, however. The calculation of speech decoder entropy either requires data from an audio recording of the test text, or the application of certain simplifying assumptions. In the case where the audio data is available, it is shown that on a set of six trigram language models, speech decoder entropy is slightly better correlated with word error rate than perplexity is. However, in the case where a recording is not available, and simplifying assumptions have to be made, the new measure predicts word error rate no better than perplexity.

Researchers at IBM have investigated a measure called *aperplexity* or *acoustic perplexity* (Jelinek, 1990). This is the average perplexity of the language model when its choice is limited to words which are acoustically confusable with the correct word. However, studies described in (Ferretti et al., 1989) showed that this measure is frequently simply proportional to perplexity.

(Bahl et al., 1989) claims that recognition errors are strongly correlated with very low language model estimates. Therefore language models are compared not only on the basis of perplexity, but also on the proportion of words in the test text which are assigned a probability estimate below a certain threshold. Word error rates resulting from their language models are not provided, however, so it is not clear how strongly this measure is correlated with recogniser performance. This measure is investigated further in Section 8.3.4.

More recently, the lack of correlation between perplexity and word error rate has received more attention. In (Iyer et al., 1997) experiments are described which aim to determine features which might predict word error rate better than perplexity. The focus is shifted away from global measures which consider the test text as a whole (e.g. perplexity) and towards local measures. These local measures aim to predict the likelihood of each individual word being correctly recognised. The features used to predict this likelihood for a particular word include not only the word's language model probability, but also information about the acoustic realisation of the word (for example, its length in phones), the word's part of speech, and various other factors. Techniques

are described for building a decision tree which, given a word's features from a pair of language models can be used to predict the relative likelihood of that word being correctly recognised. The decision tree can thus be applied to each word in the test text to provide an estimate for the relative recognition accuracy resulting from the use of the two models. It is shown that the output from the decision tree is better correlated with word error rate than perplexity is on a test set of 13 model pairs (taken from a set of 8 models).

(Chen et al., 1998) examines the lack of correlation between perplexity and word error rate and attacks it from several directions. The aim is to develop measures which are similar to perplexity, in that they are based entirely on the language model information and disregard information from the acoustic component, but which are better correlated with word error rate. 35 language models are constructed, and are divided into two groups, one containing language models of the same structure (that is, they are all N -gram models), the other containing a more disparate collection of models. Perplexity and word error rate are shown to be well correlated for the former, but very poorly for the latter.

Furthermore, if the probability of a word being recognised correctly were a linear function of the word's log language model probability, then it can be shown that the proportion of words recognised correctly would be a linear function of perplexity. However, it is shown by (Chen et al., 1998) that a word's probability of being recognised correctly is not a linear function of the word's log language model probability, and using the data from their collection of models, a more realistic empirical estimate of the actual function is generated. Since it is then possible to estimate a word's probability of being correct based on its language model probability, an estimate of the overall proportion of words correct can be calculated. This measure is referred to as *M-ref*, and it is shown that for the disparate collection of models, it predicts word error rate more accurately than perplexity does. However, for the group of similar models, perplexity is a superior predictor of word error rate.

A further alternative to perplexity proposed by (Chen et al., 1998) is the use of *artificial lattices*³. This is an attempt to mimic the speech-recognition process without the need for such large computational overheads. For the group of similar language models the *artificial word error rate* which results from rescoreing a set of sparse artificial lattices is approximately as well correlated to actual word error rate as perplexity. However, artificial word error rate performs better on the more varied set of models.

³Lattices will be described in Section 2.7.3.

We will return to the theme of perplexity, its relation to word error rate, and possible alternative measures when relevant work carried out for this thesis is described in Chapters 7 and 8.

2.5 Language model adaptation

This chapter has so far shown how statistical language models generate their probability estimates based on a body of training data. The assumption throughout has been that the training text will be representative of the style of language which one is attempting to model. If this is not the case then the language model may be close to useless. For example, one of the more common corpora used to train language models consists of text from the Wall Street Journal newspaper. Trigram language models trained on this data have been used in speech recognisers which can achieve very high accuracy – providing the input speech is someone reading from the Wall Street Journal. However, when the system is faced with spontaneous conversational speech, the results are far less impressive. The Wall Street Journal uses a very different style of language to someone chatting with a friend. Similarly, the use of language differs according to the topic of discourse – the language used in sports reports is different from that used in political journalism, for example.

There are many millions of words of text available from the Wall Street Journal to allow appropriate language models to be trained. However, for most other styles of language this is not the case. Sometimes language modellers will be in the position of having enough training text to train a “general” language model, and a little training text which is specific to the recognition task. More commonly they will have only a “general” language model, and the only information about the topic or style of language being recognised must come from the recognition of the previous sentences. In all cases, however, the aim must be to use a language model which matches the target domain as closely as possible.

2.5.1 Cache-based language models

It is a commonly observed phenomena (see, for example, the description of the “Shannon game” experiments conducted at IBM in Chapter 2 of (Rosenfeld, 1994)) that words which have occurred recently are more likely to occur in future than would be predicted by a standard N -gram model. Cache-based language models are an attempt to exploit this fact. Typically a cache-based component is linearly interpolated with an N -gram language model:

$$\hat{P}(w_i | w_1^{i-1}) = \lambda \hat{P}_{\text{Cache}}(w_i | w_1^{i-1}) + (1 - \lambda) \hat{P}_{N\text{-gram}}(w_i | w_{i-N+1}^{i-1}). \quad (2.23)$$

The most common approach is that a cache of the previous K words is maintained, and a word's cache-based probability is computed as the relative frequency of the word within the cache. That is

$$\hat{P}_{\text{Cache}}(w_i | w_1^{i-1}) = \frac{1}{K} \sum_{j=i-K}^{i-1} I_{\{w_j=w_i\}} \quad (2.24)$$

where $I_{\mathcal{E}}$ is an indicator function which equals 1 if \mathcal{E} occurred, and 0 otherwise.

Such models were first proposed in (Kuhn and De Mori, 1990). A cache-based component was added to a class-based N -gram model. A 200-word cache was maintained for each class, and the interpolation weight (λ in equation (2.23)) was chosen for each class separately. This model resulted in a 14% decrease in perplexity over their baseline class-based language model (Kuhn and De Mori, 1992).

Several researchers have built upon the work of Kuhn and De Mori. The obvious extension has been to add a cache-based component to a word-based N -gram, rather than a class-based model. Typically this has resulted in a reduction of perplexity of between 10% and 15% (see, for example, (Iyer and Ostendorf, 1996; Schechtman, 1994)), although there have often not been corresponding reductions in word error rate ⁴.

The cache need not be limited to containing single words. Work has been conducted in which the probabilities of recently occurring bigrams and trigrams are also boosted (Jelinek et al., 1991; Pye and Woodland, 1996). This has met with fairly limited success in comparison to the results of the straightforward unigram cache, possibly because there is insufficient information in the previous few hundred words to reliably estimate bigram and trigram probabilities.

Cache-based language models can be extended further by the use of *trigger pairs* (Rosenfeld, 1994). Here a word's presence in the cache not only boosts its own probability, but also that of words which are related to it. Correlations between pairs of words are found in the training text, and strongly correlated

⁴The decrease in perplexity or word error rate is not solely a function of the usefulness of the cache component. It is also a function of the initial quality of the N -gram model, as a poor N -gram will be improved more by the addition of cache-based information. For example, the results given in (Jelinek et al., 1991) show a particularly large reduction in perplexity and word error rate, but this is likely to be due to the baseline language model being a class-based model trained on only 1.2 million words.

pairs are added to the list of trigger pairs. For example, the presence of the word PIANO in the cache might result in the probability of the word MUSIC being increased.

The common finding about trigger pairs is that far more information is contained in the “self-triggers” than in any others, with the result that very little is gained from their use as compared to the basic cache-based approach. Even the non-self-triggers which contain useful information tend to be same-root triggers: MUSIC triggering MUSICAL and MUSICIAN, for example.

Further extensions to cache-based language models are presented in Chapter 6 of this thesis.

2.5.2 Mixture-based language models

Mixture-based language models are based on a set of several component language models, each of which is specific to a particular topic or style of language. The probability estimates from these component language models are linearly interpolated (see Section 2.6.1) to produce the overall language model probability, with the interpolation weights chosen to reflect the topic or style of language currently being recognised. So if there are n component language models $\mathcal{M}_1, \mathcal{M}_2, \dots, \mathcal{M}_n$, then the overall language model probability is

$$\hat{P}(w_i | w_1^{i-1}) = \sum_{j=1}^n \lambda_j \hat{P}_{\mathcal{M}_j}(w_i | w_1^{i-1}). \quad (2.25)$$

In general, the first stage of creating a mixture-based language model is to partition the training text into some number of homogeneous components⁵. One has to decide how many mixture components into which to partition the training text, and this involves a tradeoff. Too many components will lead to each constituent language model being under-trained, and hence each of the constituent probability estimates will be poor. Conversely, too few mixture components will result in a model which is unable to distinguish between topics or linguistic styles as finely as one might wish. Note that there is no reason why a section of the text in the training corpus cannot appear in more than one component. So, for example, one of the components could contain all of the training text while the other components contain the training data partitioned according to topic. In this way the problem of poor constituent probability

⁵In some circumstances this may not be necessary. For example, the text may already be labelled according to its topic, source or style.

estimates in the case where there are many mixture components can be partially overcome.

Having partitioned the training data, an N -gram model can be constructed for each of the components.

When the model is used, each component model must be assigned an interpolation weight. These can be calculated using the expectation maximisation (EM) algorithm (Dempster et al., 1977) in such a way as to maximise the likelihood of some previously seen or held-out text.

(Kneser and Steinbiss, 1993) reports a 10% decrease in perplexity using mixture-based language models based on bigram models, and trained on a manually partitioned 1.1 million word corpus. (Carter, 1994) describes experiments using automatically derived clusters for the ATIS task. It is reported that such mixture-based models based on unigrams and bigrams lead to improvements over the equivalent baseline model, but this result does not translate to models based on trigrams. The ATIS task has very little training data, however, and thus trigram models trained on a fraction of this data will be severely under-trained. (Iyer et al., 1994) describes a mixture-based model with five mixture components. This achieves a small decrease in word error rate on the Wall Street Journal task. However, the components are assigned global weights over all articles in the test set, and so it does not constitute a truly adaptive model.

Research on mixture-based models which was carried out for this thesis is presented in Chapter 5.

2.6 Methods of combining information sources

Much of the previous section's discussion on language model adaptation described language models in which two or more information sources were combined. In the examples that were given, the sources were combined using *linear interpolation*. This was mainly for the sake of clarity; linear interpolation is not the only method of combining information sources, and it is certainly not optimal in many situations.

This section will go into more detail about linear interpolation, and describe its advantages and drawbacks. Alternative methods of combining information sources, including *maximum entropy*, will also be discussed.

2.6.1 Linear interpolation

A set of language models $\mathcal{M}_1, \mathcal{M}_2, \dots, \mathcal{M}_k$ can be combined by linear interpolation as follows:

$$\hat{P}(w_i | w_1^{i-1}) = \sum_{j=1}^k \lambda_j \hat{P}_{\mathcal{M}_j}(w_i | w_1^{i-1}) \quad (2.26)$$

according to the constraints $\sum_{j=1}^k \lambda_j = 1$ and $0 \leq \lambda_j \leq 1$ for all j .

The expectation maximisation (EM) algorithm (Dempster et al., 1977) can be used to generate a set of λ_j which maximise the likelihood of some held-out text. Providing this held-out text is sufficiently large and representative of the training text, these weights will be close to optimal for the test set.

Linear interpolation is widely used in language modelling. Its simplicity makes it easy to implement, use, and analyse, and any model can be included as a component. However, this simplicity is also its major drawback.

A language model's utility will vary widely according to context. Consider the case where a trigram model is interpolated with a cache-based component. If, for example, the word history ends with a bigram which occurred very infrequently in the training data, then it will not be possible to be confident in the trigram model's probability estimate. Ideally, its interpolation weight should be reduced to reflect this. Similarly, if only a few words of a new document have been seen, then the cache component is unlikely to provide useful information. However, when the interpolation weights are defined globally, as in equation (2.26), these effects cannot be captured.

These problems can be partially overcome by partitioning the word histories into equivalence classes, and allowing a different set of interpolation weights for each equivalence class. However, each equivalence class must have enough instances in the held-out text in order to accurately estimate optimal weights, and so one cannot partition the histories particularly finely.

2.6.2 Backing-off

Section 2.3.3 showed how backing-off can be used to overcome problems of data sparsity. In the case where there is insufficient data in a language model to generate a reliable probability estimate, it was shown that it was possible to back-off to a less specific language model (e.g. from a trigram to a bigram). In this way, the bigram and trigram information sources are combined. However, the models which can be combined with backing-off can be much more general

than this. For example, one could combine information from a topic-specific language model and a general language model by backing-off from the former to the latter in cases when the topic-specific model cannot provide an accurate probability estimate.

2.6.3 Maximum entropy

In the methods of combining information sources described above, separate models are constructed and their output is combined. Maximum entropy modelling is somewhat different – one model is constructed which seeks to combine the information from each of the sources. Each of the information sources contributes a set of constraints on the parameters on the model. Of the possible models satisfying the constraints, the one with the flattest distribution (that is, the highest *entropy*) is chosen. Thus, once the constraints are satisfied, nothing further is assumed about the data.

As an example, consider two language models, \mathcal{M}_1 and \mathcal{M}_2 . \mathcal{M}_1 is a standard bigram model, so

$$\hat{P}_{\mathcal{M}_1}(w_i | w_1^{i-1}) = f(w_i, w_{i-1}) \quad (2.27)$$

and \mathcal{M}_2 is a distance-2 bigram, which is defined by

$$\hat{P}_{\mathcal{M}_2}(w_i | w_1^{i-1}) = g(w_i, w_{i-2}) \quad (2.28)$$

These probabilities will almost certainly be inconsistent. Linear interpolate resolves this by averaging the two probability estimates, and backing-off chooses either one or the other. The maximum entropy principle combines all the information sources into one model. The constraints on this model are not that equations such as (2.27) and (2.28) are true for all possible histories, but the more relaxed constraint that they are true *on average* in the training data. That is, equations (2.27) and (2.28) are replaced by

$$E(\hat{P}_{\mathcal{M}_1}(w_i | w_1^{i-1}) | w_{i-1} = a) = f(w_i, a) \quad (2.29)$$

and

$$E(\hat{P}_{\mathcal{M}_2}(w_i | w_1^{i-1}) | w_{i-2} = b) = g(w_i, b) \quad (2.30)$$

Providing that the constraints are consistent, the set of models satisfying them will be non-empty. The next step in the maximum entropy approach is to choose the model from this set which has the flattest distribution, i.e. the maximum entropy model. This can be achieved using the generalised iterative scaling (GIS) algorithm (Darroch and Ratcliff, 1972). Given consistent constraints, a unique maximum entropy solution is guaranteed to exist, to which the GIS algorithm is guaranteed to converge. It is, however, a very computationally demanding algorithm. To train a maximum entropy model containing a full set of bigram and trigram constraints from a training corpus of hundreds of millions of words requires a vast amount of CPU. This is the key reason for the current lack of widespread use of maximum entropy modelling.

A detailed study of the GIS algorithm, and of further issues involved in maximum entropy language modelling is beyond the scope of this work. A more thorough treatment can be found in Chapters 4 and 5 of (Rosenfeld, 1994), or (Berger et al., 1996).

2.6.4 Log-linear interpolation

In (Klakow, 1998), a method of combining information sources called log-linear interpolation is presented. This maintains much of the simplicity of standard linear interpolation, and yet performs substantially better.

Instead of combining language model probability estimates as in equation (2.26), the following scheme is used:

$$\hat{P}(w_i | w_1^{i-1}) = \frac{1}{Z(w_1^{i-1})} \prod_{j=1}^k \hat{P}_{\mathcal{M}_j}(w_i | w_1^{i-1})^{\lambda_j} \quad (2.31)$$

where $Z(w_1^{i-1})$ is a normalisation constant chosen to ensure that $\sum_{w \in \mathbf{V}} \hat{P}(w | w_1^{i-1}) = 1$. There is, however, no constraint of the λ_j s, which are chosen to maximise the likelihood of some held-out text. Since this quantity is convex in all the λ_j s, this maximisation can be solved by any algorithm for multidimensional optimisation.

2.7 Decoding, N -best and lattice rescoring

In Section 1.2 it was shown in an abstract way how a language model interacted with the acoustic model within a speech recognition system. However, in order to understand how language models are used in practical systems, it

is necessary to understand the means by which a speech signal is *decoded* into one or more hypothesised transcriptions.

2.7.1 The decoding process

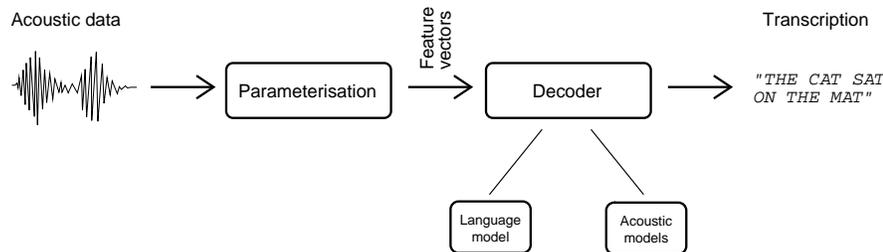


Figure 2.2 A typical speech recognition system

The input to a speech recognition system is acoustic data in the form of a waveform. This data is preprocessed by the system's front end to generate a set of feature vectors which capture the spectral characteristics of the acoustic signal at discrete time intervals. These feature vectors are then passed to the decoder.

It is the role of the decoder to search for the string of words which best matches the feature vectors, i.e. to find the \mathcal{W}' such that

$$\mathcal{W}' = \underset{\mathcal{W}}{\operatorname{argmax}} P(\mathcal{W} | \mathcal{O}). \quad (2.32)$$

where \mathcal{O} represents the feature vectors.

Since the search space is therefore the set of all possible word strings, it is necessary to find methods of reducing the size of this space to make the search tractable. Two such techniques are *path pruning* and *path merging*.

If a particular path through the search space is considered very unlikely at a particular time point, then it can be assumed that this particular path will not be the most likely at the end of the search. Under these circumstances the path can be *pruned*, and examined no further. By this method the search space can be kept down to a manageable number of paths at each time point. Note, however, that it is possible that a path which is very unlikely at some stage may become more likely relative to the other candidates later on. Thus it is possible for the best path to be pruned from the search space, leading to a *search error*.

If two or more paths converge (i.e. they hypothesise a word boundary at the same time point), then it may be possible to *merge* them, and continue from that

point with only the more likely of the paths, since a path which is less likely at the point where the paths converge will remain less likely. In order to do this, the paths to be merged must have equivalent histories as far as the language model is concerned. For example, if one is using a trigram language model, then one can merge the paths only if the final two words in the paths are the same. This is illustrated in Figures 2.3 to 2.6. Figure 2.3 shows two unmerged paths, and the following three figures show path merging in the cases where the decoding is based on unigram, bigram and trigram models respectively.

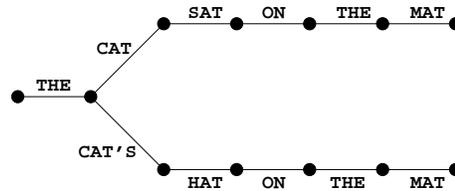


Figure 2.3 *Two unmerged paths*

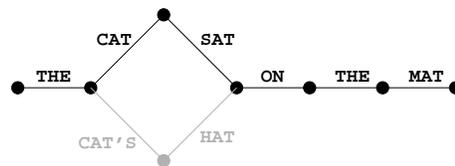


Figure 2.4 *Decoding with a unigram language model. Paths merged independently of their recent history*

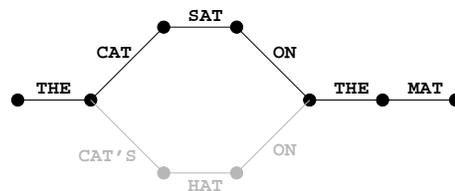


Figure 2.5 *Decoding with a bigram language model. Paths merged only if their previous word is the same*

Language models which employ more context, such as the tree-based model described in Section 2.2.2.2 cause problems with such decoding schemes, as it is not possible to perform path merging as frequently (if at all), and therefore the search space is likely to become unmanageably large.

Furthermore, some of the language models described above may have a much higher memory requirement than a standard N -gram model, which may

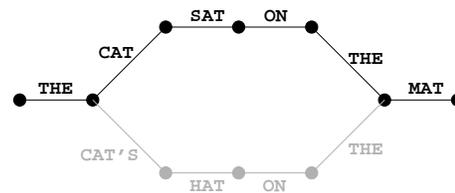


Figure 2.6 *Decoding with a trigram language model. Paths merged only if their previous two words are the same*

make it impossible to hold both the complex language model and acoustic model in memory. This would make it difficult to use these language models in the decoding step.

In order to overcome these difficulties, and to enable the use of extended-context or memory-intensive language models for speech recognition, the techniques of *N-best* or *lattice rescoring* are commonly used. These are two-pass approaches to speech recognition. The first pass uses a simple language model to generate a smaller, pruned search space. The hope is that the best hypothesis according to the final language model is not pruned out of this search space, but this cannot be guaranteed. The search space which results can take the form of an *N-best list* (Schwartz and Chow, 1990) or a *lattice* (Odell, 1995). These are then *rescored* using a more complex language model. This approach has the additional advantage that experiments which compare many language models can be run without duplicating the effort of computing the acoustic scores.⁶

2.7.2 *N-best rescoring*

An *N-best list* is simply a list of the *N* most likely hypotheses according to the initial recognition pass. Typically, each hypothesis will be accompanied by both its acoustic likelihood, and its probability according to the language model used to perform the initial decode.

N-best rescoring replaces these language model probabilities with the language model probabilities from an alternative (generally more complex) language model. The hypotheses can then be re-ranked according to their new probabilities, and the final recogniser output is the hypothesis which is now considered most likely.

⁶Note that by reducing the search space in this way, any improvements or degradations which are observed by rescoring are likely to be smaller than those which would have been observed had the models used in rescoring the lattice been incorporated into the initial decode. All recognition experiments presented in this thesis are based on lattice rescoring, and will thus be affected by this phenomenon.

2.7.3 Lattice rescoring

A *lattice* is a directed acyclic graph which contains the paths through the search space which were considered most likely by the acoustic and language models used in the initial decode. Each node is associated with a time, and corresponds to a hypothesised word boundary. Each arc of the graph is labelled with the word which is hypothesised between its nodes, and the corresponding acoustic and language model probabilities. The language model probabilities can be replaced with new ones from an alternative language model in exactly the same way as for N -best rescoring, and one can then search through the lattice for the path which is now considered most likely.

2.8 Summary

This chapter has presented an overview of language modelling theory. Perplexity, a measure of a language model's success, has been introduced and its drawbacks discussed. N -grams, the basis of the vast majority of current language modelling work, have been described, as have the reasons for their success. However, it has also been shown that they have several drawbacks. The first of these is data sparsity, and this chapter described the common methods of overcoming it. The lack of long-term memory of the N -gram is a further major drawback, and some methods of surmounting this have been discussed. These may either take the form of adopting a different language modelling structure (for example a decision tree-based model), or by employing topic adaptation techniques. It is these topic adaptation techniques which will form the focus of this thesis. The chapter concluded by describing methods of combining information from two or more language models, and the decoding and rescoring techniques by which language models are incorporated into a speech recognition system.

Construction of Language Models

This chapter describes the methods used to construct N -gram language models. It will describe the preprocessing steps required to extract the N -gram statistics from the training data, and the issues involved in converting this information into a language model. All of this is connected with software which was developed to facilitate this research (Clarkson and Rosenfeld, 1997), and which is now freely available to researchers worldwide.

3.1 Background

In 1994 Ronald Rosenfeld released the CMU Statistical Language Modelling (SLM) Toolkit, a publicly available set of Unix tools for the construction and testing of statistical language models (Rosenfeld, 1995). The tools were widely used and became the de-facto method of constructing trigram and bigram models.

In the two years following the release of the toolkit, however, the emphasis shifted somewhat in language modelling. Increasingly large corpora became available, and the computational power to which researchers had access also increased. Interest grew in moving beyond trigram language models towards 4-gram and 5-gram models. Furthermore, some of the toolkit's inefficiencies which were tolerable when dealing with small corpora became a significant problem when dealing with several hundreds of millions of words. In particular, much of the work which was carried out for this research required scores of large language models to be constructed, which would have required prodigious quantities of disk-space and CPU time using version one of the toolkit.

Version two of the toolkit was therefore developed in order to carry out this research more efficiently (Clarkson and Rosenfeld, 1997). The new version

is publicly available to researchers worldwide, and is even more widely used than its predecessor. The new version improves on version one in terms of both efficiency (the time required to create a language model has been greatly reduced) and functionality (more discounting schemes have been implemented, and users are no longer limited to the use of bigram and trigram models, but can use N -grams with arbitrarily large N).

3.2 Preprocessing tools

3.2.1 Vocabulary generation

A preliminary step in language model construction is the selection of an appropriate vocabulary. The vocabulary size will always be finite, and often constrained by factors within the recognition system, for example the set of words which exist in the pronunciation dictionary. Often, however, the vocabulary will simply consist of the most frequent words in the training data.

3.2.2 Open and closed vocabularies

All language models have a vocabulary of finite size, and as such are subject to the problem of out of vocabulary words (OOVs). Depending on the way that the language model handles the occurrence of such OOVs, its vocabulary can be said to be either *open* or *closed*.

A *closed vocabulary* model makes no provision for OOVs. Any such words which occur in the training or test data will cause an error. This type of model might be used in a command/control environment where the vocabulary is strictly limited to the commands used to operate the system, and it can therefore be guaranteed that the system will not have to process OOVs.

An *open vocabulary* model allows for OOVs to occur; all out of vocabulary words are mapped to the same symbol, typically denoted <UNK>. Generally this symbol is treated in the same way as any other word in the vocabulary. However, in some situations it may be the case that no OOVs occurred in the training data, yet one wishes to allow for the situation where they could occur in the test data. This situation could occur, for example, if a limited amount of training data is available, and it is therefore possible to choose a vocabulary which provides complete coverage of the training set. In this case, it is difficult to calculate an appropriate estimate for the probability of observing an OOV, and an arbitrary proportion of the probability mass is reserved for OOV words.

3.2.3 Context cues

Language data is viewed as a stream of words interspersed with *context cues*. These are markers which indicate events such as sentence, paragraph and article boundaries. They provide useful information to the language model (and may therefore affect prediction), but they should not be predicted by the model itself. Any subset of the vocabulary may be defined as context cues.

3.2.4 N -gram files

As the main unit of currency in these language models is the N -gram, the key preprocessing step is to generate a list of the number of occurrences of all the N -grams which occur in the training corpus.

These N -gram lists can be *vocabulary independent*, and contain every tri-gram in the training set, or *vocabulary dependent*, in which case all the OOVs are mapped to the same symbol, and the N -grams which are identical as a result are merged. It is these vocabulary dependent N -gram lists that are the input required to generate the language model.

3.3 Language model construction

From the vocabulary dependent N -gram file generated by the preprocessing tools, the task is now to generate a model which, given an M -tuple of words w_1^M (such that $M \leq N$) can generate a value for $\hat{P}(w_M | w_1^{M-1})$.

3.3.1 N -gram storage and cutoffs

It is frequently the case that all the N -gram count information must be stored in memory, both when the model is being constructed, and when it is used. A tree-structure allows this information to be stored in a compact way, although for large tasks, the memory requirement may still be unreasonable. For this reason, some N -grams are often removed from the model.

There are various methods of determining which of the N -grams should be omitted. The most common of these is the use of *cutoffs*, in which N -grams whose counts fall below a certain threshold are removed from the model. It should be noted, however, that the frequency-of-frequency information which is used to calculate the discounting coefficients (see Section 2.3.2) continues to take account of N -grams whose counts fall below the cutoff threshold.

Clearly, as the cutoffs are made more severe, there is a trade-off between the size of the language model and its quality. Table 3.1 shows how the bigram

and trigram cutoffs affect the size and perplexity of a trigram language model trained on the broadcast news corpus (which will be described in Section 4.2). The number of parameters of the models is defined as the number of bigrams and trigrams retained.

Cutoffs		Number of Parameters	Perplexity
Bigram	Trigram		
0	0	38,289,694	129.5
1	1	12,294,777	135.4
2	2	6,965,818	141.0
5	5	3,168,606	153.5
10	10	1,729,668	167.6
20	20	936,064	186.3
50	50	407,266	220.4
100	100	213,488	252.8

Table 3.1 *The effect of cutoffs on the size and perplexity of a trigram language model trained on the broadcast news corpus.*

Other methods of reducing the number of N -grams retained by the language model have been proposed. In (Seymore and Rosenfeld, 1996), a process is described whereby N -grams are selected for removal from the model according to the difference in the original probability estimate and the backed-off probability that would result should the N -gram be removed. It is shown that for language models of equivalent size, the new method of selecting N -grams for omission results in lower perplexities and word error rates. The approach described in (Stolcke, 1998) selects those N -grams whose removal makes least difference to the relative entropy between the original and pruned model. The two approaches are similar – the difference being that when an N -gram is removed, it will have an effect on the back-off weight (see Section 2.3.3), which is not accounted for by the method of (Seymore and Rosenfeld, 1996). However, the comparison of the two techniques described in (Stolcke, 1998) demonstrates that there is little practical difference between the two pruning schemes. Both techniques, however, perform slightly better than the straightforward use of cutoffs.

3.3.2 Calculating discounting coefficients

Having stored all of the N -gram information in memory, and calculated the appropriate frequency-of-frequency information (that is, the values of n_1, n_2, \dots in section 2.3.2), all that remains is to calculate the discounting coefficients and the back-off weights.

Given the appropriate frequency-of-frequency information, the discounting coefficients can easily be computed for any of the discounting schemes described in Section 2.3.2. There is, however, a slight caveat when using Good-Turing discounting (or any discounting method which applies only to counts below a certain threshold). Consider the word pair BUENOS AIRES. It is possible that this will occur sufficiently often in the training set to remain undiscounted. It is also possible that in the training data the word BUENOS is always followed by AIRES. Therefore, the language model probability estimate would be $\hat{P}(\text{AIRES} \mid \text{BUENOS}) = 1$, with zero probabilities for all other words following BUENOS. In order to overcome this problem, the counts of all histories which are not followed by any words whose counts in that context fall within the discounting range are incremented by one.

3.3.3 Calculating back-off weights

As was shown in Section 2.3.3, the back-off weights are necessary to ensure that the word probabilities sum to one. That is, $\alpha(a b)$ – the back off weight for the bigram $(a b)$ – should be chosen to ensure that $\sum_{w \in \mathbf{V}} \hat{P}(w \mid a b) = 1$. The values of these back-off weights can be computed efficiently. Consider the case of a trigram language model. If $U(a b)$ is the set of words which follow the bigram $(a b)$ in the language model model, then we require

$$\sum_{w \in \mathbf{V}} \hat{P}(w \mid a b) = \sum_{w \in U(a b)} \hat{P}(w \mid a b) + \sum_{w \notin U(a b)} \hat{P}(w \mid a b) = 1. \quad (3.1)$$

The value of $\sum_{w \in U(a b)} \hat{P}(w \mid a b)$ can simply be looked up, since all the relevant trigrams exist in the language model. $\sum_{w \notin U(a b)} \hat{P}(w \mid a b)$, on the other hand, becomes

$$\sum_{w \notin U(a b)} \hat{P}(w \mid a b) = \sum_{w \notin U(a b)} \alpha(a b) \hat{P}(w \mid b) \quad (3.2)$$

and assuming that the back-off weights for the unigrams have already been calculated, the values of $\hat{P}(w \mid b)$ for all w can easily be looked up. So

$$\alpha(a b) = \frac{1 - \sum_{w \in U(a b)} \hat{P}(w | a b)}{\sum_{w \notin U(a b)} \hat{P}(w | b)}. \quad (3.3)$$

Note that it is not possible to store the values of $\alpha(a b)$ for all possible $(a b)$. In fact, only the values for bigrams $(a b)$ which occur in the model are stored. This does not pose a problem, however, since if $(a b)$ doesn't occur, then the set $U(a b)$ is empty, and so $\alpha(a b) = 1$, and one can therefore directly use the bigram probability without the need to multiply it by a back-off weight. This is often viewed as a separate back-off case, but this is slightly misleading.

3.4 Summary

This section has detailed the steps and issues involved in creating back-off N -gram models. The distinction between open and closed vocabulary models has been explained. The use of cutoffs to reduce the size of the model has been described, as have the alternative methods of selecting N -grams to be omitted from the model. Furthermore, the method by which the discounting coefficients and back-off weights are calculated, and the use of context incrementing have been described.

Experimental Paradigm

This chapter details the experimental conditions under which the work for this thesis was carried out. It describes the nature of the broadcast news task, which is the focus of this work, and describes the manner in which the perplexity and word error rate experiments are conducted. Details of the two corpora used to construct the language models and evaluate their perplexity are given, and details of the vocabulary used by the system are presented. Finally, the framework for the word error rate experiments is discussed, and the system used to generate the lattices which are rescored is described.

4.1 Introduction

The experiments which were carried out during this research focussed on the broadcast news task, which forms the basis for the ARPA Hub-4 continuous speech recognition evaluations (Pallett and Fiscus, 1997). The task involves the transcription of a set of audio recordings of radio and television news broadcasts.

The broadcast news task was appropriate for this research for a number of reasons. Firstly, it contains speech from a wide variety of acoustic conditions, ranging from high quality speech in a studio to speech recorded over telephone channels, or speech with background noise or music. This suggests that any findings made during this research are likely to be applicable to many domains. Secondly, the broadcast news task is particularly suitable for language model adaptation experiments, as it contains a wide range of subject matter, and many different styles of language, ranging from prepared, read speech to spontaneous speech from an interview or phone-in. This indicates that there are potential gains to be made from language model adaptation. Finally, there exists a large

training corpus of suitable training data, which allows the adaptation techniques to be evaluated on a large data set.

The experiments were conducted by constructing language models from the training sets of the broadcast news corpus and the British national corpus, both of which are described in the following sections. The perplexity of these models was evaluated using their respective test sets, and their effect on word error rate by rescoring lattices from the broadcast news task.

4.2 The broadcast news corpus

The broadcast news corpus (Graff, 1996) contains transcripts of radio and television news broadcasts, and was created to be used as the baseline language model training data for the 1996 Hub-4 evaluation.

It consists of a total of approximately 148 million words, taken from radio and television programs broadcast on ABC, CNN, NPR and PBS between January 1992 and June 1996¹.

The corpus is supplied partitioned into a training set of approximately 130 million words, and a test set of approximately 18 million words.

Each point in the corpus where the topic being discussed changes is labelled, and this allows the corpus to be partitioned into topic-homogeneous articles. This is important for the construction of the mixture-based models described in Chapter 5. Using such a partition results in approximately 125,000 articles in the training set, and approximately 22,000 articles in the test set.

4.3 The British national corpus

The British national corpus (BNC) (Burnard, 1995) contains 110 million words of British English taken from diverse sources. 10 million words of the corpus were generated from transcribed spoken language, the remainder from written language. Both the written and spoken components were created from language generated by a wide variety of people whose first language was British English, and both components contain very varied styles of language. For example, the written component contains extracts from newspapers, academic books and popular fiction, memoranda, school essays, and many other kinds of text. The spoken component includes a large amount of unscripted informal conversation, together with spoken language collected in many different contexts, ranging from formal business or government meetings to radio phone-ins.

¹All text in the corpus pre-dates the shows used for recognition accuracy experiments

The corpus is split into 4,122 different articles, each containing text from a different source. For the purposes of the work conducted for this thesis, these articles are split at random into a training set of 3,737 articles, and a test set of 385 articles.

4.4 Baseline language models

The baseline language models used in this thesis are back-off trigram models, trained on either the broadcast news corpus or the British national corpus. They used the same vocabulary as the 1996 ABBOT broadcast news evaluation system (Cook et al., 1997), in order that lattice-rescoring experiments could be conducted using the lattices which had been generated using a version of that system. This vocabulary consisted of 65,531 words. The only context cue used was one indicating the start of a new sentence. Bigram and trigram cutoffs were set to one and Good Turing discounting was applied.

4.5 Speech recognition experiments

The speech recognition experiments were carried out through lattice rescoring (see Section 2.7.3). The recognition task was the development test of the 1996 Hub-4 partitioned evaluation. This consists of six half-hour shows, taken from the ABC, CNN and CSPAN television networks, and from National Public radio, comprising approximately 22,000 words.

The lattices were generated using a simplified version of the 1996 ABBOT broadcast news system (Cook et al., 1997). ABBOT is a large vocabulary continuous speech recognition system based on a connectionist/hidden Markov model architecture.

The system used to generate the lattices differed from the final evaluation system in the following two ways:

Language models The final evaluation system used separate language models for planned and spontaneous speech, with the appropriate language model being chosen based on the labels which accompany the test data. The system used to generate the lattices used for this thesis, however, used the same baseline language model throughout.

Acoustic models Channel adaptation was used in the final evaluation system such that different acoustic models were used for segments of the recognition task which used narrow bandwidth acoustic information (for example, speech over a telephone channel). Again, the appropriate models

were chosen based on the labelling of the test data. However, the system used to generate the lattices used in this work used the same acoustic models throughout.

The resulting lattices were grouped according to the article boundary markers which accompanied the data, such that each group of lattices contains a separate article. In this way, the language model adaptation which will be described in the forthcoming chapters could be performed on an article-by-article basis.

The lattice word error rate of these lattices (i.e. the error rate which would result if the best path through each of the lattices was chosen) was 7.0%, and the word error rate resulting in rescoring the language models with the baseline broadcast news language model was 37.9%².

4.6 Summary

This chapter has described the broadcast news task, and the training and test data and lattices used in these experiments. The details of the two corpora are summarised in Table 4.1.

	British national corpus	Broadcast news corpus
Training set size	105 million words 3737 articles	130 million words 125,000 articles
Test set size	10.5 million words 385 articles	18 million words 22,000 articles

Table 4.1 *Summary of corpus details*

²Note that this word error rate result, in common with all those presented in this thesis, was generated by simply aligning the first-best hypothesis from the recogniser with the reference transcription using dynamic programming. More elaborate scoring software, such as that provided by NIST for the broadcast news evaluation, was not used here.

Mixture-Based Language Models

This chapter describes the experiments involving mixture-based language models which were carried out for this thesis. It describes the clustering algorithms used to partition the training text into components, the details of the language models which were constructed for each of these components, and the way in which the component models were combined. The effect of the resulting mixture-based models were on both perplexity and speech recognition word error rate are investigated.

5.1 Introduction

Mixture-based language models were introduced in Section 2.5.2. They are based on the belief that many different topics and styles of language are represented in the language model training data, and that if one can increase the influence of the sections which are appropriate to the style or topic to which the language model is currently being applied, then the model will perform better. To this end, the training data is partitioned according to topic or style, and an individual language model is constructed for for each component. The final language model is then constructed by combining the output from each component model according to weights chosen to reflect the characteristics of the language style currently observed. In the experiments described in this chapter, the models are combined using linear interpolation, although one could also any of the methods of combining language models described in Section 2.6.

Put more formally, the training text is partitioned into k components, and a language model is built for each. The overall language model probability is then simply a linear combination of the component model probability estimates:

$$\hat{P}(w_i | w_1^{i-1}) = \sum_j \lambda_j \hat{P}_{\mathcal{M}_j}(w_i | w_1^{i-1}). \quad (5.1)$$

The major decisions that govern the design of such language models are

- Which text should be used to train each of the component language models?
- How should the mixture weights (the λ_j s) be chosen?

5.2 Clustering algorithm

The first stage in constructing a mixture-based language model is that of partitioning the training corpus into a set of topic- or style-homogeneous components. In order to do this, the corpus must be split into small segments whose topic or style can be determined. Each segment is then assigned to one or more components. Suitable segments might, for example, be the sentences or paragraphs of the corpus. The first decision that must be made, therefore, is the size of the segments into which the training text should be split. If the corpus is divided too finely (for example, at the sentence level) then it is likely to be difficult to identify the topic or style of each segment. Furthermore, the computational demands of the clustering algorithm will be high. Conversely, too coarse a segmentation may result in segments containing text from more than one topic or style.

Chapter 4 described the manner in which both the broadcast news corpus and the British national corpus are split into individual articles. This provides a suitable level of granularity at which to segment the training corpus, and thus it is these articles upon which clustering is performed. Note, however, that while the two corpora are of similar sizes, the articles in the British national corpus tend to be much longer, and hence there are fewer of them (see Table 4.1).

The next decision concerns the number of components into which training data should be partitioned. If too few components are chosen, then it is unlikely that the mixture-based model will be able to distinguish between the full range of topics and styles which it is aiming to model. However, if too many components are chosen, then the memory required to store the model, and the computation required to use it will be increased. Moreover, each individual model will be trained on a more sparse data set, and will hence provide less reliable probability estimates. This effect can be partially overcome, however, by the addition of the full language model (i.e. one trained on the full training

text) as an additional component. For the experiments described in this chapter, a range of models with differing numbers of components were constructed in order to investigate how this factor affected the model's quality.

The articles were clustered according to a straightforward k -means style algorithm. Alternative agglomerative clustering approaches were rejected as requiring excessive computation. The following description of the clustering algorithm assumes that there are A articles which have to be assigned to k clusters. A random order is assigned to the articles in order to prevent any structure within the corpus influencing the resulting partition. The algorithm then proceeds as follows:

- Assign articles 1 to k to clusters 1 to k respectively.
- Assign each of the articles $k + 1$ to A to the cluster to which it is closest.
- Repeat
 - For each article, check if there is a cluster to which it is closer than the cluster to which it is currently assigned, and if there is then move it to the closest cluster.
- Until fewer than T articles are moved on one iteration.

where T is a threshold determining when the algorithm is deemed to have converged.

Ideally, the iterated step would move articles if and only if doing so would reduce some objective function, such as the perplexity of the final mixture-based model. However, computing the value of the final language model's perplexity at each stage of the clustering algorithm would require a prohibitive amount of computation. For this reason, the clustering algorithm does not seek to minimise the final perplexity, but rather to achieve a partition with the desirable property that each article is best represented by the cluster to which it belongs. That is, similar articles will be clustered together.

The metric by which it is decided how close an article is to a cluster is based on the unigram statistics of the article and cluster. Specifically, the distance is the perplexity of a unigram language model trained on the text in the cluster if the test text were the text in the article. This can be easily computed given the unigram counts of the article and the cluster. Let $C_C(w)$ and $C_A(w)$ be the number of occurrences of the word w in the cluster and the article respectively, and C_C and C_A be the total number of words in the cluster and the article. Furthermore, let the article consist of the word string $w_1^{C_A}$. Then,

$$PP = P(w_1^{C_A} \mid \text{cluster})^{-1/C_A} \quad (5.2)$$

$$= \left[\prod_{i=1}^{C_A} P(w_i \mid \text{cluster}) \right]^{-1/C_A} \quad (5.3)$$

$$\approx \left[\prod_{i=1}^{C_A} \left(\frac{C_C(w_i) + \delta}{C_C} \right) \right]^{-1/C_A} \quad (5.4)$$

$$= \frac{1}{C_C} \left[\prod_{i=1}^{C_A} (C_C(w_i) + \delta) \right]^{-1/C_A} \quad (5.5)$$

where $\delta = 1$ is a constant floor value, which is added to all counts in order to ensure non-zero counts in the cluster for all the words in the text (and hence finite perplexities).

Of course, this metric would ideally be based on a trigram language model, to better reflect the models being built. However, the language models need to be changed at so many points in the clustering algorithm, and so many perplexity calculations are required at each iteration, that using trigram models requires a prohibitive amount of computation. Unigram models were used as their parameters can be trivially changed as articles are moved from one cluster to another, and their perplexities are much less demanding to compute.

5.3 Component language model construction

For each of the components, a trigram language model was constructed. Each of the component language models used the same parameters (in terms of the discounting, cutoffs, etc.) as the baseline language models described in Section 4.4.

For both corpora, a full language model was also constructed, again using the same parameters as the models described in Section 4.4.

Models consisting of 10, 20, 30, 40 and 50 components were constructed. These models were evaluated both with and without the full language model being used as an additional component. In addition, each article in the British national corpus has a label indicating which of ten broad topics it belongs to. This results in a manually generated partition into 10 components, which can be compared with the automatically generated partition.

One might expect that a mixture-based model would have many more parameters than the equivalent standard trigram model. Many of the trigrams and bigrams will occur in more than one component, leading to much duplication

within the model, and hence an increase in the number of model parameters. However, one can see in Table 5.1 that this effect is not as severe as might be expected. This is due to the use of cutoffs, which lead to an effect which works against the problem of bigram and trigram duplication. Bigram and trigram cutoffs of one were used for each of the component models. These cutoffs were used for the same reason as for the baseline model, namely that they result in a much smaller model without greatly affecting the model’s performance. When the cutoffs are used for the component models of a mixture-based model, however, the effect on the model size is even greater. For example, a trigram which occurs three times in the full set of training data might end up occurring once in each of three separate components, and hence be excluded from each of the component models when cutoffs are applied. While such an application of cutoffs will lead to a slight reduction in the overall model quality, it does result in models with similar numbers of parameters. This has two advantages. Firstly, the mixture-based models remain small enough that they can easily be stored in memory. Secondly, the comparisons which are made between the models are in some sense “fair”, as they are all of similar sizes.

No. of mixture components	No. of parameters – BNC	No. of parameters – broadcast news
Baseline	11,823,420	12,294,777
10	15,001,932	17,365,271
20	16,110,043	19,120,972
30	16,721,448	20,137,993
40	17,111,975	20,952,138
50	17,361,829	21,568,526

Table 5.1 *The size of mixture-based language models. Based on trigram models with bigram and trigram cutoffs set to 1.*

5.4 The topic homogeneity of the components

Experiments were conducted to investigate how well the automatic clustering algorithm picked out particular topics and styles of language.

For each component language model, a score is assigned to each word in the vocabulary. This score indicates how much the word’s unigram probability according to the component language model differs from that according to the baseline model. A χ^2 score is used to measure the difference between the

unigram probability of a word according to the component language model and the full language model. So, if \mathcal{C} is the component language model, and \mathcal{F} is the full language model, then the χ^2 score is

$$\chi^2(w) = \frac{(\hat{P}_{\mathcal{C}}(w) - \hat{P}_{\mathcal{F}}(w))^2}{\hat{P}_{\mathcal{F}}(w)}. \quad (5.6)$$

In this case, interest is limited to those words whose probability estimate in the component language model is higher than in the full language model.

This score was calculated for each word in the vocabulary for each of the language models which resulted from the automatically generated partition into ten components. The words considered more probable by the component model were sorted according to their χ^2 score. The top twenty words for each component are displayed, with the results for the broadcast news corpus being shown in Table 5.2, and those for the British national corpus being shown in Table 5.3.

These tables show that each of the clusters seems to be associated clearly with a topic or style. For example, in the broadcast news corpus, cluster 7 seems to contain articles about O.J. Simpson, cluster 5 contains information about financial markets, and cluster 2 contains stories about Bill Clinton and Whitewater. Similarly, in the British national corpus partition, cluster 7 seems to contain text concerning the music industry, cluster 9 seems to be concerned with sports, and clusters 1 and 10 both seem to contain the sort of words generally associated with unplanned, spontaneous speech (e.g. MM, ER, COS, OKAY, GONNA).

5.5 Perplexity experiments

The perplexities of the mixture-based models trained on the broadcast news corpus and British national corpus were evaluated on their respective test sets.

The assumption was made that the first third of each article is representative of the article as a whole, and contains sufficient information to allow for the selection of appropriate mixture weights. As such, for both corpora, each of the articles in the test set was split into two parts. The first part consisted of the first third of the article's text and was used to estimate the interpolation weights to be assigned to each of the component models. This was done by computing the probability of each word in the first part of the text according to each component model, and then applying the expectation maximisation (EM)

Cluster 1	Cluster 2	Cluster 3	Cluster 4	Cluster 5
PEACE	CLINTON	WOMEN	I	MARKET
U.	DOLE	CANCER	YEAH	STOCK
YELTSIN	CAMPAIGN	PATIENTS	MOVIE	RATES
UNITED	REPUBLICAN	AIDS	FILM	STOCKS
RUSSIA	PEROT	DISEASE	YOU	COMPANIES
BOSNIA	BUCHANAN	CHILDREN	MUSIC	COMPANY
MILITARY	BUSH	SCHOOL	</s>	BUSINESS
ISRAEL	PRESIDENT	DOCTORS	KNOW	INVESTORS
NATIONS	HE	MEDICAL	MY	DOLLARS
TROOPS	PARTY	DOCTOR	LOVE	PRICES
HAITI	VOTERS	CHILD	SHE	MARKETS
PRESIDENT	BOB	PARENTS	OH	INFLATION
NUCLEAR	WHITEWATER	BREAST	HER	PERCENT
KOREA	CANDIDATE	SCHOOLS	LIKE	DOW
S.	CANDIDATES	DRUG	PLAY	BOND
MINISTER	SENATOR	SEX	WAS	INDUSTRY
NATO	REPUBLICANS	STUDENTS	HOLLYWOOD	FED
STATES	BILL	STUDY	SONG	ECONOMY
IRAQ	FORBES	KIDS	MOVIES	DOLLAR
WAR	WHITE	TREATMENT	ME	SHARES
Cluster 6	Cluster 7	Cluster 8	Cluster 9	Cluster 10
SPACE	SIMPSON	TAX	POLICE	U.
WATER	DEFENSE	BUDGET	TRIAL	BOSNIAN
SHUTTLE	EVIDENCE	REPUBLICANS	CASE	SARAJEVO
ASTRONAUTS	HONOR	HEALTH	F	SERB
MILES	JURY	MEDICARE	GUILTY	SERBS
NASA	PROSECUTION	PLAN	OFFICERS	WAR
HURRICANE	YES	DEFICIT	OKLAHOMA	SOLDIERS
EARTHQUAKE	J.	CUTS	WACO	TROOPS
WEATHER	BLOOD	CONGRESS	VERDICT	N.
RIVER	WITNESS	SPENDING	LAW	ARMY
EARTH	O.	CARE	PRISON	PEACE
TELESCOPE	JUDGE	TAXES	KORESH	REFUGEES
FLOOD	CASE	REFORM	MURDER	BOSNIA
STORM	CORRECT	WELFARE	MCVEIGH	FORCES
SCIENTISTS	SIMPSON'S	BILL	COMPOUND	S.
FISH	TESTIMONY	DEMOCRATS	JUSTICE	ISRAELI
SPECIES	FUHRMAN	PRESIDENT	I.	MILITARY
RAIN	ITO	SENATE	PENALTY	THE
DAMAGE	OBJECTION	CUT	GUN	MOGADISHU
WINDS	DETECTIVE	BALANCED	AGENTS	MUSLIM

Table 5.2 Top twenty words by χ^2 score for each of 10 automatically generated clusters from the broadcast news corpus

Cluster 1	Cluster 2	Cluster 3	Cluster 4	Cluster 5
YEAH	WATER	AWARD	SOVIET	OF
</s>	SPECIES	POINT	NINETEEN	SOCIAL
YOU	FISH	ZERO	MINISTER	IS
I	ARE	PERCENT	GOVERNMENT	EDUCATION
MM	IS	PATIENTS	PRESIDENT	HON.
OH	PLANTS	SOFTWARE	PARTY	NINETEEN
THAT'S	SURFACE	M.	U.	WHICH
GOT	TANK	TWO	THIRTY	ARE
ER	COLOUR	INC	OCT.	LANGUAGE
COS	CAN	R.	JAN.	BE
IT'S	AQUARIUM	UNIX	ELECTIONS	THE
KNOW	KNITTING	C.	THOUSAND	OR
YES	BIRDS	DATA	FOREIGN	SCHOOLS
WELL	BREEDING	B.	MILITARY	SUCH
ERM	FOOD	CORP	NINETY	CURRICULUM
DON'T	EGGS	THREE	S.	IN
GONNA	OR	H.	UN	TEACHERS
ALRIGHT	HOTEL	LIFESPAN	DECEMBER	EXAMPLE
NO	YOUR	REF	DEMOCRATIC	MAY
IT	SOIL	SYSTEMS	FEBRUARY	PUPILS
Cluster 6	Cluster 7	Cluster 8	Cluster 9	Cluster 10
THOUSAND	STUDIO	SHE	YESTERDAY	ER
HUNDRED	TAPED	HER	POUNDS	ERM
CENTURY	MUSIC	HE	DARLINGTON	YOU
OF	VIDEO	I	MR	I
ONE	FILM	</s>	PLAYERS	WE
EIGHT	GUITAR	HIM	CUP	THINK
ART	SPEAKER	HIS	SEASON	IT'S
THE	VOICE	HAD	LAST	YES
KING	READ	SAID	CLUB	THAT
AND	SAYS	WAS	MIDDLESBROUGH	THAT'S
CHURCH	ALBUM	ME	LEAGUE	ACTUALLY
HIS	BAND	YOU	TEAM	WE'VE
WILLIAM	GLOUCESTERSHIRE	MY	GAME	WE'RE
IN	MEDAU	EYES	YEAR	OKAY
MUSEUM	FOLLOWS	LOOKED	MATCH	KNOW
WAS	MALE	DOOR	WIN	MEAN
ROMAN	POP	KNEW	CHAMPIONSHIP	THANK
REIGN	SWINDON	BACK	SAID	DO
SEVEN	WHO	FACE	RUGBY	GOT
POPE	JESUS	HERSELF	MANAGER	WHAT

Table 5.3 Top twenty words by χ^2 score for each of 10 automatically generated clusters from the British national corpus

algorithm (Dempster et al., 1977) to choose linear interpolation weights which maximise the text’s likelihood.

The second part of the article (containing the remaining two thirds) was then used for the perplexity calculation. The probability estimates according to the component models of each word in the second part of the article were computed, and combined by linear interpolation according to the weights computed on the first part of the text. The perplexity was calculated from these. The results are shown in Tables 5.4 and 5.5.

Number of mixture components	Perplexity	
	With full	Without full
Baseline	134.4	
10	121.3	126.6
20	119.3	127.7
30	117.9	128.7
40	117.1	130.0
50	116.7	131.1

Table 5.4 *Perplexities of mixture-based language models. Broadcast news corpus*

Number of mixture components	Perplexity	
	With full	Without full
Baseline	229.3	
10	196.9	204.9
10 (BNC)	199.3	208.7
20	191.8	204.6
30	189.9	205.8
40	188.3	207.1
50	187.3	208.4

Table 5.5 *Perplexities of mixture-based language models. British national corpus*

These results show that the mixture-based models clearly outperform the baseline trigram model for both corpora, regardless of the number of component language models. At best, a perplexity reduction of 13.2% is observed for the broadcast news data, and 18.3% for the British national corpus data. The larger decrease in perplexity observed for the British national corpus is likely to be due to it containing text which is much more varied in terms of topic and

style than the broadcast news corpus, and therefore standing to gain more from language model adaptation. A further possibility is that the difference is due to the larger article length in the British national corpus leading to more text being available to make reliable estimates of the mixture weights.

It is interesting to note that the addition of the full language model component improves performance more in the cases where there are many component language models. This is due to the fact that the full language model compensates for the data sparsity problems which become particularly severe when there are more (and hence less well trained) component language models.

As can be seen from Table 5.5, the manually generated partition into 10 clusters of the British national corpus performs less well than the automatically generated partition into the same number of components. This, along with the information provided by the χ^2 tests in Section 5.4, indicate that the automatic clustering algorithm is achieving its objectives.

5.6 Speech recognition experiments

The results in the previous section show that the mixture-based models have considerably lower perplexities than the baseline trigram model. Speech recognition experiments were carried out in order to investigate whether this would translate into a reduction in word error rate.

The lattices described in Section 4.5 were rescored using a mixture-based model trained on the broadcast news corpus. The model had 30 components, plus the full language model. The choice of the number of components was based on the perplexity experiments. Very little reduction in perplexity was observed as the number of components was increased from 30 to 50. However, the increase in computational demands which results from using more mixture components is significant. The recognition experiments were conducted using 30 components and the full language model as this was viewed as the best compromise between computational demands and model quality.

The mixture weights were chosen separately for each article. This was possible because of the article boundary markers which accompany the data. The weights were computed using the EM algorithm to maximise the likelihood of the text contained in the first-pass transcription of the relevant article.

As can be seen from the results in Table 5.6, the mixture-based model performed slightly less well than the baseline trigram model. Given the reductions in perplexity which were observed using this model, this is a surprising result. There are a number of potential reasons for it; the errors in the first-pass tran-

scription could be affecting the quality of the adaptation, for example. The result also throws some doubt over the usefulness of perplexity as a measure of a language model's quality. These issues are investigated further in Chapter 7.

Model	Word error rate
Baseline	37.9%
Mixture-based model	38.2%

Table 5.6 *Word-error rates from using mixture-based language models. Based on 30 mixture components, using the full language model as an additional component.*

5.7 Summary

This chapter has described the experiments that have been conducted using mixture-based language models. It has been shown that such models lead to considerable reductions in perplexity, but lead to a degradation in speech recognition performance compared to a standard trigram model trained on the same text. The reasons for this apparent discrepancy will be investigated in Chapter 7. The next chapter, however, describes experiments using a different adaptive language modelling technique, namely cache-based language modelling.

Cache-Based Language Models

This chapter presents the work on cache-based language modelling which was carried out for this thesis. It starts by presenting baseline experiments, using standard cache-based models on the same test sets as were used to evaluate the mixture-based models in the previous chapter. Next, experiments to investigate how reoccurrence probabilities vary as the previous word occurrence moves further into the past are described. The cache-based model is changed to reflect the findings, by use of a *decaying history*. Finally, lattice rescoring experiments using cache-based models are presented.

6.1 Introduction

Cache-based language models were introduced in (Kuhn and De Mori, 1990), and were discussed here in Section 2.5.1. The premise is that words occur in “bursts” – words that have occurred recently are more likely to reoccur than a static language model would predict.

In this chapter, experiments using cache-based language models that were conducted for this thesis are presented. The chapter begins by describing experiments to evaluate the perplexity of traditional cache-based models as described in Section 2.5.1. Next, experiments which investigate the manner in which word reoccurrence probabilities vary with their distance from the original occurrence are presented. The findings motivate the development of an extension to the standard cache-based model, in which a word’s contribution to the cache-based probability decreases with its distance from the word being predicted. This is shown to lead to a decrease in perplexity over the standard cache.

The second part of this chapter is concerned with word error rate experi-

ments involving cache-based language models. It is shown that even the language models which showed the largest reduction in perplexity result in little or no reduction in word error rate.

6.2 Perplexity experiments

6.2.1 Regular cache

All of the cache-based models which were used to generate the results in this chapter are based on combining a cache-based component with the baseline trigram language model using linear interpolation:

$$\hat{P}(w_i | w_1^{i-1}) = \lambda \hat{P}_{\text{Cache}}(w_i | w_1^{i-1}) + (1 - \lambda) \hat{P}_{\text{N-gram}}(w_i | w_{i-N+1}^{i-1}). \quad (6.1)$$

The models vary according to how $\hat{P}_{\text{Cache}}(w_i | w_1^{i-1})$ is defined. The regular cache sets a fixed cache-size K , and defines the cache-based probability as

$$\hat{P}_{\text{Cache}}(w_i | w_1^{i-1}) = \frac{1}{K} \sum_{j=i-K}^{i-1} I_{\{w_j=w_i\}}. \quad (6.2)$$

where $I_{\mathcal{E}}$ is an indicator function which takes the value 1 if \mathcal{E} occurred, and 0 otherwise.

In order to deal with the situation where n words of the current document have been seen and $n < K$, this can be more accurately written as

$$\hat{P}_{\text{Cache}}(w_i | w_1^{i-1}) = \frac{1}{\min(n, K)} \sum_{j=i-\min(n, K)}^{i-1} I_{\{w_j=w_i\}}. \quad (6.3)$$

Models of this type were evaluated on both the broadcast news and British national corpora. Two sets of experiments were carried out. In the first, the contents of the cache was “flushed” after each article (so that n in equation (6.3) is reset to zero at each article boundary), so words from the previous article cannot interfere with the current prediction. Experiments in which the cache was not flushed were also conducted, in order to investigate the performance of cache-based models under circumstance where article-boundary information is not available.

The baseline trigram models and test sets which were described in Section 4.4 and used in the mixture-based language modelling experiments in the previous chapter were also used here. Once again, the test sets were split up

such that one third of the test set was used to estimate an appropriate value for λ (the interpolation weight assigned to the cache component using the EM algorithm)¹. The remaining two-thirds was used to calculate the perplexity.

A range of cache sizes was investigated. The results for the broadcast news corpus are shown in Table 6.1 and those for the British national corpus in Table 6.2. The perplexities of the cache-based models are reported, along with the interpolation weights assigned to the cache component (the λ in (equation 6.1)) by the EM algorithm.

Cache size	Perplexity	
	Flushed	Not flushed
Baseline	134.3	
10	131.3 (0.016)	131.3 (0.016)
20	128.7 (0.029)	128.7 (0.029)
50	124.6 (0.050)	124.6 (0.050)
100	122.2 (0.064)	122.3 (0.065)
200	120.5 (0.076)	121.1 (0.077)
500	119.4 (0.086)	121.4 (0.081)
1000	119.0 (0.090)	123.1 (0.080)

Table 6.1 *Perplexities of regular cache-based language models for the broadcast news corpus. Values in parentheses indicate the interpolation weight assigned to the cache component.*

These results show that the basic cache-based model leads to a consistent reduction in perplexity. The best results show a 11% decrease in perplexity for the broadcast news corpus, and a 17% decrease for the British national corpus. The larger reduction observed for the British national corpus is consistent with the similar results observed using mixture-based models. Once again, this could be explained either by the British national corpus being more varied (and hence standing to benefit more from adaptation), or by the longer articles of the British national corpus allowing more reliable cache-based components to be estimated. However, the latter fact is not a factor when smaller cache sizes are used, yet there is still more of a reduction for the British national corpus when very small caches are applied. For example, using a flushed cache of size 20 reduces perplexity by 4.2% for the broadcast news corpus, and by 5.6% for the British national corpus. This implies that it is the more varied nature of the British national corpus which accounts for the larger perplexity reductions.

¹A global value for λ was used across all articles.

Cache size	Perplexity	
	Flushed	Not flushed
Baseline	211.3	
10	205.5 (0.013)	205.5 (0.013)
20	199.4 (0.029)	199.4 (0.029)
50	188.8 (0.062)	188.8 (0.062)
100	181.9 (0.087)	182.0 (0.087)
200	177.6 (0.109)	177.6 (0.109)
500	175.3 (0.134)	175.5 (0.135)
1000	175.5 (0.149)	175.8 (0.150)

Table 6.2 *Perplexities of regular cache-based language models for the British national corpus. Values in parentheses indicate the interpolation weight assigned to the cache component.*

It can be seen that flushing the cache makes very little difference for the British national corpus, but a more appreciable difference for the broadcast news corpus. This is explained simply by the fact that the articles are shorter in the broadcast news corpus. Thus there are more instances where the cache would be flushed, and so the decision as to whether or not to do this will have more of an impact.

6.2.2 Decaying history

The perplexity reductions seen through the use of the regular cache provide evidence for the hypothesis that words do indeed occur in bursts. With this in mind, it seems plausible that the more recent words are more influential in predicting forthcoming words than those in the more distant past. For this reason, experiments involving a *decaying history*² were conducted³.

The manner in which word probabilities vary with distance from the previous occurrence was investigated by examining the broadcast news corpus. Given that the last occurrence of a word w was at w_{i-k} , the manner in which

²This is what was referred to as a *decaying cache* in (Clarkson and Robinson, 1997). However, this nomenclature is somewhat misleading, as the term “cache” would imply that words are removed from it at some point. In fact they continue to affect the cache-based probabilities, but their contributions simply tend towards zero as their distance from the current word increases.

³This work has similarities with work carried out concurrently by Thomas Niesler and described in (Niesler and Woodland, 1997). The similarities and differences are discussed in Section 9.1.1.

the probability that $w_i = w$ varies with k was investigated. That is, we are concerned with the function

$$p_w(k) = P(w_i = w \mid w_{i-k} = w, \sum_{j=1}^{k-1} I_{w_{i-j}=w} = 0) \quad (6.4)$$

Let $s_w(k)$ be the number of times that two occurrences of the word w are separated by exactly k words in the training corpus, a quantity which can easily be computed by examining the training corpus. It then follows that the function $p_w(k)$ can be estimated as follows:

$$\hat{p}_w(k) = \frac{s_w(k)}{\sum_{c \geq k} s_w(c)}. \quad (6.5)$$

In practice, the data sample is too sparse to generate reliable estimates, and so the function $p_w(k)$ is smoothed by averaging its values over a window of length 21 (that is, the central time point plus ten time points on either side):

$$\hat{p}'_w(k) = \frac{1}{21} \sum_{j=k-10}^{k+10} p_w(j). \quad (6.6)$$

Sample probability decay graphs are shown in Figures 6.1 to 6.4. Exponential decay curves of the form $y = ae^{bx} + c$ are shown as a comparison. The values of a , b and c were chosen to minimise the squared error between the curve and the observed data.

These graphs indicate that a word's probability of reoccurring decays with its distance from its previous occurrence, and that the rectangular window used by the regular cache does not accurately model the behaviour of the word reoccurrence probabilities. Indeed, the manner in which the word reoccurrence probabilities decay seems to be approximately exponential.

This phenomena is modelled by dictating that a word's contribution to the cache-based probabilities decays exponentially with increasing distance from the current word. This is done by simply defining

$$\hat{P}_{\text{Cache}}(w_i \mid w_1^{i-1}) = \beta \sum_{j=1}^{i-1} I_{\{w_i=w_j\}} e^{-\alpha(i-j)} \quad (6.7)$$

where β is a constant chosen to ensure that $\sum_{w \in \mathbf{V}} \hat{P}_{\text{Cache}}(w \mid w_1^{i-1}) = 1$.

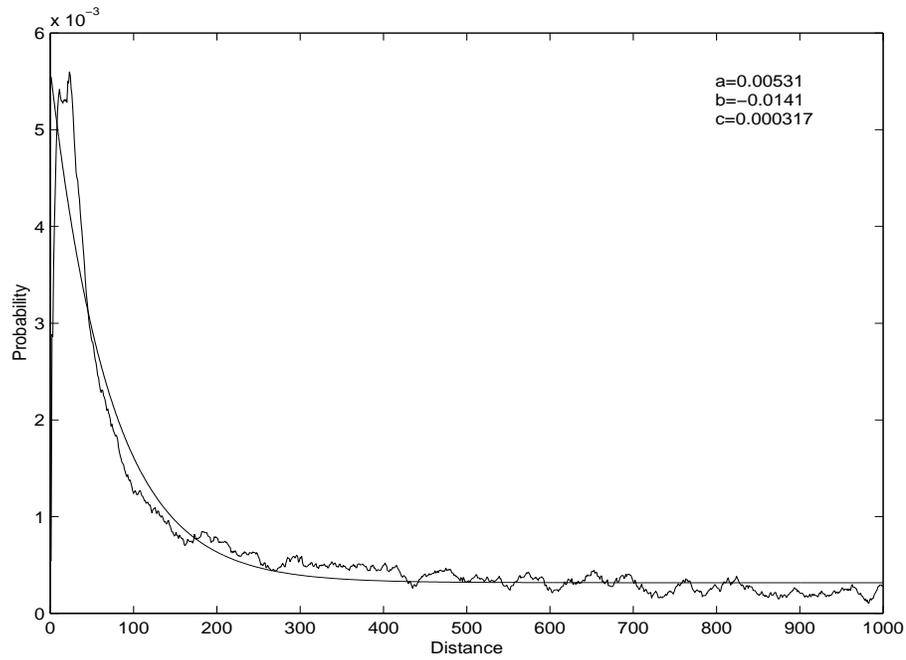


Figure 6.1 The decay of recurrence probability of the word *DOLLAR* in the broadcast news corpus.

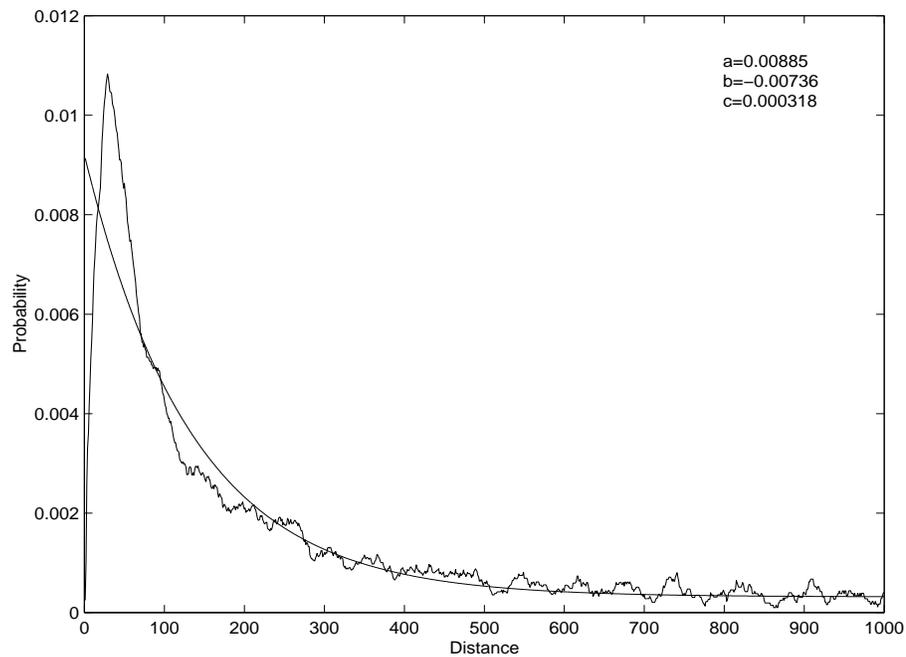


Figure 6.2 The decay of recurrence probability of the word *NUCLEAR* in the broadcast news corpus.

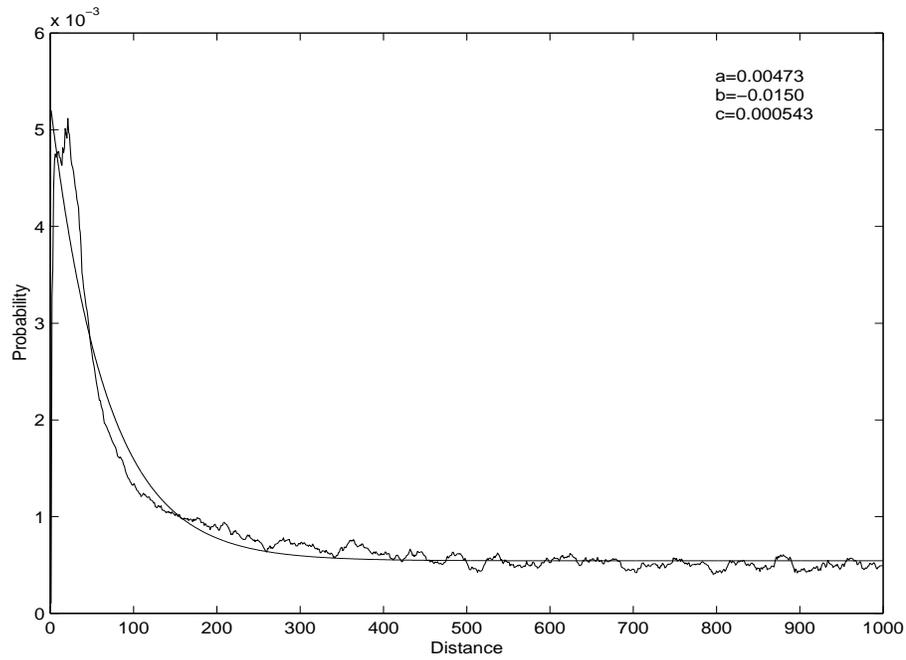


Figure 6.3 The decay of reoccurrence probability of the word *NIGHT* in the broadcast news corpus.

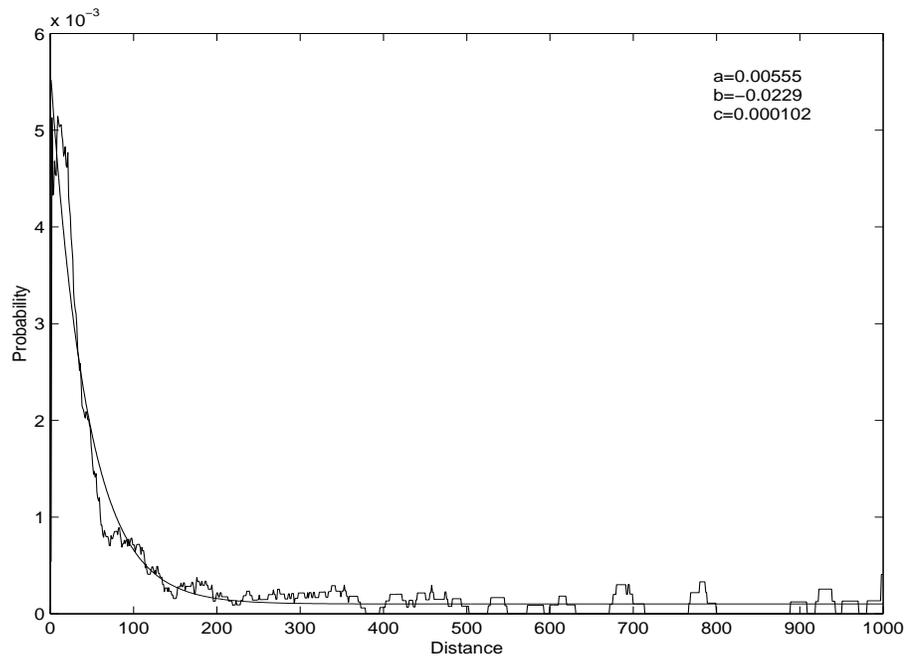


Figure 6.4 The decay of reoccurrence probability of the word *POUND* in the broadcast news corpus.

Note that this approach uses a decay rate which is constant for all words in the vocabulary. However, the graphs in Figures 6.1 to 6.4 indicate that there is a certain amount of variability in decay rate. Despite this, there are good reasons for choosing a constant decay rate. The first is that automatically estimating the decay parameters for each of the words in the vocabulary would require a large quantity of computational time when the model is trained. Secondly, adopting a constant decay rate has the advantage that the normalisation factor β in equation (6.7) is constant. If the decay rate differed for each word in the vocabulary, then β would differ according to the which words had been observed recently, and would thus have to be recalculated each time the cache was updated. This would have a major impact on the computation required to use the model.

One intuitive argument against modelling word recurrence probabilities using exponential decay is that a word's probability is clearly not at a maximum *immediately* after its initial observation. Indeed, it is considered poor use of English to use the same word many times in quick succession. This phenomenon can be observed in the decay graphs in Figures 6.1 to 6.4 – the probability rises to a maximum at a distance of approximately 20 words, and decays approximately exponentially after that. This discrepancy will have a very small effect on the quality of the final model, and it was felt that the simplicity of the exponential decay was not worth sacrificing in order to attempt to model it.

Perplexity experiments were conducted using the exponentially decaying history as a replacement for the standard cache component. The same baseline language model as was used with the regular cache was used here, and the same sections of the test text were used for estimating λ and calculating perplexity. Again, the experiments were performed both with and without flushing the word history at article boundaries, and a range of decay rates were investigated. Figure 6.5 shows the results for the broadcast news corpus, and Figure 6.6 shows the results for the British national corpus.

These results show a further reduction in perplexity, as compared to the results using a standard cache. However, it is worth noting that for the broadcast news corpus, the decaying cache performs only slightly better in the case where the cache is flushed at article boundaries. This is likely to be because the articles are so short that there is not sufficient space in each one for the decay of probabilities to become a major factor. Indeed, with articles so short, it seems that all previous information from the article is approximately equally important.

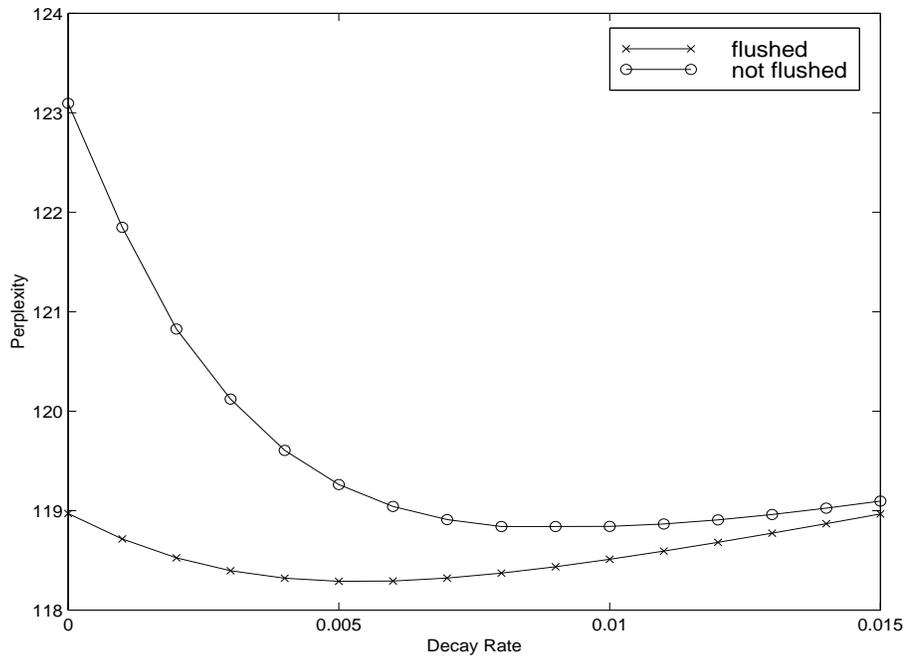


Figure 6.5 *The effect of decay rate on the perplexity of decaying history-based language models. Broadcast news corpus*

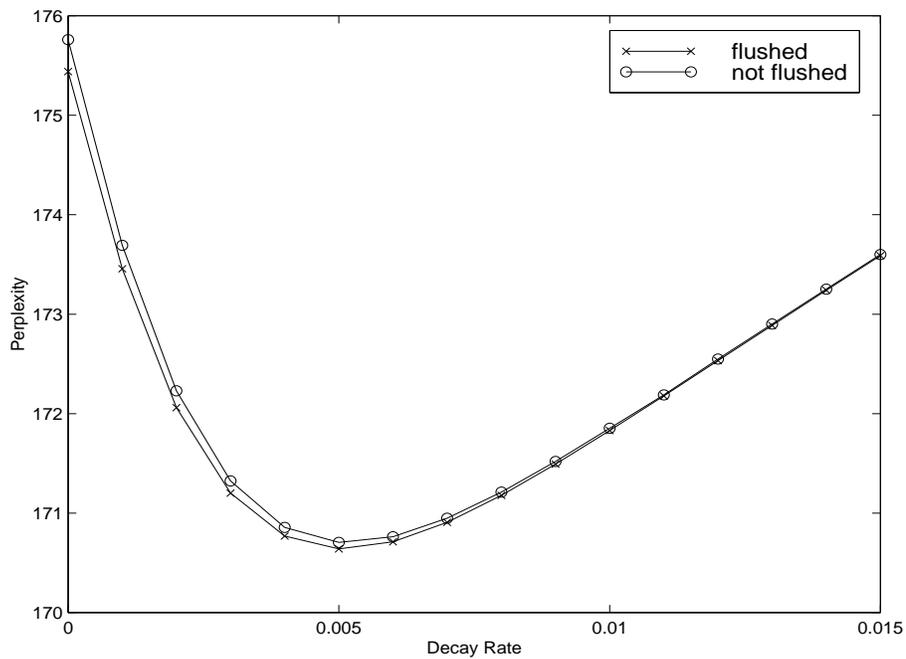


Figure 6.6 *The effect of decay rate on the perplexity of decaying history-based language models. British national corpus*

6.3 Speech recognition experiments

6.3.1 Regular cache

Lattice rescoring experiments were conducted on the broadcast news lattices described in Section 4.5.

As so little gain in perplexity was achieved by applying the decaying history to the broadcast news data, in this case a standard cache is used for the sake of computational efficiency. Furthermore, in the case where the cache was flushed, a reduction in perplexity was consistently observed as the cache size was increased. So for these experiments a cache of unlimited size was maintained⁴, and this was flushed at the end of each article.

The cache-based probabilities are updated at the end of each utterance, and are based on the output of the first-pass decode.

Table 6.3 shows the results of the lattice rescoring experiments using different values of the interpolation parameter λ .

λ	Word error rate
0 (Baseline)	37.9%
0.05	37.9%
0.1	38.0%
0.2	38.6%

Table 6.3 *Word error rates from using cache-based language models.*

These results show that once again, the reductions in perplexity do not translate to an improvement in recognition performance.

6.3.2 Forward-backward cache

The next step was to attempt to include a larger quantity of useful information in the cache. Since the cache is not applied until a first-pass transcription has already been generated, one can use information from the future portion of the article, as well as the past (Woodland et al., 1996). Therefore the cache-based probabilities are computed such that they are equal to the word frequencies of the whole first pass transcription of the current article (with the utterance currently being decoded removed). This will be referred to this as the *forward-backward cache*. This is illustrated in Figure 6.7. The section currently being decoded is not included in the cache since this will bias the language model in

⁴The cache size was greater than the length of the longest article the test text.

favour of the words contained in the initial decode, with the result that there is unlikely to be any difference between the initial decode and the outcome of rescoring the lattice using the cache-based model.

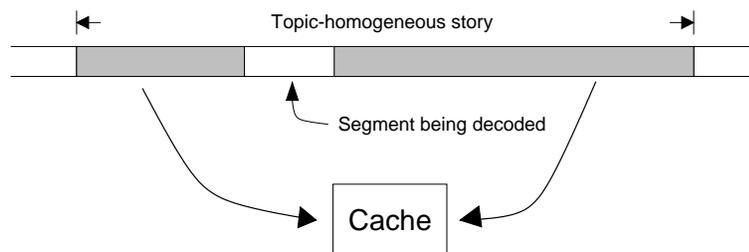


Figure 6.7 *The use of the forward-backward cache*

Again, lattice rescoring experiments were conducted using different values of the interpolation parameter λ . The results are shown in Table 6.4.

λ	Word error rate
0 (Baseline)	37.9%
0.05	37.9%
0.1	38.0%
0.2	38.4%

Table 6.4 *Word error rates from using cache-based language models with a forward-backward cache.*

The extra information in the forward-backward cache leads to little or no improvement over the conventional cache.

6.4 Summary

This chapter has described experiments involving cache-based language models. It has been shown that the standard cache-based model leads to substantial reductions in perplexity, evidence has been provided that word reoccurrence probabilities are better modelled by an exponential decay than by the conventional rectangular window, and it has been shown that modifying the cache-based model to reflect this results in a further reduction in perplexity. However, when applied to lattice-rescoring experiments, the cache-based models have been shown to lead to no reduction in word error rate. These results are consistent with the findings of other researchers. For example, (Woodland

et al., 1996) reports that for the Hub-3 read business news task a very small reduction in word error rate results from the addition of a cache component, and (Woodland et al., 1998) shows no improvement at all when a cache is used for the Hub-4 broadcast news task. The next chapter attempts to shed some light on the reasons for this phenomenon.

Perplexity and Word Error Rate

The previous two chapters have explored adaptive language models. These models have been shown to lead to considerable reductions in perplexity, but these have not translated into reductions in word error rate. This chapter explores some of the reasons for this apparent discrepancy.

Three sets of experiments are conducted. In the first, the extent to which the errorful nature of the first pass transcription is affecting the quality of adaptation is evaluated. Secondly, the adaptation techniques are applied to a baseline language model which is less well adapted to the target domain. Finally, language models with identical perplexities, but different word error rates are created and the areas in which they differ are investigated.

7.1 The effect of an errorful initial transcription

The preceding chapters have explored techniques by which a language model can be adapted to a particular topic or style of discourse. Each of these methods has required a first pass transcription from the recogniser upon which to base the adaptation. This first pass transcription has been generated using a standard trigram model, and has a word error rate of 37.9%. It seems plausible that such a high error rate could hamper the adaptation performed in the recognition experiments, and that this could therefore explain the discrepancy between the perplexity and word error rate results.

This represents a potentially major problem for adaptive language modelling. A good initial transcription will benefit little from language model adaptation (and would probably indicate that the baseline language model was well adapted in the first place), whereas a poor initial transcription will contain so much noise, and so little useful information, that successful adaptation will

prove difficult.

The problem is not insurmountable. One could, for example, apply confidence measures (Eide et al., 1995) and base the adaptation only on the areas of the initial transcription in which one has high confidence. But the first step is to investigate how much the errorful nature of the initial transcription actually affects performance.

In order to do this, *supervised adaptation* experiments were conducted in which the adaptation was based on the reference transcription (that is, the transcription of what was *actually* said), rather than the initial decode. In addition, perplexity experiments were carried out in which the adaptation was based on artificially corrupted text. Finally, experiments were performed in which the perplexity was calculated on the reference transcription, but adaptation was based on the first pass decode.

7.1.1 Supervised adaptation

Recognition experiments were conducted in which the adaptation was based not on the first pass decode, but on the reference transcription. Of course, this information would not normally be available during a standard recognition pass, but the experiment was conducted to investigate the effect on adaptation of the errors in the first pass. For the mixture-based model, each article's mixture weights were calculated based on the reference transcription. Similarly, the cache-based probabilities are estimated by the forward-backward cache (see Section 6.3.2), but are now based on the reference transcription, rather than the initial decode.

Table 7.1 shows the effect of using supervised adaptation as compared to unsupervised adaptation.

Model	Word Error Rate	
	Unsupervised	Supervised
Baseline	37.9%	
Cache ($\lambda = 0.05$)	37.9%	37.6%
Cache ($\lambda = 0.1$)	38.0%	37.7%
Mixture	38.2%	38.0%

Table 7.1 Comparison of supervised and unsupervised adaptation

As would be expected, supervised adaptation does perform better than unsupervised adaptation. However, the difference is not great, which suggests

that the errorful nature of the initial transcription is not the primary reason for the poor performance of these adaptive language models. This isn't to say that it isn't a factor at all; merely that there are other issues which need to be addressed first. There is clearly little to be gained at present from applying techniques such as confidence scoring to select portions of the initial transcription for use in adaptation.

It is interesting to note that even when supervised adaptation is applied, the mixture-based model is still outperformed by the baseline trigram model. This is a curious result, given that a mixture weight of one for the full language model, and of zero for all the other components would yield exactly the baseline model. Thus, the mixture weights that are actually used are clearly sub-optimal in terms of the resulting word error rate, despite the fact that they were chosen to maximise the likelihood of the reference transcription. This is an important point, since it gives a further indication that perplexity may not be the correct measure to optimise when developing language models.

7.1.2 Corrupting the adaptation text

All of the perplexity experiments described in Chapters 5 and 6 based the adaptation on the previously seen, uncorrupted text. The adaptation could therefore be viewed as supervised. By basing the adaptation on an artificially corrupted version of the test text, unsupervised adaptation can be simulated within the framework of perplexity experiments. The perplexity is calculated on the same text, but the text used to calculate the probabilities of the cache-based component and the mixture weights is corrupted.

The cache component was corrupted such that a pre-specified proportion of words were chosen at random from the cache, and replaced with words chosen at random (with a uniform distribution) from the vocabulary.

The text used to calculate the mixture weights for the mixture-based model was corrupted in a similar way. In the previous perplexity experiments using mixture-based models, the mixture weights for each article were calculated using the first third of each article. Here, a pre-specified proportion of words from the first third are chosen at random and replaced by words chosen at random from the vocabulary before the mixture weights are computed.

Different degrees of corruption were investigated for the mixture-based model and for the cache-based model with interpolation parameters of 0.05, 0.1 and 0.2. The results are displayed in Figure 7.1.

These results offer little insight, since for the cache-based model with interpolation parameters of 0.05 and 0.1 and the mixture-based model, the perplex-

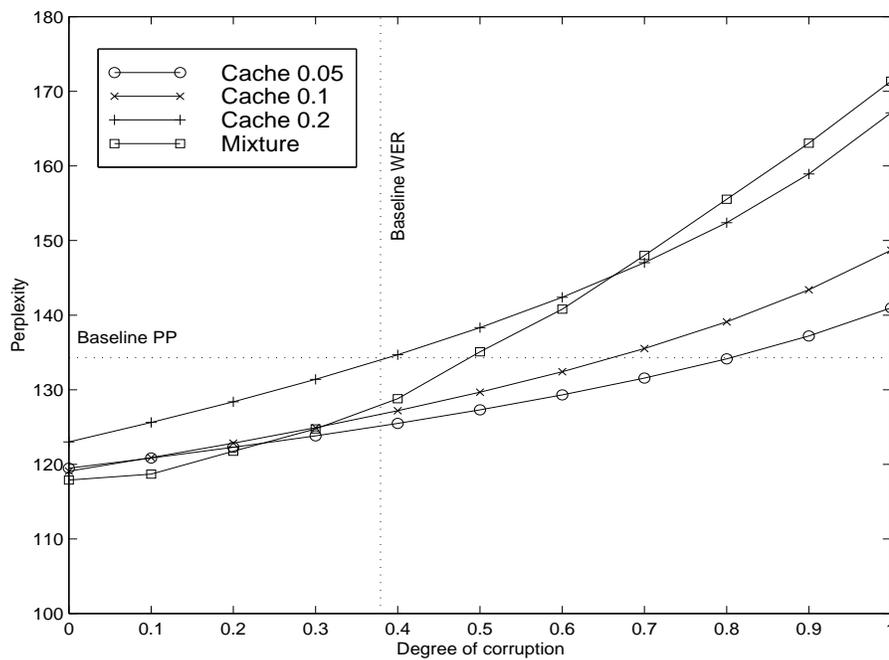


Figure 7.1 *The perplexities of adaptive language models when the adaptation text has been corrupted.*

ity is still substantially lower than the baseline at corruption levels of 37.9%, which is the baseline word error rate. However, they do provide additional evidence that there is little difference between supervised and unsupervised adaptation at the error rate of the first pass transcription.

7.1.3 Perplexity computed on the reference transcription

All of the perplexity experiments described so far in this thesis have been based on the test text. However, if the perplexity calculation is based instead on the reference transcription of the speech one is trying to decode, then both supervised and unsupervised adaptation can be applied. Adaptation based on the first pass decode can be thought of as unsupervised, whereas adaptation based on the reference transcription is supervised.

Table 7.2 shows the effect of both supervised and unsupervised adaptation when cache- and mixture-based models are used.

As might be expected, these results provide similar insight to the results of using corrupted adaptation text. It can be seen that even unsupervised adaptation leads to appreciable reductions in perplexity, and that there is not a great deal of difference between the performance of supervised and unsupervised

Model	Perplexity	
	Unsupervised	Supervised
Baseline	251.4	
Cache ($\lambda = 0.05$)	233.7	224.5
Cache ($\lambda = 0.1$)	234.8	224.4
Mixture	210.4	205.8

Table 7.2 *Comparison of supervised and unsupervised adaptation for perplexity*

adaptation.

7.2 The use of a more general language model

So far, this thesis has described attempts to adapt the baseline broadcast news language model to the individual sub-topics and styles of discourse within the broadcast news task. These adaptation techniques have not had positive results in terms of recognition accuracy. However, the baseline language model is already well adapted to the target domain (it is, after all, trained on transcribed broadcast news shows). Might there be something to be gained from language model adaptation if the baseline language model were not so specific to the target domain?

In order to investigate this, the baseline British national corpus language model was applied to the broadcast news language model test set and lattices. Cache- and mixture-based adaptation was then applied to this model in order to investigate whether a more general language model might benefit more from language model adaptation.

Training text	Adaptation	Perplexity
Broadcast news	None	134.4
BNC	None	277.5
BNC	Cache	216.5
BNC	Mixture	233.6

Table 7.3 *The effect on perplexity of a more general language model*

Table 7.3 shows that the general BNC language model has a much higher perplexity than the broadcast news language model when applied to the broadcast news test set. However, the adaptation techniques reduce the perplexity of

the BNC language model by more than they do for the broadcast news model. Cache-based adaptation reduces the perplexity of the BNC model by 22%, and mixture-based adaptation by 16%.

Training text	Adaptation	Word Error Rate
Broadcast news	None	37.9%
BNC	None	42.9%
BNC	Cache (unsupervised)	42.1%
BNC	Cache (supervised)	41.3%
BNC	Mixture (unsupervised)	42.3%
BNC	Mixture (supervised)	41.8%

Table 7.4 *The effect on word error rate of a more general language model*

Table 7.4 shows the word error rate results from applying cache-based¹ and mixture-based adaptation to the BNC language model. The results show that, in this scenario, the reductions in perplexity do translate to reductions in word error rate. Both forms of adaptation lead to improvements in recognition accuracy, even for unsupervised adaptation. This indicates that the usefulness of these language model adaptation techniques varies according to the baseline language model, and seems to suggest that they are more likely to be of use in situations where the baseline language model is less well suited to the target domain.

7.3 Same perplexity language models

7.3.1 Construction of same perplexity language models

Chapters 5 and 6 showed that it is possible to construct mixture- and cache-based language models which have lower perplexities than the baseline model, but result in the same (or higher) error rate. If one reduces the amount of training data used to train the adaptive language models, their perplexities will be increased. Indeed, if one selects the correct amounts of training data for each language model, it will be possible to generate cache- and mixture-based models that have the same perplexity as the baseline trigram model.

Reducing the amount of training data available to the adaptive models is likely to lead to a degradation in recognition accuracy. Therefore, an under-

¹The interpolation weight assigned to the cache component (i.e. the λ in equation (6.1)) was 0.1, which was chosen to reflect the optimal choice for λ calculated by the EM algorithm during the perplexity experiments.

trained adaptive language model would be expected to result in a higher word error rate than the baseline model. These models will therefore differ in some way that is important in terms of word error rate, despite having identical perplexities. By investigating the manner in which the models differ it is to be hoped that some light might be shed on the discrepancy between word error rate and perplexity.

Cache- and mixture-based language models were generated which had the same perplexity as the baseline trigram language model. This was achieved by using only a fraction of the training data for the adaptive models, while keeping all other factors the same. As such, a randomly chosen set of articles were removed from the training text. It was found that using 37% of the training data for the cache-based model and 42% for the mixture-based model resulted in models with the same perplexity as the baseline model. Lattice rescoring experiments were conducted using these models in order to determine the resulting word error rate². The resulting models are summarised in Table 7.5.

Model	% Training Data	Perplexity	Word Error Rate
Baseline	100	134.4	37.9
Cache-based	37	134.4	39.3
Mixture-based	42	134.4	39.3

Table 7.5 Summary of same-perplexity language models

The differences between the recognition resulting from the use of the baseline model and the two adaptive models are statistically significant at the 1% level according to the matched pairs sentence segments word error test (Gillick and Cox, 1989) described in Appendix A.

7.3.2 Estimating the number of words correct

Recall from Section 2.1 that perplexity is calculated from the probability assigned to some test text w_1^n :

$$PP = P(w_1^n)^{-1/n} = e^{-\frac{1}{n} \sum_{i=1}^n \log[P(w_i | w_1^{i-1})]}. \quad (7.1)$$

²As before, the EM algorithm was used to optimal value for the interpolation weight assigned to the cache component for the perplexity experiments. The resulting value was 0.09. This value was then used as the interpolation weight for the cache component in the lattice rescoring experiments.

The value of perplexity is therefore based on the mean log probability of the words in the test set.

Consider a function $f_{\mathcal{M}}(x)$ which indicates the normalised frequency with which words are assigned a log probability of x by the language model \mathcal{M} . Thus $\int_{-\infty}^0 f_{\mathcal{M}}(x)dx = 1$. The value μ of the mean log probability of the words in the test text can be computed given the values of $f_{\mathcal{M}}(x)$:

$$\mu = \int_{-\infty}^0 x f_{\mathcal{M}}(x) dx. \quad (7.2)$$

It therefore contains at least as much useful information as the value of perplexity, and possibly somewhat more.

Consider also a function $g(x)$ which indicates the probability that a word with language model log probability x will be recognised correctly. $f_{\mathcal{M}}(x)$ and $g(x)$ can be combined to generate an estimate of the expected number of words correct (see Section 2.1)³:

$$E_{\mathcal{M}}(\text{Words correct}) = \int_{-\infty}^0 f_{\mathcal{M}}(x)g(x)dx \quad (7.3)$$

and since one would expect word error rate to be strongly correlated with the proportion of words correct, this is potentially a more useful predictor of recognition performance than perplexity.

Note that this assumes that the function $g(x)$ is constant across all language models. This assumption is necessary, as $g(x)$ can only be estimated based on knowledge of whether words with particular language model probabilities were correctly recognised. Generating this information requires a recognition pass, so if it were necessary for every language model under consideration, one could simply evaluate the word error rate directly, and there would be no need to use techniques to estimate its value.

7.3.3 Estimating the functions f and g

The function $f_{\mathcal{M}}(x)$ was estimated by partitioning the probability range into 100 bins which are spaced equally in the log domain. For each language model, the number of words in the test text which have language model probabilities in each bin is computed. The function $f(x)$ was estimated for the baseline trigram model, as well as for the same-perplexity cache- and mixture-based

³This technique was first investigated by (Chen et al., 1998), and was used to derive M -ref, a measure of language model quality.

models. Figures 7.2 and 7.3 show the resulting functions for the cache- and mixture-based models compared with the baseline trigram model.

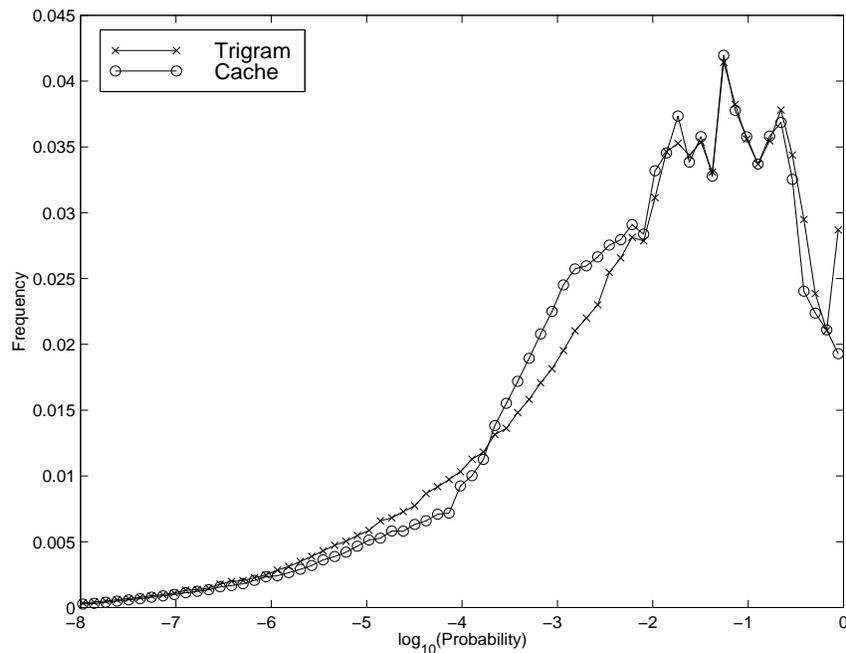


Figure 7.2 *Probability distribution graph. Comparison of $f_{\text{trigram}}(x)$ and $f_{\text{cache}}(x)$*

The function $g(x)$ was computed individually for each of the language models. The transcription generated using each language model was aligned with the reference transcription using the same dynamic programming algorithm as was used for word error rate scoring, and hence each word in the reference transcription was labelled according to whether it was correctly recognised when each of the language models were used. The probability range was split into 30 equally log spaced bins⁴, and the values of $g(x)$ were then estimated at each of the bin centres according to

$$g(x) = \frac{\text{\#words with LM probs in bin that were correctly recognised}}{\text{\#words with LM probs in bin}} \quad (7.4)$$

The resulting functions are displayed in Figures 7.4 and 7.5. Figure 7.4 shows the comparison of $g_{\text{trigram}}(x)$ with $g_{\text{cache}}(x)$, and Figure 7.5 compares $g_{\text{trigram}}(x)$ and $g_{\text{mixture}}(x)$.

⁴Fewer bins were used than in the computation of $f(x)$ since information from a recognition pass was required, as opposed to information from a large, text-only test set, and therefore the data was more sparse.

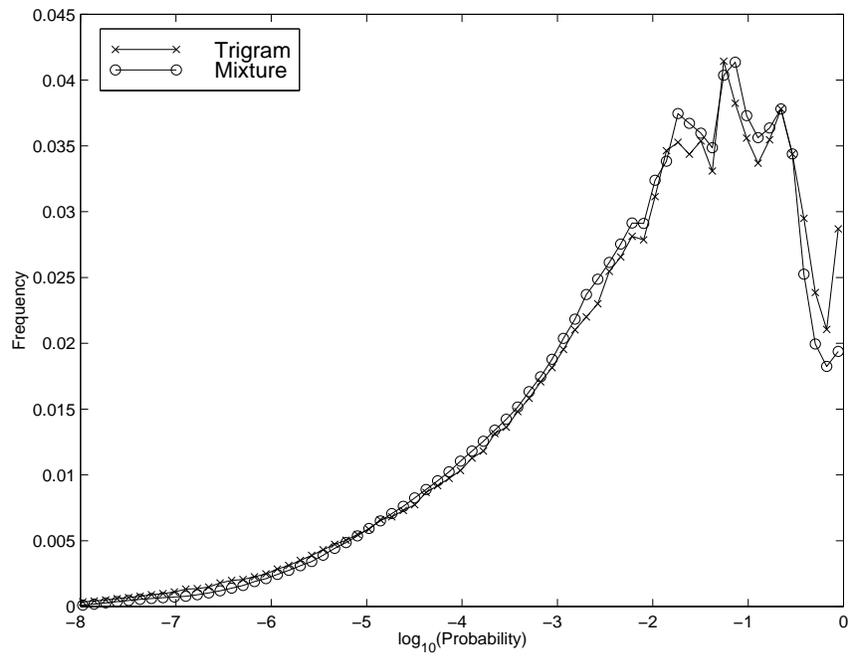


Figure 7.3 Probability distribution graph. Comparison of $f_{\text{trigram}}(x)$ and $f_{\text{mixture}}(x)$

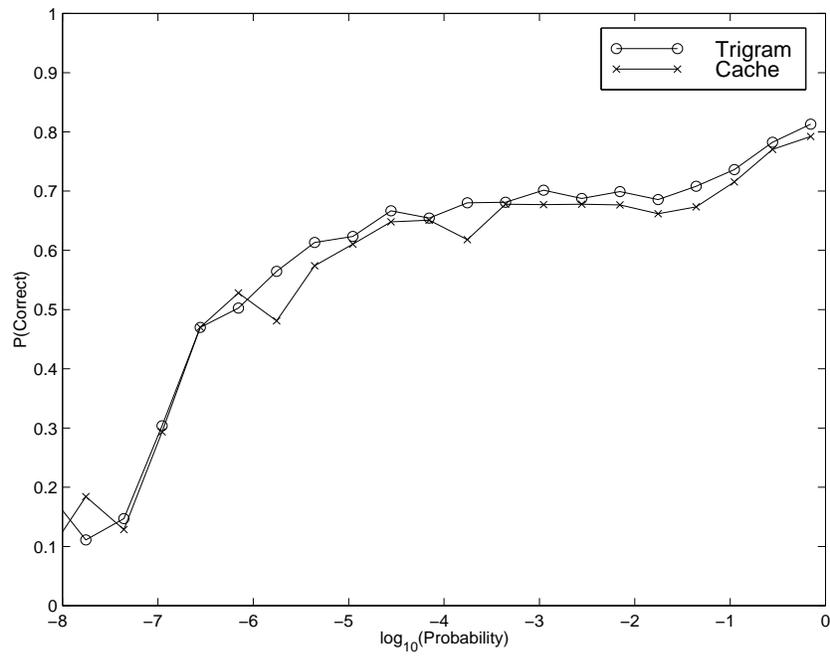


Figure 7.4 Comparison of $g_{\text{trigram}}(x)$ and $g_{\text{cache}}(x)$.

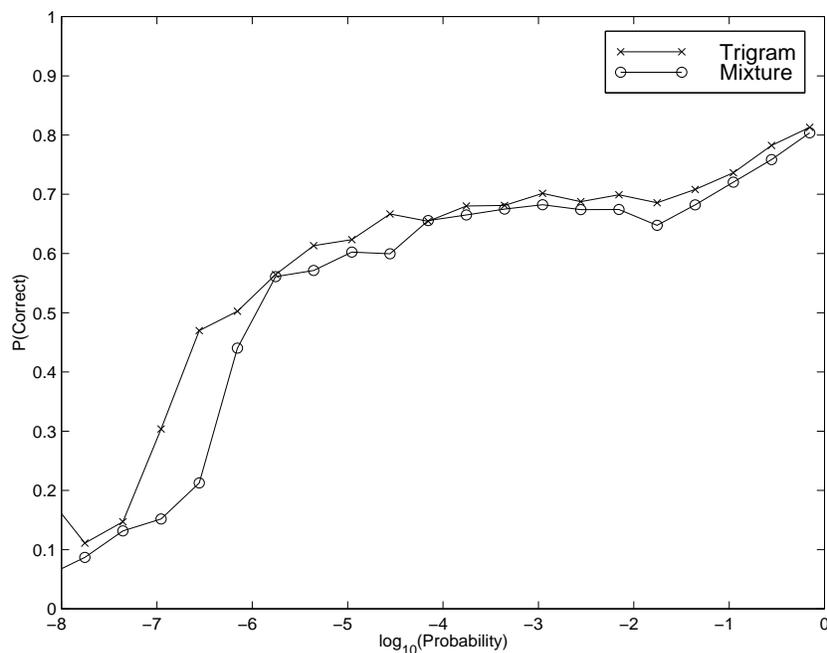


Figure 7.5 Comparison of $g_{\text{trigram}}(x)$ and $g_{\text{mixture}}(x)$.

7.3.4 Results

The estimates for the values of the $f(x)$ and $g(x)$ were used to generate estimates for the number of correct words according to the following approximation⁵:

$$E(\text{Proportion of words correct}) = \int_{-\infty}^0 f(x)g(x)dx \approx \sum_{x \in \text{Bin centres}} f(x)g(x). \quad (7.5)$$

The estimated proportions of words correct were generated in two ways. Firstly, the estimate was based on the appropriate version of $g(x)$, in order to generate the most accurate value. That is, the expected proportion of words correct using the model \mathcal{M} was calculated using $\sum f_{\mathcal{M}}(x)g_{\mathcal{M}}(x)$. However, since an estimate for $g(x)$ for each language model will not be available in practice, an estimate was also generated using $g_{\text{trigram}}(x)$. So, for the model \mathcal{M} , the expected proportion of words correct was $\sum f_{\mathcal{M}}(x)g_{\text{trigram}}(x)$. The results are shown in Table 7.6.

⁵The “bin centres” referred to are the 100 bin centres used in the estimation of $f_{\mathcal{M}}(x)$. The values for $g(x)$ were taken then from the bin with the closest value.

Model	Actual words correct	$\sum f_{\mathcal{M}}(x)g_{\mathcal{M}}(x)$	$\sum f_{\mathcal{M}}(x)g_{\text{trigram}}(x)$
Baseline	69.4%	70.6%	70.6%
Cache	67.4%	68.4%	70.6%
Mixture	67.2%	68.4%	70.7%

Table 7.6 *Estimates of proportion of words correct*

The discrepancy between the actual word error rate, and the values of $\sum f_{\mathcal{M}}(x)g_{\mathcal{M}}(x)$ is due to the fact that $f_{\mathcal{M}}(x)$ is estimated based on the test text, rather than the reference transcript. The language models have a lower perplexity with respect to the test text than they do with respect to the reference transcription, and this leads to a high estimate for words correct. If the reference transcript were used to generate estimates for the values of $f_{\mathcal{M}}(x)$, then the actual word error rate would be identical to $\sum f_{\mathcal{M}}(x)g_{\mathcal{M}}(x)$.

7.3.5 Discussion

The first observation to make from Figures 7.2 to 7.5 is that it is not the case that $g(x)$ is constant across all the models. This suggests that it is unlikely to be possible to predict a model's effect on word error rate accurately from its probability distribution curve alone. Indeed, the values of $\sum f_{\mathcal{M}}(x)g_{\text{trigram}}(x)$ in Table 7.6 show that the expected values of proportion of words correct generated by $g_{\text{trigram}}(x)$ do not show any difference between the three same-perplexity language models.

This phenomena is shown even more strongly by Figure 7.3, which shows that the probability distribution curve for the mixture-based model is almost identical to that of the baseline trigram model. These models result in significantly different word error rates, yet there is not sufficient information in the probabilities of the words in the test text to distinguish between them. It is clear then, that the information needed to discriminate between these models is not contained in the probabilities of the words which actually occur in the test text. It therefore seems likely that the information needed to distinguish between the models is contained in the way in which the remaining probability mass is distributed over the *alternative* words, which will compete with the correct word in the decoder of a speech recogniser. It is this observation which motivates the work in the next chapter.

7.4 Summary

This chapter has explored the discrepancy between word error rate and perplexity which had been observed in the previous chapters. Supervised adaptation experiments showed that the errorful nature of the initial decode is not a primary reason for this discrepancy. It was further shown that, when applied to a baseline language model less well suited to the target domain, the language model adaptation techniques reduced word error rate as well as perplexity. Finally, language models with the same perplexity, but different word error rates were developed and investigated. This led to the conclusion that the probabilities of the words in the test text are not enough in themselves to allow accurate prediction of a language model's effect on word error rate. In order to better predict a language model's effect on word error rate, it seems that it is necessary also to consider the way in which the remaining probability mass is distributed over the alternative words. The next chapter develops this idea further.

Alternative Language Model Evaluation Schemes

The previous chapter investigated some of the reasons for the apparent discrepancy between the perplexity of the adaptive language models described in this thesis and their effect on the word error rate of a recognition system. It concluded by demonstrating that merely considering the probability of the words in the test text is not sufficient to distinguish between models which result in different word error rates. This led to the conclusion that one must also consider the probabilities of alternative words which will compete with the correct words during the recognition process. This chapter investigates measures which incorporate such information, and which correlate better with word error rate than perplexity does.

The chapter starts by investigating the means by which one can calculate the entire probability distribution given a word history, without having to look up the probability of each word in the vocabulary individually. Next, the correlation coefficients used to evaluate how well the new measures correlate with word error rate are presented. The new measures are evaluated on a test set of fifty language models, and these are briefly described. A set of proposed new measures of language model quality are then described and evaluated. It is shown that the useful information from some of these measures is somewhat orthogonal, and means of combining this information are investigated. The resulting measures are shown to be considerably better correlated with word error rate than perplexity is. Finally, it is shown how the information from the new measures can lead to the selection of more appropriate interpolation weights for the mixture-based language models described in Section 5. Such interpolation weights lead to a small, but consistent and statistically significant improvement in word error rate as compared to the original maximum likelihood weights.

8.1 Introduction

8.1.1 The use of the whole distribution

Throughout this thesis, language models have been evaluated according to their perplexity. At each point in the test text, the computation of perplexity considers only the probability of the next word in the text. The language model evaluation schemes explored in this chapter, however, are based on the probability distribution over the whole vocabulary. That is, if the test text is w_1^n , then perplexity is based on the values of $\hat{P}(w_i | w_1^{i-1})$, and the new measures will be based on the values of $\hat{P}(w | w_1^{i-1})$ for all $w \in \mathbf{V}$. The word w_i which actually follows the word history w_1^{i-1} in the test text will be referred to as the *target word*.

At first glance, it might seem that computing $\hat{P}(w | w_1^{i-1})$ for all words in the vocabulary will require more computation by a factor of V (where V is the number of words in the vocabulary) than simply computing one probability, since V language model probabilities need to be calculated. However, there are some short-cuts which can be applied which mean that this is not the case.

Consider the case of a trigram model, where one is attempting to find the values of $\hat{P}(w | w_1^2)$ for all w . The language models used in this research were created using the CMU-Cambridge Statistical Language Modelling Toolkit (see Chapter 3 or (Clarkson and Rosenfeld, 1997)). In these models the information is stored in a tree structure, with the unigram information at the node and the trigram information at the leaves. Therefore, one can find the position of w_1^2 in the bigram layer of the tree, and from there look up the probabilities of all the words w such that the trigram (w_1, w_2, w) exists in the language model. One can then find the position of w_2 in the unigram layer, and look up the probabilities of all words w such that the trigram (w_1, w_2, w) does not exist in the language model, but (w_2, w) does. Finally, the probabilities of the w which have not yet been computed can simply be looked up from the unigram layer.

While this process is considerably more computationally expensive than looking up the probability of just the target word, it is much more efficient than simply performing V language model look-ups. In practice, for the software generated for this research, it requires approximately 500 times more computational time to compute the whole distribution for a 65,000 word vocabulary than it does to compute one language model probability.

8.1.2 Measures of correlation

The extent of the correlation between the new evaluation schemes and word error rate will be evaluated using three correlation coefficients; the Pearson product-moment correlation coefficient r , the Spearman rank-order correlation coefficient r_s , and the Kendall rank-order correlation coefficient T . A brief discussion of these three measures follows, and more detailed information can be found in many standard statistics books (see, for example (Berenson et al., 1988) for the Pearson product-moment correlation coefficient and (Siegel and Castellan, 1988) for the Spearman and Kendall rank-order coefficients).

Note that the values of all three correlation coefficients will fall in the range $[-1, 1]$. Large positive or negative values indicate a strong positive or negative correlation; values near zero indicate little correlation.

8.1.2.1 The Pearson product-moment correlation coefficient r

The Pearson product-moment correlation coefficient measures the degree of *linear* correlation between two random variables X and Y . If the variables X and Y take the values x_1, x_2, \dots, x_n and y_1, y_2, \dots, y_n respectively, then the Pearson product-moment correlation coefficient r is computed as

$$r = \frac{\text{Cov}(X, Y)}{\sqrt{\text{Var}(X)\text{Var}(Y)}} \quad (8.1)$$

$$= \frac{(\sum_{i=1}^n x_i y_i) - \frac{(\sum_{i=1}^n x_i)(\sum_{i=1}^n y_i)}{n}}{\sqrt{\sum_{i=1}^n x_i^2 - \frac{(\sum_{i=1}^n x_i)^2}{n}} \sqrt{\sum_{i=1}^n y_i^2 - \frac{(\sum_{i=1}^n y_i)^2}{n}}} \quad (8.2)$$

The key disadvantage of this measure is that it assumes that any correlation between the X and Y is linear. This is certainly not the case with all the language model evaluation measures which will be considered in this section. For example, any correlation between perplexity and word error rate is not linear, as will seen in Section 8.3.1.

8.1.2.2 The Spearman rank-order correlation coefficient r_s

The Spearman rank-order correlation coefficient r_s is based not on the actual values of random variables, but on the ranks of the observations. Thus, if X takes the values x_1, x_2, \dots, x_n , then the lowest observed value the value 1, and the highest takes the value n . The correlation coefficient is then simply the Pearson product-moment correlation coefficient computed on these ranks.

This has the advantage that it makes no assumptions about the nature of the correlation between the two variables. Its disadvantage is that by considering only the ranks of the observations, the distance between them is disregarded.

Thus if two language models receive very similar scores from an evaluation scheme such as perplexity, and lead to very similar word error rates, this will not be detected – only the ordering of the observations is important.

8.1.2.3 The Kendall rank-order correlation coefficient T

The Kendall rank-order correlation coefficient T is also based on the ranks of the observations. It measures the number of pairs that have the same order in the two distributions. So, for each pair (x_i, y_i) and (x_j, y_j) , we say that X and Y agree if $(x_i > y_i \text{ and } x_j > y_j)$ or $(x_i < y_i \text{ and } x_j < y_j)$. The correlation coefficient is then

$$T = \frac{\# \text{agreements} - \# \text{disagreements}}{\text{Total number of pairs}} \quad (8.3)$$

Since it is based on the ranks of the observations, the Kendall rank-order correlation coefficient has much the same set of advantages and disadvantages as the Spearman rank-order correlation coefficient.

8.2 Language model test set

In order to investigate the correlation between word error rate and new language model evaluation measures, it is clearly necessary to have a large set of language models upon which to base the experiments.

A set of 50 language models was constructed. These models comprise bigram, trigram, mixture- and cache-based models, which have been trained on either the broadcast news corpus or British national corpus. Different quantities of the training corpora were used to train each language model, and various cutoffs were applied. The broadcast news lattices were rescored using each language model and the resulting word error rate and proportion of words correct were computed for each. The set of models is summarised in Table 8.1. The table indicates whether the broadcast news (bn) or British national corpus (BNC) was used to train the model, the type of language model (either bigram or trigram), the type of adaptation used (where $C(x)$ represents cache-based adaptation with an interpolation weight of x for the cache component), the proportion of the training data used to train the model, the cutoffs applied, the word error rate and words correct. The final three models were based on a two component mixture model with one component trained on the broadcast news corpus, and the other on the British national corpus. The interpolation weights

were fixed, and not intended to reflect the target domain. The adaptation argument $M(x)$ indicates that the mixture weight assigned to the broadcast news component was x .

Some interesting points are brought to light by Table 8.1 that are worth mentioning in passing. By comparing the word error rates of models 38 and 46, it can be seen that adding a cache component to a bigram language model leads to a reduction in word error rate for models trained on the broadcast news corpus. Similarly, comparing models 6 and 31 reveals that adding a cache component to the broadcast news model with bigram and trigram cutoffs of 20 results in a reduction in word error rate and comparing models 9 and 26 reveals that adding a cache component to a trigram model trained on only a small fraction of the broadcast news data also yields a reduction in word error rate. These results are all in contrast to the results of adding a cache component to the baseline broadcast news model, and show that a cache component is useful in the case when the baseline language model is not of such high quality.

In Section 7.3 experiments were carried out which predicted the proportion of words correct, with the premise that this would be well correlated with word error rate. The set of language models summarised in Table 8.1 allows the correlation between word error rate and the proportion of words correct to be computed directly. The correlation was computed according to the three correlation measures described in Section 8.1.2. The results are shown in Table 8.2. These results demonstrate that there is a very strong linear correlation between the proportion of words correct and word error rate. Therefore, knowledge of the proportion of words correct allows one to predict word error rate with high accuracy.¹

The perplexity results reported in previous chapters were calculated with respect to the language model test text². However, the language model evaluation schemes which will be described in the next section are evaluated with respect to the reference transcription of the broadcast news shows upon which the word error rate scores are based. There are two reasons for this. The most important reason is the potential mismatch between the language model test text and the recognition task. When perplexity is calculated using the language

¹The strong correlation between the proportion of words correct and word error rate is not surprising, since the only way in which the measures differ is in the inclusion of insertion errors. Since there are generally far fewer insertion errors than substitution errors, this difference will be small compared to the overall values, and hence the correlation will be strong. However, the values of these correlation coefficients allow one to evaluate how accurately one can expect to predict word error rate given the proportion of words correct.

²The only exception to this being the experiments described in Section 7.1.3.

Model	Training Corpus	Type	Adaptation	Fraction of Training Data	Cutoffs	WER	Words Correct	Perplexity
1	bn	3-gram	None	1.0	0,0	38.0	69.3	250.1
2	bn	3-gram	None	1.0	1,1	37.9	69.3	251.0
3	bn	3-gram	None	1.0	2,2	38.2	69.0	256.3
4	bn	3-gram	None	1.0	5,5	39.2	68.2	274.1
5	bn	3-gram	None	1.0	10,10	40.0	67.4	295.6
6	bn	3-gram	None	1.0	20,20	40.9	66.4	321.9
7	bn	3-gram	None	1.0	50,50	41.9	65.3	368.9
8	bn	3-gram	None	1.0	100,100	43.1	64.1	413.0
9	bn	3-gram	None	0.001	1,1	52.0	54.8	954.3
10	bn	3-gram	None	0.01	1,1	45.7	61.5	515.9
11	bn	3-gram	None	0.1	1,1	40.9	66.3	331.7
12	bn	3-gram	None	0.25	1,1	39.5	67.6	292.3
13	bn	3-gram	None	0.5	1,1	39.0	68.3	267.7
14	BNC	3-gram	None	1.0	0,0	43.5	64.2	382.4
15	BNC	3-gram	None	1.0	1,1	42.8	64.7	379.9
16	BNC	3-gram	None	1.0	2,2	43.1	64.5	390.6
17	BNC	3-gram	None	1.0	5,5	43.5	64.0	418.1
18	BNC	3-gram	None	1.0	10,10	43.9	63.4	445.7
19	BNC	3-gram	None	1.0	20,20	44.7	62.5	481.7
20	BNC	3-gram	None	1.0	50,50	45.8	61.3	543.9
21	BNC	3-gram	None	1.0	100,100	47.2	59.8	606.0
22	BNC	3-gram	None	0.01	1,1	50.6	56.4	829.3
23	BNC	3-gram	None	0.1	1,1	46.2	61.4	532.2
24	BNC	3-gram	None	0.25	1,1	44.5	63.1	457.7
25	BNC	3-gram	None	0.5	1,1	43.7	63.9	418.5
26	bn	3-gram	C(0.1)	0.001	1,1	50.6	56.1	705.2
27	bn	3-gram	C(0.05)	1	1,1	37.9	68.9	214.4
28	bn	3-gram	C(0.1)	1	1,1	38.0	68.7	211.0
29	bn	3-gram	C(0.25)	1	1,1	38.8	67.8	219.0
30	bn	3-gram	C(0.5)	1	1,1	40.6	66.0	266.1
31	bn	3-gram	C(0.1)	1	20,20	40.5	66.3	266.7
32	bn	3-gram	Mix	1	1,1	38.2	68.2	211.5
33	bn	3-gram	Mix	0.42	1,1	39.3	67.2	240.1
34	BNC	3-gram	Mix	1	1,1	41.8	64.3	329.3
35	BNC	3-gram	C(0.1)	1	20,20	43.8	62.9	376.0
36	BNC	3-gram	C(0.1)	1	1,1	42.0	65.0	301.6
37	BNC	3-gram	C(0.1)	0.01	1,1	49.1	57.5	605.1
38	bn	2-gram	None	1	1	41.7	64.8	335.9
39	bn	2-gram	None	1	5	42.2	64.4	353.1
40	bn	2-gram	None	0.01	1	47.1	59.9	554.4
41	bn	2-gram	None	0.5	1	42.0	64.5	346.3
42	BNC	2-gram	None	1	1	45.2	61.5	470.8
43	BNC	2-gram	None	1	5	45.7	61.1	497.3
44	BNC	2-gram	None	0.01	1	50.7	56.1	841.7
45	BNC	2-gram	None	0.5	1	45.8	61.0	496.9
46	bn	2-gram	C(0.1)	1	1	41.4	64.5	282.3
47	BNC	2-gram	C(0.1)	1	1	44.4	61.7	375.0
48	bn+BNC	3-gram	M(0.25)	1	1,1	38.9	67.7	247.1
49	bn+BNC	3-gram	M(0.5)	1	1,1	38.1	68.4	224.4
50	bn+BNC	3-gram	M(0.75)	1	1,1	38.0	68.6	217.9

Table 8.1 Summary of language models used to investigate new language model evaluation schemes

	r	r_s	T
Words correct	-0.994	-0.988	-0.918

Table 8.2 *Correlation of words correct with word error rate*

model test text, there is an implicit assumption that this text is representative of the speech which one is attempting to recognise. Due to the nature of the language model test set in this case, this assumption is reasonable. However, one can avoid the need to make it at all by basing the perplexity calculation on the reference transcription. The second reason is a pragmatic one. Since the evaluation of measures based on the whole distribution requires approximately 500 times more computational time than the calculation of perplexity, the opportunity to evaluate them based on a test set of 22,000 words (as opposed to the 18 million word test text) is appealing.

8.3 Proposed language model evaluation measures

8.3.1 Perplexity

Perplexity has been used as a method of evaluating language models throughout this thesis, and in this chapter it serves as the baseline measure. In this section, the correlation between word error rate and perplexity on the test set of fifty language models will be evaluated. However, as a first step, the nature of the relationship between word error rate and perplexity will be investigated. Furthermore, the extent to which the observations of the previous sections continue to hold when perplexity is evaluated on the reference transcription rather than the language model test text will be explored.

A commonly used rule-of-thumb among language modelling researchers is that perplexity and word error rate are related such that word error rate is a linear function of the square root (or cube root) of perplexity. In order to investigate the validity of this heuristic, the following experiment was conducted.

It is hypothesised that word error rate w and perplexity p are related according to the following law:

$$w = ap^r \quad (8.4)$$

where a and r are constants to be determined. So therefore

$$\log w = \log a + r \log p \quad (8.5)$$

and thus if $\log w$ is plotted against $\log p$ one would expect to see a graph that is approximately linear.

A scatter plot of $\log w$ against $\log p$ is shown in Figure 8.1, with the best-fit straight line (i.e. that which minimised the mean squared error between the line and the points) also shown. The gradient and y -intercept of this line provide estimates for the parameters r and a respectively. In this case the values were $r = 0.22^3$ and $a = 2.45$. According to these values, one would expect that the decreases in perplexity of around 15% which were observed using mixture- and cache-based models in Chapters 5 and 6 would lead to a 3.5% decrease in word error rate. Figure 8.2 shows the same graph mapped back from the log scale.

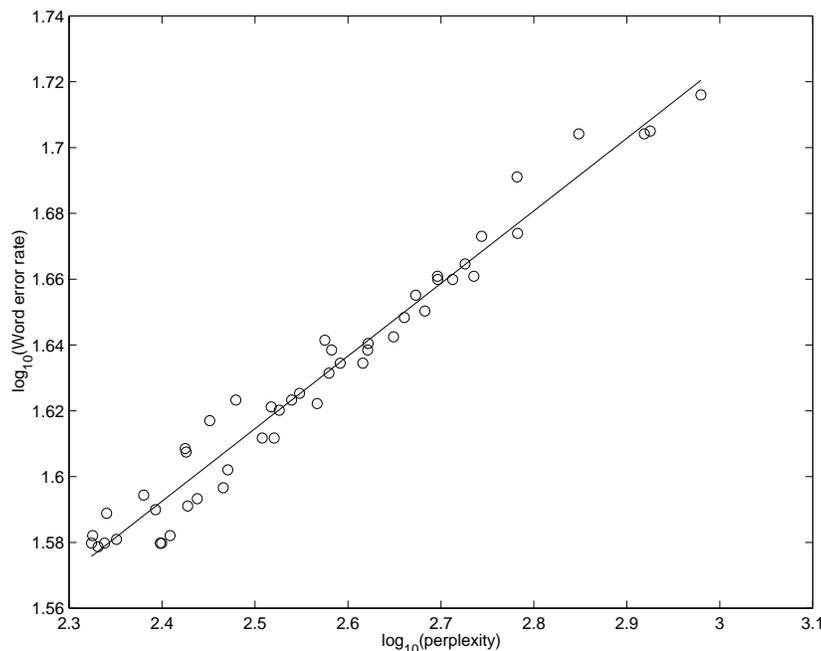


Figure 8.1 A plot of $\log(\text{word error rate})$ against the corresponding $\log(\text{perplexity})$, with the best fit line satisfying $\log(w) = a + r \log(p)$ shown as a comparison.

Knowing the manner in which we expect word error rate to vary with perplexity allows us to use the Pearson product-moment correlation coefficient in a more principled manner, since we can transform the variables so that they are approximately *linearly* correlated. Thus, while the Pearson product-moment correlation coefficient is not the ideal method to use to measure the correlation between word error rate and perplexity, it is appropriate for computing the

³Since the use of lattices here reduces the potential improvement or degradation in recognition accuracy, the value of r may have been higher had the language models been incorporated into the initial decode.

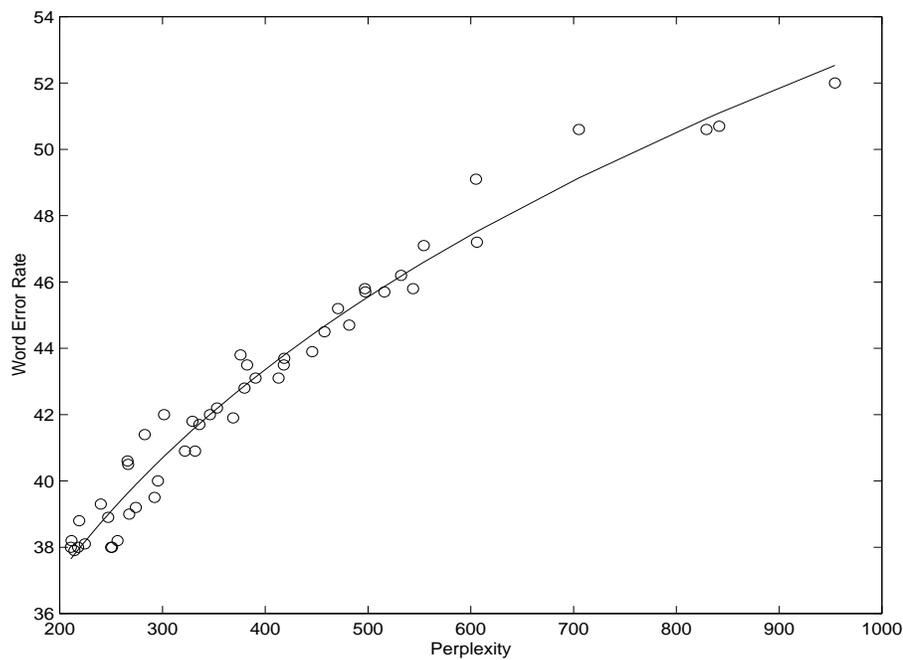


Figure 8.2 Word error rates plotted against the corresponding perplexities, with the best fit line satisfying $w = ap^r$ shown as a comparison.

correlation between word error rate and p^r .

The perplexity and value for p^r was calculated for each of the language models, and the correlation between these measures and word error rate was evaluated. The results are shown in Table 8.3.

	r	r_s	T
Perplexity	0.955	0.955	0.840
$p^{0.22}$	0.968	0.955	0.840

Table 8.3 Correlation of perplexity-based measures with word error rate

Finally, since the perplexities of the language models are now being evaluated based on the reference transcription, it is desirable to verify that there is not a mismatch between previously used language model test set and the actual recognition task.

From Table 8.1, it can be seen that model 2 is the baseline trigram model used in previous chapters, model 32 is the thirty component mixture model described in Chapter 5, and model 28 is the cache-based model with a cache component weight of 0.1 described in Chapter 6.

The perplexity figures computed on both the reference transcription and

the language model test set are presented for each of these three models. The results are shown in Table 8.4. They show that similar reductions in perplexity are observed when adaptation is applied, regardless of whether the perplexity is computed on the test text or the reference transcription.

Model	Adaptation	Perplexity (test)	Perplexity (ref)	Word Error Rate
2	None	134.4	251.0	37.9
28	Cache	119.0	211.0	38.0
32	Mixture	117.9	211.5	38.2

Table 8.4 Word error rates and perplexities of selected models

8.3.2 Rank-based measures

Perplexity measures the language model's success according to the probability it assigns to each of the words in the test text. An alternative is to evaluate the language model according to the proportion of words which have a higher probability than the target word at each time point. By so doing, the measure would encode the quality of the target word's prediction relative to the other words with which it will be competing within the speech decoder.

The *rank* of the target word, given a particular history is defined as the word's position in an ordered list of the word probabilities. Thus the most likely word has rank one, the least likely has rank V .

For each language model, the rank of each word in the reference file was calculated. Hence the *mean log rank* of each language model was computed, and the strength of the correlation between this measure and word error rate evaluated. The results are shown in Table 8.5.

	r	r_s	T
Mean log rank	0.967	0.957	0.846

Table 8.5 Correlation of mean log rank with word error rate

These results indicate that the mean log rank is at least as well correlated with word error rate as perplexity is.

8.3.3 Entropy

Given a particular word history w_1^i and a language model, the *entropy* (see Section 2.1) of the probability distribution over the vocabulary is given by

$$H = - \sum_{w \in \mathbf{V}} \hat{P}(w | w_1^i) \log_2 \hat{P}(w | w_1^i). \quad (8.6)$$

Therefore, the entropy is related to the expected value of the log probability given the word history in the following way:

$$E(\log_2 \hat{P}(w | w_1^i)) = -H. \quad (8.7)$$

Since log probability and entropy are related in this way, and perplexity is based on the *mean* log probability of words in the test text, the measure that was developed was based on the *mean* entropy over the test text.

The strength of the correlation between this measure and the word error rate was calculated, and the results are shown in Table 8.6.

	r	r_s	T
Mean entropy	-0.799	-0.792	-0.602

Table 8.6 *Correlation of mean entropy with word error rate*

These results show that the entropy is not as well correlated with word error rate as some of the other proposed language model evaluation measures. In particular, it is inferior to perplexity. This is unsurprising, since the entropy contains only information about the distribution in general, and no information about the target word in particular. However, there is a clear correlation displayed by these results, so some useful information is certainly present in this measure. In Section 8.4 the manner in which this information can be used in a more fruitful way will be investigated.

8.3.4 Low probability estimates

The set of fifty language models makes it possible not only to investigate new language model evaluation measures, but also to evaluate previously proposed ones. In particular, (Bahl et al., 1989) compare language models according to the number of words in the test text which receive probability estimates below a certain threshold. Their premise is that recognition errors are strongly

correlated with very low language model estimates. Therefore, the correlation between word error rate and measures of the form $L(x)$ which measure the proportion of words whose probability estimate is less than or equal to x was investigated. The results are shown in Table 8.7.

	r	r_s	T
$L(2^{-5})$	-0.919	-0.893	-0.726
$L(2^{-10})$	-0.915	-0.917	-0.768
$L(2^{-15})$	-0.833	-0.817	-0.640
$L(2^{-20})$	-0.646	-0.544	-0.388

Table 8.7 *Correlation of low probability estimates with word error rate*

These results indicate that the number of very low probability estimates in the test set is not as well correlated with word error rate as some of the other measures investigated in this chapter. Certainly the measure used in (Bahl et al., 1989) ($L(2^{-15})$) is much less well correlated with word error rate than perplexity is for the set of language models investigated here.

8.4 Combining measures

This section investigates methods of combining the information from some of the measures described above. It begins by examining the correlation between these measures to investigate which might usefully be combined. Then measures which combine the information from the target word's log probability and entropy are constructed and evaluated.

8.4.1 Correlation of features

The following features were selected:

- Probability
- Rank
- Entropy

and the value of r_s for each pair of features was calculated based on their values for each of the words in the test text according to the baseline broadcast news language model (model 2 in Table 8.1). The results are shown in Table 8.8.

Feature 1	Feature 2	r_s
Probability	Rank	-0.985
Probability	Entropy	-0.378
Rank	Entropy	0.381

Table 8.8 Correlation between language model features

These results clearly show that there is a very strong correlation between a word's probability and its rank. That is, the two features provide very similar information. Conversely, there seems to be much less correlation between a word's probability and the entropy of the distribution at that point in the test text. Thus, the information provided by these features is, in some sense, orthogonal. Given that both features provide information which is useful in predicting word error rate (see Tables 8.6 and 8.3), it seems that if the information sources can be combined, a superior measure of language model quality would result.

8.4.2 Combination of log probability and entropy

In order to develop measures of language model quality which are better correlated with word error rate, the information from the probability of the target word and the entropy at each point in the test text was combined.

Since the entropy H is the negative value of the expected log probability of the forthcoming word, the values that were combined were the log probability of the target word $\log_2(\hat{P}(w_i | w_1^{i-1}))$ and the negative entropy $-H(w_1^{i-1}) = \sum_{w \in \mathbf{V}} \hat{P}(w | w_1^{i-1}) \log_2(\hat{P}(w | w_1^{i-1}))$. These values were combined using linear interpolation, both in the log domain, leading to a measure which will be referred to as $C_{\log}(\lambda)$ and after converting back from the log domain, giving a measure called $C_{\text{lin}}(\lambda)$. If the test text is w_1^n , then these measures can be expressed as

$$C_{\log}(\lambda) = \frac{1}{n} \sum_{i=1}^n \left[-\lambda H(w_1^{i-1}) + (1 - \lambda) \log_2(\hat{P}(w_i | w_1^{i-1})) \right] \quad (8.8)$$

and

$$C_{\text{lin}}(\lambda) = \frac{1}{n} \sum_{i=1}^n \left[\lambda 2^{-H(w_1^{i-1})} + (1 - \lambda) \hat{P}(w_i | w_1^{i-1}) \right]. \quad (8.9)$$

The values of these measures were computed for a range of values of λ . The strength of the correlation between the resulting measures and word error rate was computed. The results are shown in Table 8.9.

	r	r_s	T
$C_{\log}(0)$ (Baseline)	0.966	0.955	0.840
$C_{\log}(0.05)$	0.969	0.960	0.853
$C_{\log}(0.1)$	0.971	0.965	0.868
$C_{\log}(0.2)$	0.971	0.964	0.863
$C_{\log}(0.3)$	0.964	0.957	0.837
$C_{\text{lin}}(0.001)$	0.970	0.962	0.853
$C_{\text{lin}}(0.002)$	0.970	0.963	0.856
$C_{\text{lin}}(0.01)$	0.965	0.955	0.842
$C_{\text{lin}}(0.02)$	0.959	0.952	0.835

Table 8.9 Correlation of combined measures with word error rate

These results show that combining the information from the two sources leads to language model evaluation measures which are better correlated with word error rate than either of the individual measures. In particular, $C_{\log}(0.1)$ performs considerably better than perplexity in this respect. This clearly demonstrates that information concerning the manner in which the probability mass is distributed over non-target words is useful in predicting word error rate.

8.5 Application to language model development

In Chapter 5, mixture-based language models were described. The interpolation weights assigned to each component were selected to maximise the likelihood (and hence to minimise the perplexity) of previously seen text. This led to models which had considerably lower perplexities than the baseline trigram model, but no decrease in word error rate. Furthermore, in section 7.1.1, it was shown that even choosing the interpolation weights to maximise the likelihood of the reference transcription of the speech being decoded led to no improvement in word error rate.

This chapter has described the development of measures of language model quality which correlate better with word error rate than perplexity does. Since the ultimate aim of mixture-based models is to reduce word error rate, the interpolation weights should be chosen with this in mind. Attempting to optimise them with respect to perplexity has not had positive results, so in this section

the interpolation weights are optimised with respect to measures which have been shown to correlate better with word error rate.

Recall from Chapter 5 that the probability estimate from a mixture-based language model is simply a linear combination of the probability estimates from a set of component language models:

$$\hat{P}(w_i | w_1^{i-1}) = \sum_j \lambda_j \hat{P}_{\mathcal{M}_j}(w_i | w_1^{i-1}). \quad (8.10)$$

The aim, therefore, is to select interpolation weights λ_j in order to optimise a more appropriate measure. In this case, we aim to maximise $C_{\log}(0.1)$. If the test text is w_1^n , then in the case of the mixture-based model this takes the value

$$C_{\log}(0.1) = \sum_{i=1}^n \left[0.1 \left(\sum_{w \in \mathbf{V}} \sum_j \lambda_j \hat{P}_{\mathcal{M}_j}(w | w_1^{i-1}) \log_2 \left(\sum_j \lambda_j \hat{P}_{\mathcal{M}_j}(w | w_1^{i-1}) \right) \right) + 0.9 \log_2 \left(\sum_j \lambda_j \hat{P}_{\mathcal{M}_j}(w_i | w_1^{i-1}) \right) \right]. \quad (8.11)$$

Evaluating this function requires all of the appropriate probability estimates from each of the language models to be available. Storing them in memory is impractical (31 language models, each with 65,000 probability estimates for each of the 22,000 words in the test text will require the storage of more than 4×10^{10} probabilities), and computing them all on the fly will require a similarly unreasonable quantity of CPU time. Furthermore, during the course of most numerical optimisation algorithms, this objective function would need to be evaluated several hundred times. Clearly, some form of simplification is required in order to pick optimal (or near optimal) weights.

Since $C_{\log}(0.1)$ is a linear combination of an expected log probability and a log probability, there is some justification for treating it as a log probability. Thus, the value of $2^{C_{\log}(0.1)}$ may be thought of as a “modified probability”. These modified probabilities can be computed for each word in the test text by each of the component language models in the mixture-based model. Interpolation weights can then be computed based on these streams of modified probabilities using the EM algorithm in the same way as it was used on the conventional probability streams in Chapter 5.

This technique was applied to generate new interpolation weights for the 30 component (plus the full language model) mixture-based models trained on both the broadcast news corpus and the British national corpus. In both

cases, supervised and unsupervised adaptation were applied. Lattice rescoring experiments were carried out using the resulting interpolation weights. The results are presented in Table 8.10, and are compared with the results of using the conventional maximum likelihood weights.

Training text	Adaptation	Weights	
		Maximum likelihood	WER optimised
Broadcast news	Unsupervised	38.2%	38.1%
Broadcast news	Supervised	38.0%	37.9%
BNC	Unsupervised	42.3%	41.9%
BNC	Supervised	41.8%	41.6%

Table 8.10 Comparison of word error rates achieved by maximum likelihood and word error rate optimised interpolation weights

These results show that the new *word error rate optimised* weights perform consistently better than the the old maximum likelihood weights. While the difference in performance is small, the overall difference between the word error rate optimised and maximum likelihood weights is statistically significant at the 1% level according to the matched pairs sentence segments word error test (Gillick and Cox, 1989) described in Appendix A.

It should be noted that the ability to choose more appropriate interpolation weights does not merely allow for improved adaptive language models to be created. Any language model in which multiple information sources are combined using weights chosen to minimise perplexity can be improved using this method. Such techniques are currently used in many state of the art systems for the transcription for broadcast news. In the 1998 broadcast news systems from both LIMSI (Gauvain et al., 1999) and Cambridge University's HTK group (Woodland et al., 1999), individual language models are constructed for each source of training data, and combined using linear interpolation with weights chosen to minimise perplexity. In such cases, it is to be expected that word error rate could be reduced by the application of word error rate optimised interpolation weights.

8.6 Summary

This chapter has described the development of new measures of language model quality. It has been shown that some measures based on the entire probability distribution (rather than simply the probability of the target words) are better

correlated with word error rate than perplexity is. Moreover, it has been shown that mixture-based language models are improved by applying interpolation weights which are optimised with respect to these new measures.

Conclusions and Future Work

This thesis has examined means by which statistical language models can be adapted towards the current topic and style of discourse. In the course of this research, the manner in which the word error rate of a recogniser is dependent on the language model's perplexity has been investigated, and alternative measures to perplexity which correlate better with word error rate have been developed.

In Chapter 5 experiments using adaptive mixture-based language models were presented, and in Chapter 6 similar experiments using cache-based models were described. Both were shown to lead to a considerable decrease in perplexity as compared to the baseline trigram model, but neither improved recognition performance. This provided motivation for the work described in Chapter 7 in which several experiments were carried out to investigate the reasons for this discrepancy. The results of these experiments led to the conclusions that in order to accurately predict word error rate from a set of held-out text, it is important to not only consider the probabilities of correct words (as perplexity does) but also the manner in which the remaining probability mass is distributed over the alternative words. This observation led to the work described in Chapter 8 in which alternatives to perplexity are developed and evaluated.

This final chapter summarises the experimental work which has been described in the previous chapters, describes the original contribution, and places it in the context of both the previously existing work, and work which was carried out by other researchers during the course of this research.

9.1 Review of original contribution

9.1.1 Adaptive language modelling

In Chapter 5 mixture-based language models were investigated. Algorithms used to automatically cluster the articles into topic-homogeneous components were described, and the quality of the resulting partition was investigated. The resulting mixture-based language models were shown to have a considerably lower perplexity than the baseline model, but did not lead to a corresponding reduction in word error rate.

This work was the first attempt to generate mixture-based language models by applying automatic clustering techniques to very large, very varied corpora, and to compare the effectiveness of the techniques across corpora. It also represented the first attempt to evaluate the effectiveness of these techniques. This evaluation was conducted in a quantitative way (by comparing the automatically generated partition with the manually generated partition of the BNC) and a qualitative way (by examining the words with high χ^2 scores for each component).

Since this work was conducted, (Seymore and Rosenfeld, 1997) has described similar work on the broadcast news task. The approach described in that paper differs from that described in this thesis in two important ways. Firstly, in (Seymore and Rosenfeld, 1997) the training corpus is partitioned according to keywords which are attached to each article in the corpus. This results in a set of over 5,000 topic-specific components. This approach therefore requires some degree of hand-labelling of the corpus which is not required by the approach described in this thesis. Secondly, not all of the topic-specific components are ultimately used. Instead, information theoretic measures are used to select the n most appropriate topic-specific language models (where $n = 5, 10, 20$), and these are interpolated with the full language model. In so doing, perplexity is reduced by approximately the same amount as the mixture-based models described in this thesis. However, very little reduction in word error rate is reported.

In Chapter 6, cache-based language models were explored. The effectiveness of cache-based adaptation for both the broadcast news corpus and British National Corpus was investigated, in terms of its effect on both perplexity and word error rate. It was observed that the rectangular window applied by standard cache-based techniques does not accurately model word re-occurrence probabilities, and this motivated the development of the decaying history. This represents the significant original contribution of Chapter 6. It does, however, have some similarity to work conducted concurrently by Thomas Niesler

(Niesler and Woodland, 1997).

The work in (Niesler and Woodland, 1997) differs from that described in this thesis in a number of respects. The first is that it applies to a class-based model (see Section 2.2.2.1) as opposed to the word-based models used in this thesis. Secondly, the approach is applied to trigger-pairs, rather than the solely self-trigger-based model described here. Finally, in (Niesler and Woodland, 1997) the exponential decay parameters are estimated individually for each trigger-pair in the model, whereas in the work in this thesis a global decay weight is assigned.

Allowing different decay parameters for each word in the vocabulary leads to a huge increase in the computational effort required to use the model. In the approach described in Chapter 6, all the words have the same decay parameters, and so the normalisation term β in equation (6.7) is constant at each point. However, if all words are allowed to have different decay parameters (as in (Niesler and Woodland, 1997)), it is necessary to recalculate the value of β each time the cache was updated. The computation required to do this for a word-based model is prohibitive. However, for the class-based model it is a less significant problem, since each normalisation must take place only over the words in one class, rather than the entire vocabulary.

9.1.2 Perplexity and word error rate

Chapter 7 described a set of experiments designed to investigate the apparent discrepancy between the results of the perplexity and word error rate experiments conducted in the previous two chapters. It was shown that the errors in the initial transcription are not a major cause of the discrepancy, and that performing adaptation on a language model trained on less appropriate, more general text yielded reductions in both perplexity and word error rate. Furthermore, language models with the same perplexities which resulted in different word error rates were developed, and the differences between them were investigated. These experiments revealed that there was insufficient information contained in the language model probability estimates of the words in the test text to distinguish between such models. Therefore, in order to develop more successful predictors of error rate, it is necessary to also consider the language model probabilities of alternative words. This conclusion motivated the word described in the following chapter.

In Chapter 8, the development of new measures of language model quality was described. The strength of the correlation between these measures and word error rate was evaluated on a set of fifty language models. This set of

language models was also used to quantitatively evaluate the manner in which perplexity is related to error rate.

It was shown that by using information concerning the way in which the probability mass is distributed over alternative, non-target words it is possible to derive measures of language model success which are considerably better correlated with word error rate than perplexity is. It was further shown that the recognition performance achieved using mixture-based language models was increased when the mixture weights were chosen to optimise these new measures, rather than to minimise perplexity. This last result is particularly important, as many state-of-the-art systems currently employ language models combined using linear interpolation weights chosen to satisfy a maximum likelihood criterion. The results presented here imply that the performance of such systems could be improved if the mixture weights were selected according to a more appropriate criterion, although since these systems have many fewer mixture components, the effect may not translate.

9.2 Future directions

The work in this thesis can be extended in several ways. The scope for future work is principally within the areas of mixture- and cache-based language modelling, and measures of language model quality.

9.2.1 Mixture-based language models

The key area in which the research into mixture-based language models presented in Chapter 5 can be extended is in the development of improved algorithms for clustering the articles into topic or style homogeneous components. The algorithm presented in Section 5.2 can be improved in a number of ways.

Firstly, the metric by which the distance between an article and a cluster is computed could be improved. In this work it was calculated according to unigram counts. If some computationally feasible way of basing this on trigram counts could be found, then one would expect to derive a more appropriate partition of the training text.

Secondly, the structure of the algorithm itself has some scope for development. A simple extension would be to apply an agglomerative, rather than k -means clustering algorithm. Further development could focus on algorithms which allow each article to belong to more than one component, since some articles will be relevant to more than one topic.

9.2.2 Cache-based language models

Since cache-based language models were first introduced in (Kuhn and De Mori, 1990), most of the embellishments which have been proposed have shown little improvement. For example, most of the gains from trigger-pair models come from self-triggers (Rosenfeld, 1994), and bigram and trigram caches are seldom much more effective than the standard unigram approach (Pye and Woodland, 1996). It must therefore be considered unlikely that applying these extensions to the simple cache-based models presented in Chapter 6 will have a major impact on their performance.

However, the decaying history based model does offer some scope for further development. If some method of allowing the reoccurrence probability of each word to decay at a different rate could be found (as in (Niesler, 1997)) without the computational requirement becoming prohibitive, it seems likely that the model would be improved.

9.2.3 Measures of language model quality

In Chapter 8, measures of language model quality were developed which are better correlated with word error rate than perplexity is. These measures were evaluated on a set of fifty language models which resulted in varying word error rates. One limitation of this work is the range of word error rates was a little limited, since none of the models resulted in error rates below 37.9%. An obvious avenue for future research would be to ensure that the findings of this chapter are also valid when systems with lower error rates are considered, as state of the art systems for broadcast news currently achieve error rates of around 14% (Woodland et al., 1999).

The new measures of language model quality are based not only on the probabilities assigned to the target words, but also on the probabilities assigned to alternative words. There are many ways in which this information could be used in measures of language model quality, and only a few of these have been investigated in this thesis. There is therefore a good deal of scope for developing measures which correlate even more strongly with word error rate. Furthermore, additional information concerning the acoustic confusability of words could be incorporated. Such improved measures will be useful in themselves, but there is also much potential for the development of more sophisticated methods of optimising mixture weights based on the new measures than that described in Section 8.5. Improvements in these areas should lead to greater recognition accuracy using mixture-based language models.

9.3 Outlook

As speech recognition technology matures, and is applied to more varied tasks, so the need for more flexible, adaptive language models becomes more pressing. This thesis has contributed to the understanding of such models, and has described new measures by which their success can be evaluated.

The Matched Pairs Sentence Segment Word Error Test

When evaluating new techniques for speech recognition, it is important to be able to determine whether any observed differences in performance are statistically significant, or whether they can reasonably be attributed to random chance.

The statistical significance of performance differences between some of the language models described in this thesis have been evaluated using the matched pairs sentence segment word error test (Gillick and Cox, 1989).

Suppose that we have the output from two speech recognition systems A and B , and that this output is divided into segments bounded by two or more words correctly recognised by each system, or by utterance boundaries. It is assumed that the errors in one segment are statistically independent of the errors in any other segment. The null hypothesis \mathbf{H}_0 is that the performance of the two systems is the same, and that the mean number of errors in each segment is therefore the same.

Let N_A^i be the number of errors in the output of system A in segment i , and N_B^i be the number of errors made by system B . Also let $Z_i = N_A^i - N_B^i$ for $i = 1, 2, \dots, n$, where n is the number of segments. Then the sample mean and variance of the Z_i are

$$\hat{\mu}_z = \frac{1}{n} \sum_{i=1}^n Z_i \tag{A.1}$$

and

$$\hat{\sigma}_z^2 = \frac{1}{n-1} \sum_{i=1}^n (Z_i - \hat{\mu}_z)^2 \tag{A.2}$$

If W is defined as

$$W = \frac{\hat{\mu}_z}{(\hat{\sigma}_z/\sqrt{n})}. \quad (\text{A.3})$$

then if n is sufficiently large W approximates to a Normal distribution with unit variance. The null hypothesis is that expected number of errors of the two systems are the same, i.e. that $\mu_z = 0$. To test the null hypothesis we compute $p = 2P(Y \geq |w|)$, where Y is a random variable with distribution $\mathcal{N}(0, 1)$, and w is the measured value of W . If $p < \alpha$, for a chosen significance level α , the null hypothesis is rejected. Here $\alpha = 0.01$ was used.

Bibliography

Bahl, L., Brown, P., de Souza, P., and Mercer, R. (1989). A Tree-Based Statistical Language Model for Natural Language Speech Recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 37(7).

Bahl, L., Jelinek, F., and Mercer, R. (1983). A Maximum Likelihood Approach to Continuous Speech Recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 5(2).

Berenson, M., Levine, D., and Rindskopf, D. (1988). *Applied Statistics. A first course*. Prentice-Hall International.

Berger, A., Della Pietra, S., and Della Pietra, V. (1996). A Maximum Entropy Approach to Natural Language Processing. *Computational Linguistics*, 22(1).

Brown, P., de Souza, P., Mercer, M., Della Pietra, V., and Lai, J. (1992). Class-Based n-gram Models of Natural Language. *Computational Linguistics*, 18(4):467–479.

Burnard, L. (1995). *Users Reference Guide for the British National Corpus*. Oxford University Computing Services.

Carter, D. (1994). Improving Language Models by Clustering Training Sentences. Technical report, SRI International.

Charniak, E. (1993). *Statistical Language Learning*. The MIT Press.

Chase, L. (1997). Blame Assignment for Errors Made in Large Vocabulary Speech Recognizers. In *Proceedings of the European Conference on Speech Communication and Technology*, Rhodes, Greece.

- Chen, S., Beeferman, D., and Rosenfeld, R. (1998). Evaluation Metrics for Language Models. In *Proceedings of the ARPA Workshop on Human Language Technology*.
- Clarkson, P. and Robinson, A. (1997). Language Model Adaptation using Mixtures and an Exponentially Decaying Cache. In *Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing*, Munich, Germany.
- Clarkson, P. and Robinson, A. (1998). The Applicability of Adaptive Language Modelling for the Broadcast News Task. In *Proceedings International Conference on Spoken Language Processing*, Sydney, Australia.
- Clarkson, P. and Rosenfeld, R. (1997). Statistical Language Modeling Using the CMU-Cambridge Toolkit. In *Proceedings of the European Conference on Speech Communication and Technology*, Rhodes, Greece.
- Cook, G., Kershaw, D., Christie, J., and Robinson, A. (1997). The Transcription of Broadcast Television and Radio News: The 1996 Abbot System. In *ARPA Spoken Language Technology Workshop*.
- Cover, T. and Thomas, J. (1991). *Elements of Information Theory*. John Wiley.
- Darroch, J. and Ratcliff, D. (1972). Generalized Iterative Scaling for Log-Linear Models. *Annals of Mathematical Statistics*, (43):1470–1480.
- Dempster, A., Laird, N., and Rubin, D. (1977). Maximum Likelihood from Incomplete Data Using the EM Algorithm. *Journal of the Royal Society of Statistics*, 39(1):1–38.
- Eide, E., Gish, H., Jeanrenaud, P., and Mielke, A. (1995). Understanding and Improving Speech Recognition Performance Through the Use of Diagnostic Tools. In *Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing*, Detroit, USA.
- Ferretti, M., Maltese, G., and Scarci, S. (1989). Language Model and Acoustic Information in Probabilistic Speech Recognition. In *Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing*, Glasgow, UK.
- Gauvain, J.-L., Lamel, L., and Jardino, M. (1999). The LIMSIS 1998 Hub-4E Transcription System. In *Proceedings of DARPA Broadcast News Workshop*, Herndon, Virginia, USA.

- Gillick, L. and Cox, S. (1989). Some Statistical Issues in the Comparison of Speech Recognition Algorithms. In *Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing*, Glasgow, UK.
- Good, I. (1953). The Population Frequencies of Species and the Estimation of Population Parameters. *Biometrika*, 40.
- Graff, D. (1996). The 1996 Broadcast News Speech and Language-Model Corpus. In *Proceedings ARPA Workshop on Human Language Technology*.
- Iyer, R. and Ostendorf, M. (1996). Modeling Long Distance Dependencies in Language: Topic Mixtures vs. Dynamic Cache Models. In *Proceedings International Conference on Spoken Language Processing*, Philadelphia, USA.
- Iyer, R., Ostendorf, M., and Meteer, M. (1997). Analysing and Predicting Language Model Improvements. In *Proceedings IEEE Workshop on Speech Recognition and Understanding*.
- Iyer, R., Ostendorf, M., and Rohlicek, J. (1994). Language Modeling with Sentence-Level Mixtures. In *Proceedings of the ARPA Workshop on Human Language Technology*.
- Jelinek, F. (1990). Self-Organized Language Models for Speech Recognition. In Waibel, A. and Lee, K.-F., editors, *Readings in Speech Recognition*, pages 450–506. Morgan Kaufman Publishers.
- Jelinek, F., Merialdo, B., Roukos, S., and Strauss, M. (1991). A Dynamic Language Model for Speech Recognition. In *Proceedings of Speech and Natural Language DARPA Workshop*, pages 293–295.
- Johansson, S., Atwell, E., Garside, R., and Leech, G. (1986). *The Tagged LOB Corpus User's Manual*. Norwegian Centre for the Humanities.
- Katz, S. (1987). Estimation of Probabilities from Sparse Data for the Language Model Component of a Speech Recognizer. *IEEE Transactions on Acoustics, Speech and Signal Processing*, 35(3):400–401.
- Klakow, D. (1998). Log-Linear Interpolation of Language Models. In *Proceedings International Conference on Spoken Language Processing*, Sydney, Australia.
- Kneser, R. and Steinbiss, V. (1993). On the Dynamic Adaptation of Stochastic Language Models. In *Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing*, Minneapolis, USA.

- Kuhn, R. and De Mori, R. (1990). A Cache-Based Natural Language Model for Speech Reproduction. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 12(6):570–583.
- Kuhn, R. and De Mori, R. (1992). Corrections to ‘A Cache-Based Natural Language Model for Speech Reproduction’. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 14:691–692.
- Ney, H., Essen, U., and Kneser, R. (1994). On Structuring Probabilistic Dependencies in Stochastic Language Modelling. *Computer Speech and Language*, 8(1):1–38.
- Niesler, T. (1997). *Category-based Statistical Language Models*. PhD thesis, Cambridge University Engineering Department.
- Niesler, T. and Woodland, P. (1996). A Variable-Length Category-Based n-gram Language Model. In *Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing*, Atlanta, USA.
- Niesler, T. and Woodland, P. (1997). Modelling Word-pair Relations for Category-based Language Models. In *Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing*, Munich, Germany.
- Odell, J. (1995). *The Use of Context in Large Vocabulary Speech Recognition*. PhD thesis, Cambridge University Engineering Department.
- Pallett, D. and Fiscus, J. (1997). 1996 Preliminary Broadcast News Benchmark Tests. In *Proceedings of DARPA Speech Recognition Workshop*, Chantilly, Virginia, USA.
- Placeway, P., Schwartz, R., Fung, P., and Nguyen, L. (1993). The Estimation of Powerful Language Models from Small and Large Corpora. In *Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing*, Minneapolis, USA.
- Pye, D. and Woodland, P. (1996). Cache language model adaptation. Unpublished Technical Report.
- Rosenfeld, R. (1994). *Adaptive Statistical Language Modeling: A Maximum Entropy Approach*. PhD thesis, School of Computer Science, Carnegie Mellon University. Published as *Technical Report CMU-CS-94-138*.

- Rosenfeld, R. (1995). The CMU Statistical Language Modeling Toolkit, and its use in the 1994 ARPA CSR Evaluation. In *ARPA Spoken Language Technology Workshop, Austin, Texas, USA*.
- Rosenfeld, R. (1997). Personal communication.
- Schechtman, R. (1994). Lexical Priming for Improved Language Modelling. Master's thesis, Cambridge University Engineering Department.
- Schwartz, R. and Chow, Y.-L. (1990). The N-best Algorithm: An efficient and exact procedure for finding the N most likely sentence hypotheses. In *Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing, Albuquerque, USA*.
- Seymore, K. and Rosenfeld, R. (1996). Scalable Backoff Language Models. In *Proceedings International Conference on Spoken Language Processing, Philadelphia, USA*.
- Seymore, K. and Rosenfeld, R. (1997). Using Story Topics for Language Model Adaptation. In *Proceedings of the European Conference on Speech Communication and Technology, Rhodes, Greece*.
- Shannon, C. and Weaver, W. (1949). *Mathematical Theory of Communication*. Urbana, Ill.
- Siegel, S. and Castellan, N. (1988). *Nonparametric Statistics for the Behavioural Sciences*. McGraw Hill, 2nd edition.
- Stolcke, A. (1998). Entropy-based Pruning of Backoff Language Models. In *Proceedings of the ARPA Workshop on Human Language Technology*.
- Stolcke, A., König, Y., and Weintraub, M. (1997). Explicit Word Error Minimization in N-Best List Rescoring. In *Proceedings of the European Conference on Speech Communication and Technology, Rhodes, Greece*.
- Witten, I. and Bell, T. (1991). The Zero-Frequency Problem: Estimating the Probabilities of Novel Events in Adaptive Text Compression. *IEEE Transactions on Information Theory*, 37(4):1085–1094.
- Woodland, P., Gales, M., Pye, D., and Valtchev, V. (1996). The HTK Large Vocabulary Recognition System for the 1995 ARPA H3 Task. In *Proceedings DARPA Speech Recognition Workshop*.

Woodland, P., Hain, T., Johnson, S., Niesler, T., Tuerk, A., Whittaker, E., and Young, S. (1998). The 1997 HTK Broadcast News Transcription System. *Proceedings DARPA Broadcast News Transcription and Understanding Workshop*.

Woodland, P., Hain, T., Moore, G., Niesler, T., Povey, D., Tuerk, A., and Whittaker, E. (1999). The 1998 HTK Broadcast News Transcription System: Development and Results. In *Proceedings of DARPA Broadcast News Workshop*, Herndon, Virginia, USA.

Wright, J. H., Jones, G. J. F., and Wrigley, E. N. (1992). Hybrid Grammar-Bigram Speech Recognition System with First-Order Dependence Model. In *Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing*, San Francisco, USA.