Category-based statistical language models

Thomas Niesler

St. John's College

June 1997

Thesis submitted to the University of Cambridge in partial fulfilment
of the requirements for the degree of Doctor of Philosophy

Synopsis

Language models are computational techniques and structures that describe word sequences produced by human subjects, and the work presented here considers primarily their application to automatic speech-recognition systems. Due to the very complex nature of natural languages as well as the need for robust recognition, statistically-based language models, which assign probabilities to word sequences, have proved most successful.

This thesis focuses on the use of linguistically defined word categories as a means of improving the performance of statistical language models. In particular, an approach that aims to capture both general grammatical patterns, as well as particular word dependencies, using different model components is proposed, developed and evaluated.

To account for grammatical patterns, a model employing variable-length *n*-grams of part-of-speech word categories is developed. The often local syntactic patterns in English text are captured conveniently by the *n*-gram structure, and reduced sparseness of the data allows larger n to be employed. A technique that optimises the length of individual *n*-grams is proposed, and experimental tests show it to lead to improved results. The model allows words to belong to multiple categories in order to cater for different grammatical functions, and may be employed as a tagger to assign category classifications to new text.

While the category-based model has the important advantage of generalisation to unseen word sequences, it is by nature not able to capture relationships between particular words. An experimental comparison with word-based *n*-gram approaches reveals this ability to be important to language model quality, and consequently two methods allowing the inclusion of word relations are developed.

The first method allows the incorporation of selected word *n*-grams within a backoff framework. The number of word *n*-grams added may be controlled, and the resulting tradeoff between size and accuracy is shown to surpass that of standard techniques based on *n*-gram cutoffs. The second technique addresses longer-range word-pair relationships that arise due to factors such as the topic or the style of the text. Empirical evidence is presented demonstrating an approximately exponentially decaying behaviour when considering the probabilities of related words as a function of an appropriately defined separating distance. This definition, which is fundamental to the approach, is made in terms of the category assignments of the words. It minimises the effect syntax has on word co-occurrences while taking particular advantage of the grammatical word classifications implicit in the operation of the category model. Since only related words are treated, the model size may be constrained to reasonable levels. Methods by means of which related word pairs may be identified from a large corpus, as well as techniques allowing the estimation of the parameters of the functional dependence, are presented and shown to lead to performance improvements.

The proposed combination of the three modelling approaches is shown to lead to considerable perplexity reductions, especially for sparse training sets. Incorporation of the models has led to a significant improvement in the word error rate of a high-performance baseline speech-recognition system.

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. This thesis has not been submitted in whole or in part for a degree at any other institution. Some of the work has been published previously in conference proceedings ([57], [60], [62]). The length of this thesis, including appendices and footnotes, is approximately 43,000 words.

Acknowledgements

First and foremost I would like to thank my supervisor, Phil Woodland. His experienced opinion and sound advice have guided the course of this research to its successful conclusion, and with his consistent support I have already been able to publish and present parts of this research at respected conferences. It has been a privilege to work under his supervision.

Then I would like to express my gratitude to St. John's College, who have provided financial support in the form of a Benefactor's Scholarship, without which my doctoral studies would not have been possible. Furthermore I would like to acknowledge the generous additional funds provided by the Harry Crossley Foundation and the Cambridge University Engineering Department, which have allowed me to present my research at conferences in Atlanta, Philadelphia and Munich.

I am very grateful to Selwyn Blieden, Anna Lindsay, Christophe Molina, Eric Ringger, Piotr Romanowski, and my sister Carola, for proofreading all or sections of this manuscript. Without their help, I could not have presented the result of my research as it appears here. Then I have been fortunate to spend many working hours with my colleagues here in the Speech Laboratory, whom I thank for their continued assistance, good company, and for all that tea. Furthermore I owe a great deal to Patrick Gosling, Andrew Gee and Carl Seymour, the maintainers of the excellent computing facilities, without which this work would certainly not have been possible. The recognition experiments in this thesis were accomplished using the Entropic Lattice and Language Modelling Toolkit, and I thank Julian Odell for his major contribution to the preparation of this software, which has made the last phase of my research much simpler. Finally I would like to say how grateful I am to my family for their enduring support throughout the duration of my studies, and for their unquestioning faith in my abilities.

Table of contents

Chapter 1

Introduction

Research into **language modelling** aims to develop computational techniques and structures that describe word sequences as produced by human subjects. Such models can be employed to deliver assessments of the correctness and plausibility of given samples of text, and have become essential tools in several research fields. The work in this thesis primarily considers their application to speech recognition, which has developed into a major research area over the past 30 years. The current principal objective is the development of large-vocabulary recognisers for natural, unconstrained, connected speech. Despite consistent progress, considerable advances are still necessary before widespread industrial application becomes feasible. However, the very broad spectrum of potential applications[1] will ensure that this technology will gain extreme importance once it matures.

Two major approaches to the modelling of human language may be identified. The first relies on syntactic and semantic analyses of the sample text to determine the hierarchical sentence structure. Such analyses employ a set of rules to ascertain whether a sentence is permissible or not, and although it has been possible to describe a significant proportion of English usage in this way, complete coverage has remained elusive. This is due, at least in part, to the continuous changes taking place in a living language. Furthermore, utterances that are clearly not grammatical occur often in natural language, but cannot be dealt with by such analyses. This significant likelihood of failure under such naturally occurring circumstances has led to infrequent use of the rule-based approach in speech recognition systems. Instead, a second approach based on statistical techniques with intrinsically greater robustness to grammatical irregularities is usually taken. Such **statistical language models** assign to each word in an utterance a probability value according to its deemed likelihood within the context of the surrounding word sequence. The probabilities are inferred from a large body of example text, referred to as the *training corpus*. In this way the model may come to reflect language usage as found in practice, and not only its grammatical idealisation. Moreover, advantage may be taken of the recent vast increases in the amount of available training text, which now runs into hundreds of millions of words.

A speech-recognition system must find the most likely sentence hypothesis for each spoken utterance. When dealing with connected speech, not only are the words themselves unknown, but also their number and boundaries in time. For large vocabularies, this leads to an extremely high number of possible alternative segmentations of the acoustic signal into words. In this context, the language model evaluates the linguistic plausibility of partial or complete hypotheses. In conjunction with the remainder of the recognition system, this estimate assists in finding those hypotheses which are most likely to lead to the correct result, as well as those which may be discarded in order to limit the number to practical levels.

[1] Examples of such applications include: automatic dictation systems, hearing systems for the deaf, automated telephone enquiry, automated teaching of foreign languages, and voice-control of electronic and computer equipment.

Although this work focuses on their application to speech-recognition systems, language models are important components also in other areas[2], such as handwriting recognition, machine translation, spelling correction, and part-of-speech tagging.

The following sections will describe the main elements of a speech recognition system, highlight the role of the language model, and outline the scope of the remainder of this thesis.

1.1. The speech recognition problem

In speech recognition, the aim is to find the most likely sequence of words $\mathbf{w}$ given the observed acoustic data $\mathbf{x}$. This is achieved by finding that $\mathbf{w}$ which maximises the conditional probability $P(\mathbf{w}\,|\,\mathbf{x})$. From Bayes rule,

$$P(\mathbf{w}\,|\,\mathbf{x}) \;=\; \frac{P(\mathbf{x}\,|\,\mathbf{w})\cdot P(\mathbf{w})}{P(\mathbf{x})} \tag{1}$$

but since $P(\mathbf{x})$ is constant for a given acoustic signal, an equivalent strategy is to find :

$$\arg\max_{\forall \mathbf{w}} \left\{ P(\mathbf{x}\,|\,\mathbf{w})\cdot P(\mathbf{w}) \right\} \tag{2}$$

The acoustic component of the speech recogniser must compute $P(\mathbf{x}\,|\,\mathbf{w})$, whereas the language model must estimate the prior probability of a certain sequence of words, $P(\mathbf{w})$. Before focusing our attention on the latter, we will present some background on the acoustic processing.

1.1.1. Preprocessing of the speech signal

Speech obtained from a microphone or recording device is in the form of an analogue electrical signal. This is processed into a form suitable for use within a speech-recognition system by a series of operations commonly referred to collectively as the **front end**. Usually these steps include band-limiting the signal, sampling it, and then applying some spectral transformation to encode its frequency characteristics. This last step normally employs the short-term discrete Fourier transform, and a particular popular choice is to encode the speech as Mel-frequency cepstral coefficients [72], which capture the spectral characteristics of the signal on a Mel-frequency scale. The resulting discrete-time sequence of **observation vectors** is the output of the front-end, and is passed to the recognition algorithm.

1.1.2. The acoustic model

Central to every speech recognition system is a means of encoding the sounds comprising human speech. This has been most successfully achieved through the use of hidden Markov models (HMMs) [71], [94], although other approaches have also met with success [77]. By describing the observation vectors as a probabilistic time series, the HMM takes the inherent natural variability of human speech characteristics into account. Given a set of examples of a particular sound in the form of the corresponding observation vector sequences $\mathbf{x}$, the parameters of the model $\mathcal{M}$ may be adjusted to best represent this data in a probabilistic sense, often by optimising $P(\mathbf{x} \mid \mathcal{M})$. The model may consequently be employed to evaluate the likelihood of a new observation sequence with respect to its parameters, thus giving an indication of how similar the new measurement is to those originally used to determine its parameters. This likelihood is used to make a statistical decision regarding the utterance to be recognised, as illustrated in the

[2] A brief overview of these is given in section 2.8.

following figure, which depicts a simple speech recognition system capable of distinguishing between the words “hello” and “goodbye”.

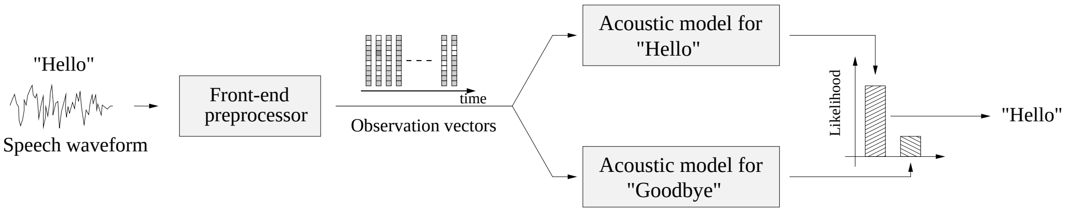

Figure 1.1: A simple speech recognition system.

HMMs may either be trained to model entire words directly, or to model subword units (such as phonemes) which are concatenated to obtain words. The latter approach is usually adopted since, even for moderately-sized vocabulary, there may not be sufficient training material to determine whole-word models reliably.

1.1.3. The language model

While the acoustic model indicates how likely it is that a certain sequence of words matches the measured acoustic evidence, it is the task of the language model to estimate the prior probability of the word sequence itself, i.e.:

$$\hat{P}\Big(\mathbf{w}(0, K-1)\Big) \tag{3}$$

where $\mathbf{w}(0, K-1) = w(0)\,, w(1)\,, \ldots, w(K-1)$ is the sequence of K words in question, and the caret denotes an estimate. This probability can assist the speech recognition system in deciding upon one of possibly several acoustically-similar, competing ways of segmenting the observation vectors into words according to their linguistic likelihood. From the definition of conditional probabilities, we may decompose the joint probability of equation (3) into a product of conditional probabilities:

$$\hat{P}\Big(\mathbf{w}(0, K-1)\Big) = \prod_{i=0}^{K-1} \hat{P}\Big(w(i)\,|\,\mathbf{w}(0, i-1)\Big) \tag{4}$$

where in practice $\mathbf{w}(0, -1)$ indicates the start-of-sentence symbol. Since use of a language model in the speech recognition process usually requires evaluation of the conditional probabilities appearing on the right hand side of equation (4), these are normally estimated directly. In developing a language model, the task is to find suitable structures for modelling the probabilistic dependencies between words in natural language, and then to use these structures to estimate either the conditional or the joint probabilities of word sequences.

1.2. Dominant language modelling techniques

This section presents a very brief summary of language modelling approaches prevalent in speech-recognition systems. The aim here is merely to place the scope of this thesis in context, and a more extensive review is presented in chapter 2.

Currently the most popular statistical language model is the **word *n*-gram**, which estimates probabilities from the observed frequencies of word *n*-tuples[3] in the training corpus. In particular, the probability of

[3] A sequence of *n* consecutive words.

a word is calculated using the frequency of the *n*-tuple, constituted by the preceding $(n-1)$ words of the utterance and the word itself. These models have the advantage of being quite simple to implement, computationally efficient during recognition, and able to benefit from the increasing amount of available training data. However, since each *n*-tuple is treated independently, these models fail to capture general linguistic patterns (such as the fact that adjectives do not normally precede verbs). Instead, they attempt to model each possible English *n*-tuple individually. This sometimes inefficient use of the information within the corpus can lead to data fragmentation and consequent poor generalisation to *n*-tuples that do not occur in the training set but which are nevertheless possible in real utterances. Moreover, since the number of *n*-tuples becomes extremely large as *n* increases, the models are very complex in terms of the number of parameters they employ. Both their large size (and consequent memory requirements), and the training set sparseness associated with large numbers of parameters, limits *n* to 2, 3 or perhaps 4. Hence these models cannot capture associations that span more than this number of words. Despite these restrictions, they remain the most successful type of language model currently used.

In order to counter the sparseness of the training corpus, and to improve model generalisation, language models that group words into **categories** have been proposed. By pooling data for words in the same category, model parameters may be estimated more reliably, and by capturing patterns at the category- as opposed to the word-level, it becomes possible to generalise to word sequences not present in the training data. The category definitions themselves may be available *a-priori*, for instance as **part-of-speech** classifications indicating the grammatical function of each word, or they may be determined automatically by means of an optimisation process. Models based on bigrams of part-of-speech categories have exhibited competitive performance relative to their word-based counterparts for sparse training corpora, but fare less well when the amount of training material increases. Optimisation algorithms for the category assignments allow this gap to be narrowed, particularly because the number of categories can be increased in sympathy with the size of the training set, but they suffer from high computational complexity. Bigram and trigram language models based on automatically determined categories have been used successfully in recognition systems when the training set is small, and also with some success for larger tasks in conjunction with word-based *n*-gram models.

A fundamental limitation of the *n*-gram approach is that it is not possible to capture dependencies spanning more than n words. Consequently the model is generally unable to capture long-range relations arising from such factors as the topic and style of the text. Empirical evidence suggests that a word which has already been seen in a passage is significantly more likely to recur in the near future than would otherwise be expected. A **cache** language model component addresses this by dynamically increasing the probability of words that have been seen in the recent text history, and in this way adapts to the local characteristics in the training set. Caches are usually employed in conjunction with *n*-gram models, and have led to performance improvements. However, they do not capture relationships between different words, and hence work has been carried out in finding associations between **word pairs**. Correlated word pairs have been detected,by measuring their mutual information (or related measure), and have been incorporated into a language model using various techniques. Although performance improvements have been achieved, these have been shown to be mostly due to correlations of words with themselves, an effect already addressed by a cache.

State-of-the-art speech-recognition systems continue to employ *n*-gram language models, based on words when the training sets are large enough, or based on word-categories when they are smaller, with the possible addition of a cache component.

1.3. Scope of this thesis

A basic assumption made in this thesis is that one may classify patterns found in language in the following manner:

1. **Syntactic patterns**, which refer to aspects of text structure imposed by grammatical constraints, for instance the phenomenon that adjectives are often followed by nouns.
2. **Semantic patterns**, which result from the meaning of the words themselves, and may themselves be subclassified as:
   (a) Fixed short-range relations: For example, the particular adjective “bright” often immediately precedes the particular noun “light”.
   (b) Loose long-range relations: For example, the nouns “sand” and “desert” may be expected to occur within the same sentence.

Word-based *n*-grams attempt to model all this information simultaneously by treating each possible *n*-tuple individually. While failing to identify general linguistic patterns, the proven success of this approach emphasises the importance of detailed word relations. The objective of this work has been to develop a model that deals separately with each type of pattern in the above classification, so as to reduce the data fragmentation inherent in word *n*-grams. In particular, since syntactic behaviour will be modelled explicitly, we may take advantage of available prior linguistic knowledge that is neglected by word-based approaches. The syntactic patterns in English are expected to be more universally consistent than word *n*-tuple frequencies, and therefore should lead to better generalisation to text styles and topics different from those of the training material.

1.3.1. Modelling syntactic relations

Word order is important for grammatical correctness in English and is captured naturally and effectively by *n*-grams. Furthermore, English has many local syntactic constructs [37] which may be captured by *n*-grams despite their limited range. For these reasons the syntactic language model component described in chapter 3 has been based on *n*-grams of word categories, where these categories have been chosen to correspond to part-of-speech classifications as a means of incorporating *a-priori* grammatical information in a straightforward manner. Since words generally have more than one grammatical function, it is necessary for the model to maintain several possible classifications of the sentence history in terms of category sequences. The generally much smaller number of categories than words lowers the training corpus sparseness and allows n to be increased, so that longer-range syntactic correlations may be accounted for. In particular, a technique is proposed which selectively extends the length of individual *n*-grams based on their expected benefit to performance, and thus allows the number of parameters to be minimised while improving performance. The result is a variable-length category-based *n*-gram language model.

1.3.2. Modelling fixed short-range semantic relations

Certain word combinations occur much more frequently than an extrapolation from the overall syntactic behaviour would suggest, such as the bigram of proper nouns “United Kingdom”. Sequences such as these are best modelled by word *n*-grams. Chapter 4 presents a technique by means of which important word *n*-grams may be combined with the syntactic model component within a backoff framework. In particular, the word *n*-gram is used to calculate a probability whenever possible, while the syntactic model serves as a fallback for all other cases. This allows frequent word *n*-tuples to be modelled directly, while others are captured by the syntactic model. Care is necessary when designing this mechanism since the multiple classifications of the sentence history into category sequences complicate the normalisation of the statistical model.

1.3.3. Modelling long-range semantic relations

Semantic relations due to factors such as the topic or the style of the text may span many words, and can therefore not be modelled by fixed-length sequences. In chapter 5 it is found that, following an appropriate definition of the distance between words, the occurrence probability of the more recent member in a related word pair displays an exponentially decaying behaviour as a function of this separation. The distance measure has been defined with the particular goal of minimising the effect that syntax has on word co-occurrences, and takes advantage of the grammatical word classifications implicit in the operation of the syntactic model. By assuming an exponentially decaying functional dependence between the occurrence probability of a word, and the distance to a related word, long-range correlations may be captured. Since only related words are treated and we restrict ourselves to word pairs, the model size may be constrained to reasonable levels. Methods are presented by means of which related word pairs may be identified from a large corpus, as well as techniques allowing the parameters of the functional dependence to be estimated.

1.4. Thesis organisation

Chapter 2 presents a detailed exposition of the statistical language modelling field. Chapter 3 describes the development of the syntactic language model component, chapter 4 the incorporation of short-range semantic relations by word *n*-grams, and chapter 5 the modelling of long-range semantic relations by means of word-pair dependencies. Chapter 6 shows how language models may be integrated into a speech recognition system, and presents some recognition results using the models developed in this thesis. Finally, chapter 7 presents a summary and conclusions.

Chapter 2

Overview of language modelling techniques

This chapter introduces the major statistical language modelling concepts and approaches. Appendix A gives a brief summary of the text corpora for which experimental results are encountered most frequently, and these will be referred to simply by name in the following. However, less frequent corpora will be described individually as they become relevant.

2.1. Perplexity

The true quality of a language model can only be evaluated through execution of an entire recognition experiment, since its utility is implicitly linked to the behaviour of the acoustic model. In particular, the language model is important for distinguishing between acoustically similar word hypotheses on grounds of their possible linguistic dissimilarity. Conversely, the language model need not discriminate strongly between words that are significantly dissimilar from an acoustic point of view. However, since the execution of a complete recognition experiment is computationally costly, it is desirable to have some way of evaluating the quality of the language model independently, before linking it to the acoustic component of the recogniser. Consequently the language model will first be viewed in isolation.

Let us regard the natural language under study to have been produced by an information source that emits symbols $z(i)$ at discrete time intervals $i \in \{0, 1, \ldots, \infty\}$ from a certain finite set according to some statistical law. The symbols might be the words of the language themselves, or they might refer to a more general concept, such as syntactic word categories. Let the process of emission of a symbol by the source be referred to as an event. Assuming the symbols $\mathbf{z}(0, K-1) = \{z(0)\,, z(1)\,, \ldots, z(K-1)\}$ to have been emitted by the source, and the probability of emission of this sequence to be $P(\mathbf{z}(0, K-1))$, the per-event self information of the sequence $\mathbf{z}(0, K-1)$ is [74], [85]:

$$I_s\Big(\mathbf{z}(0, K-1)\Big) \;=\; -\frac{1}{K}\cdot\log\Big[P\Big(\mathbf{z}(0, K-1)\Big)\Big]$$

The self information is an information-theoretic measure of the amount of information gained by witnessing the sequence $\mathbf{z}(0, K-1)$. Rare sequences, with correspondingly low associated probabilities, carry a larger amount of information than sequences seen very frequently. The per-event entropy, which is the average per-event self information, of the source is then:

$$\hbar \;=\; -\lim_{K\to\infty}\left(\frac{1}{K}\right)\cdot\sum_{\forall\mathbf{z}(0,K-1)}\left[P\Big(\mathbf{z}(0,K-1)\Big)\cdot\log\left[P\Big(\mathbf{z}(0,K-1)\Big)\right]\right]$$

where the summation is taken over all possible event sequences of length K that may be produced by the source. Entropy is an average measure of the amount of information contained in the set of sequences the source is capable of producing. A source that is able to emit a wide variety of different sequences will have a higher entropy than one capable of producing only a limited number. Assuming that the information source is ergodic[4] we may write the above ensemble average as :

$$\hbar \;=\; -\lim_{K\to\infty}\left(\frac{1}{K}\right)\cdot\log\left[P\Big(\mathbf{z}(0,K-1)\Big)\right]$$

Up to this point it has been assumed that the true probability associated with each sequence $\mathbf{z}(0,K-1)$ is known, so that the above equation will yield the actual entropy of the source. However, since the true mechanism by means of which language is produced is unknown, we are able to calculate only approximations of the probabilities $P(\mathbf{z}(0,K-1))$. Furthermore, the length of sequences $\mathbf{z}(0,K-1)$ at our disposal is always finite in practical situations, so that we must approximate the above equation by :

$$\hat{\hbar} \;=\; -\frac{1}{K}\cdot\log\left[\hat{P}\Big(\mathbf{z}(0,K-1)\Big)\right]$$

Thus the entropy of the source is approximated by the average per-event log probability of the observed sequence. In a language modelling framework, this sequence is the training corpus. It can be shown that, for an ergodic source, it is always true that:

$$\hat{\hbar} \geq \hbar$$

Due to the ergodicity of the source, the analysis of any sufficiently long sequence results in the same entropy estimate, thus :

$$\lim_{K\to\infty} P\Big(\mathbf{z}(0,K-1)\Big) \;=\; \varsigma$$

The corresponding figure obtained from the probability estimate is:

$$\hat{P}\Big(\mathbf{z}(0,K-1)\Big) \;=\; \hat{\varsigma}$$

When the approximation is perfect, $\hat{P}(\cdot) = P(\cdot)$ and therefore $\hat{\varsigma} = \varsigma$. However, an imperfect approximation $\hat{P}(\cdot)$ will assign a nonzero probability to invalid sequences (for which $P(\cdot) = 0$), and thus, since

[4] The property of ergodicity implies that the ensemble averages of a certain desired statistical characteristic (a mean, for example) equal the corresponding time averages [84]. An ensemble average is the average of the time averages of all possible member functions, where a member function is a single observed sequence. Let the member function be $\mathbf{z}(0,K-1)$, and the statistical property be $\vartheta\Big(\mathbf{z}(0,K-1)\Big)$. The time average for this case is given by $\varrho_t = \lim_{K\to\infty}\frac{1}{K}\cdot\vartheta\Big(\mathbf{z}(0,K-1)\Big)$ and the ensemble average by $\varrho_e = \lim_{K\to\infty}\frac{1}{K}\cdot\left[\sum_{\forall\mathbf{z}(0,K-1)}\left\{P\Big(\mathbf{z}(0,K-1)\Big)\cdot\vartheta\Big(\mathbf{z}(0,K-1)\Big)\right\}\right]$, where the summation is over all member functions. When the process producing the member functions $\mathbf{z}(0,K-1)$ is ergodic, it can be shown that $\varrho_t = \varrho_e$.

the probabilities must sum to unity, i.e.:

$$\sum_{\forall \mathbf{z}(0,K-1)} P\Big(\mathbf{z}(0,K-1)\Big) = 1 \qquad \text{and} \qquad \sum_{\forall \mathbf{z}(0,K-1)} \hat{P}\Big(\mathbf{z}(0,K-1)\Big) = 1$$

the estimated probability of the observed (and therefore valid) sequence must be lower than the exact figure and we have:

$$\hat{\varsigma} < \varsigma$$

This result is intuitively appealing since it implies that only a perfect model of the source will assign the highest probability to an actually observed sequence $\mathbf{z}(0,K-1)$, and that any imperfections in the model due to the approximation $\hat{P}(\cdot)$ will lead to a lower probability. A good model is therefore one which assigns the highest probabilities to the data gathered from the source output, and therefore this probability (or its logarithm) may be taken as a measure of the model quality. To make the measure independent of the sequence length, a per-event average may be used, i.e. either:

$$\left[\hat{P}\Big(\mathbf{z}(0,K-1)\Big)\right]^{-\frac{1}{K}} \tag{5}$$

or, taking the logarithm to base 2, we obtain once more the entropy estimate:

$$\hat{\hbar} \;=\; -\frac{1}{K}\cdot\log_2\left[\hat{P}\Big(\mathbf{z}(0,K-1)\Big)\right]$$

It has become customary in language modelling applications to use the **perplexity** *PP* as a measure of model quality. The perplexity is uniquely related to the estimated entropy :

$$\begin{aligned} PP \;&=\; 2^{\hat{\hbar}} \\ &=\; \left[\hat{P}\Big(\mathbf{z}(0,K-1)\,\Big)\right]^{-\frac{1}{K}} \end{aligned} \tag{6}$$

The perplexity is the reciprocal of the geometric mean of the probability of the sequence $\mathbf{z}(0,K-1)$, and may therefore be interpreted as the average branching factor of the sequence at every time instant according to the source model. For example, when the vocabulary size is N, there is no grammar[5], and we assume $\mathbf{z}(0,K-1)$ to be the sequence of words $\mathbf{w}(0,K-1)$, we have:

$$P\Big(\mathbf{w}(0,K-1)\Big) \;=\; \frac{1}{N^K}$$

and therefore equation (6) leads to the result

$$PP \;=\; N$$

[5] In the absence of a grammar, any word may follow any other word with the equal probability $\frac{1}{N}$.

Using the decomposition formula (4), the perplexity corresponds to:

$$\log\left(PP\right) \;=\; -\frac{1}{K}\sum_{i=0}^{K-1}\log\left[P\Big(z(i)\,|\,\mathbf{z}(0,i-1)\,\Big)\right] \tag{7}$$

Thus the perplexity of a passage of text for a given language model may be calculated by substituting the conditional probabilities computed by the model into the right hand side of equation (7). Apart from the constant $-\frac{1}{K}$, the perplexity is identical to the log probability of observing the sequence $x(0)\,,x(1)\,,\ldots,x(K-1)$ and thus minimising the perplexity is equivalent to maximising the log probability of the word sequence. Note that, since equation (5) and the language model probability component in Bayes rule (1) are optimised under the same conditions for a fixed sequence length, the assessment of language model quality by its perplexity effectively separates the acoustic and language model probability components to first approximation. The perplexity measure was first proposed by Jelinek, Mercer and Bahl [34].

Perplexity allows an independent (and thus computationally less demanding) assessment of the language model quality. However, because it is an average quantity, it does not indicate local changes across the corpus over which it is being calculated. In particular, the interaction with the acoustic probabilities is not considered, and therefore the perplexity measure does not account for the varying acoustic difficulty of distinguishing words. An improved quality measure that addresses this aspect is proposed in [38], but has the disadvantage of becoming specific to the nature of the particular acoustic component used in the speech recognition system.

2.2. Equivalence mappings of the word history

Recalling equation (4), the language model often estimates the conditional probabilities:

$$P\Big(w(i)\,|\,\mathbf{w}(0,i-1)\,\Big) \tag{8}$$

We will henceforth refer to $\mathbf{w}(0,i-1)$ as the **history** of the word $w(i)$. Due to the extremely large number of possible different histories, statistics cannot be gathered for each, and the estimation of the conditional probability must be made on the grounds of some broader classification of $\mathbf{w}(0,i-1)$.

Define an operator $H(w(i)\,)$ which maps the history $\mathbf{w}(0,i-1)$ of the word $w(i)$ onto one or more distinct **history equivalence classes**. Denote these by $h_j \,:\, j\in\{0,1,\ldots,N_H-1\}$, where N_H is the number of different equivalence classes found in the training corpus, so that the classification $H(\,\cdot\,)$ segments the words of the corpus into N_H subsets referred to collectively as $\mathbf{H}$:

$$\mathbf{H} = \Big\{h_0, h_1, \ldots, h_{N_H-1}\Big\} \tag{9}$$

When the history operator $H(\,\cdot\,)$ is many-to-one, i.e. each word history corresponds to exactly one equivalence class, the conditional probabilities (8) may be estimated by:

$$P\Big(w(i)\,|\,\mathbf{w}(0,i-1)\,\Big) \quad\approx\quad P\Big(w(i)\,|\,H(w(i)\,)\,\Big) \tag{10}$$

In general, however, the operator $H(\cdot)$ is many-to-many [38], in which case calculation of the probability involves summing over all history equivalence classes that correspond to $\mathbf{w}(0, i-1)$:

$$P\Big(w(i)\,|\,\mathbf{w}(0,i-1)\Big) \quad \approx \sum_{\forall h: h \in H\left(w(i)\right)} P\Big(w(i)\,|\,h\Big) \cdot P\Big(h\,|\,\mathbf{w}(0,i-1)\Big) \tag{11}$$

where $P(h\,|\,\mathbf{w}(0,i-1)\,)$ gives the probability that the word history $\mathbf{w}(0,i-1)$ belongs to the equivalence class h, and:

$$\sum_{j=0}^{N_H-1} P\Big(h_j\,|\,\mathbf{w}(0,i-1)\Big) \quad = \quad 1 \quad \forall \quad \mathbf{w}(0,i-1)$$

Note that equation (11) reduces to equation (10) when $\mathbf{w}(0,i-1)$ is many-to-one, since then $P(h\,|\,\mathbf{w}(0,i-1)\,)$ is nonzero for exactly one equivalence class. As examples, **bigram** language models define:

$$H(w(i)\,) \overset{def}{=} \mathbf{w}(i-1,i-1) = \{w(i-1)\} \tag{12}$$

and **trigram** language models define:

$$H(w(i)\,) \overset{def}{=} \mathbf{w}(i-2,i-1) = \{w(i-2)\,, w(i-1)\} \tag{13}$$

where $\mathbf{w}(i{-}n,i)$ refers to the sequence of $n{+}1$ words $\{w(i{-}n)\,, w(i{-}n{+}1)\,, \ldots, w(i)\,\}$. In contrast to the approach taken by equations (12) and (13), a decision tree is employed in [3] to map the word histories onto the equivalence classes.

2.3. Probability estimation from sparse data

In many instances language modelling involves the estimation of probabilities from a sparse set of measurements. Specialised estimation techniques are required under such circumstances, and the most prevalent of these in the language model field are described in this section. To begin with, we present a formal notation and framework within which the ensuing statistical problems may be formulated.

Consider a **population** $\bar{\Sigma}$ of measurements taken from all possible text in the language of interest. In practice we obtain a subset Σ of these measurements from a training corpus, so that Σ is a **sample** of the population, i.e. $\Sigma \subseteq \bar{\Sigma}$. The set of all possible unique measurements in $\bar{\Sigma}$ is the **sample space**, which will be denoted by $|\Sigma|$. Furthermore, define the **events** both of the population and of the sample space to correspond exactly to the outcomes of the measurements[6]. From this we note that all events in Σ and in $\bar{\Sigma}$ are members of $|\Sigma|$, but there may be events in $|\Sigma|$ and in $\bar{\Sigma}$ that are not members of Σ. Let the measurements be the detection of particular word patterns, the specific nature of which will depend on the method by which we choose to model the language[7]. Furthermore, assume the measurements to have a finite number of possible discrete values, so that the total number of different events in the sample space $|\Sigma|$ may be defined as N_σ, and each individual event denoted by σ_i with $i \in \{0, 1, \ldots, N_\sigma{-}1\}$.

[6] By way of illustration, these outcomes will often correspond to the words of the vocabulary.

[7] The most frequent choice of such pattern is an *n*-tuple of consecutive words for a chosen n, which is used as basis for an *n*-gram language model. Note that, in the formalism introduced here, bigrams and trigrams (for example) differ since they represent different measurements.

The total number of events in Σ may now be expressed as:

$$N_{\Sigma} = \sum_{i=0}^{N_{\sigma}-1} N(\sigma_i)$$

where $N(\sigma_i)$ is the number of times σ_i occurs in Σ, and we may have $N(\sigma_i) = 0$ for some σ_i.

In order to make this conceptual framework more concrete, consider as an example the word trigram language model, for which details are presented in section 2.4. In this case the events are the words of the vocabulary, i.e. $\sigma_i \equiv w_i$ and $N_{\sigma} = N_w$ where N_w is the vocabulary size. The trigram computes the probability of word $w(i)$ considering only the previous two words $\mathbf{w}(i-2, i-1)$, referred to as the **bigram context**. Now consider in isolation a particular bigram context, $\{w_a, w_b\}$, so that each measurement entails determining which word $w(i)$ follows a different occurrence of this bigram. The population $\bar{\Sigma}$ is then the collection of all words that have ever followed this bigram in the language of interest, and Σ is the particular subset obtained from the training corpus. Each distinct bigram context found in the training corpus defines a different population, but all populations have the same sample space $|\Sigma|$.

In general the history classification[8] $H(w)$ segments the training corpus into N_H subsets Σ_j, where $j = \{0, 1, \ldots, N_H - 1\}$ and N_H is the number of distinct history classifications found in the corpus. As the history classifications $H(\cdot)$ become more refined (for example, when we move from a bigram to a trigram), the discrimination of the language model improves since it can treat a greater variety of histories separately, but at the same time N_H increases and the number of events in each sample Σ_j decreases. This increasing sparseness of the Σ_j makes it more difficult to estimate event probabilities reliably. In particular, consider the maximum likelihood probability estimate for the event σ_i in the sample Σ_j, which corresponds to the relative frequency [18] :

$$P_{_{ML}}(\sigma_i \,|\, \Sigma_j) = \frac{N(\sigma_i \,|\, \Sigma_j)}{N_{\Sigma_j}}$$

where N_{Σ_j} is the total number of events in Σ_j. This estimator will assign a probability of zero to any event that has not been seen in the sample Σ_j. But when Σ_j is sparse, many events in $\bar{\Sigma}_j$ may be expected not to occur in Σ_j, and hence it is important that the probability estimator be refined to allow estimation of probabilities for such **unseen events**. The following sections review the Good-Turing, deleted and discounted estimation techniques, each of which address the estimation of marginal probabilities from sparse data and give particular attention to the computation of the unseen event probability. The backing-off and deleted interpolation approaches then suggest how this unseen probability may be distributed sensibly among the possible unseen events. Finally a discounting estimator which does not fit neatly into either of these categories is described.

2.3.1. The Good-Turing estimate

The Good-Turing probability estimation technique hinges on the **symmetry requirement**: the assumption that two events each occurring the same number of times in the sample must have the same probability of occurrence. For the original development of the Good-Turing estimate refer to [26], and for two alternative ways of arriving at the same result see [52].

[8] As introduced in section 2.2.

We begin by letting C_r refer to the number of different events occurring exactly r times in Σ, so that:

$$C_r \;=\; \sum_{i=0}^{N_\sigma - 1} \delta\Big(N(\sigma_i) = r\Big) \tag{14}$$

where:

$$\delta\left(x\right) = \begin{cases} 1 & \text{when } x \text{ is true} \\ 0 & \text{when } x \text{ is false} \end{cases}$$

From the definition (14) note that:

$$\sum_{r=1}^{R} C_r \;=\; M_\Sigma \tag{15}$$

and

$$\sum_{r=1}^{R} r \cdot C_r \;=\; N_\Sigma \tag{16}$$

where R is the largest number of times any event occurs in Σ, and M_Σ denotes the number of different events in Σ so that $M_\Sigma \leq N_\Sigma$. Finally, denote the set of all events that occur exactly r times in Σ by φ_r.

$$\varphi_r \;=\; \Big\{\sigma_i : N(\sigma_i) = r \quad \text{and} \quad i \in \{0, 1, \ldots, N_\sigma - 1\}\Big\}$$

The strategy employed by Good, as advocated by Turing [26], is to estimate the probability with which an event occurs in $\bar{\Sigma}$ when it is known to appear exactly r times in Σ. Denote this estimate by q_r so that, from the symmetry requirement, we may write:

$$q_r \;=\; \frac{\text{Probability in } \bar{\Sigma} \text{ of any event occurring } r \text{ times in } \Sigma}{\text{Expected number of events occurring exactly } r \text{ times in } \Sigma} = \frac{P(\varphi_r)}{\mathcal{E}\{C_r\}} \tag{17}$$

Assuming that the events occur independently (Bernoulli trials), the probability that there are precisely r sightings of σ_i in Σ is given by the binomial probability distribution [18]:

$$P\Big(N(\sigma_i) = r\Big) \;=\; \binom{N_\Sigma}{r} \cdot p_i^r \cdot (1 - p_i)^{N_\Sigma - r}$$

where for notational convenience we have defined $p_i = P(\sigma_i)$. Now, from (14), note that:

$$\mathcal{E}_{N_\Sigma}\{C_r\} \;=\; \sum_{i=0}^{N_\sigma - 1} \mathcal{E}\Big\{\delta\Big(N(\sigma_i) = r\Big)\Big\}$$

where the subscript of the expectation operator denotes the sample size. Note furthermore that:

$$\mathcal{E}_{N_\Sigma}\Big\{\delta\Big(N(\sigma_i) = r\Big)\Big\} \;=\; \text{probability of } \sigma_i \text{ occurring } r \text{ times in } \Sigma \;=\; \binom{N_\Sigma}{r} \cdot p_i^r \cdot (1 - p_i)^{N_\Sigma - r}$$

and therefore:

$$\mathcal{E}_{N_\Sigma}\{C_r\} \;=\; \binom{N_\Sigma}{r} \cdot \sum_{i=0}^{N_\sigma-1} p_i^r \cdot (1-p_i)^{N_\Sigma - r} \tag{18}$$

Now, since the events $\sigma_i : i = \{0, 1, \ldots, N_\sigma - 1\}$ form partitions of $\bar{\Sigma}$:

$$\begin{aligned} P(\varphi_r) \;&=\; \sum_{i=0}^{N_\sigma-1} \underbrace{P(N(\sigma_i) \;=\; r)}_{\binom{N_\Sigma}{r}\cdot p_i^r \cdot (1-p_i)^{N_\Sigma-r}} \cdot \underbrace{P(\sigma_i)}_{p_i} \\ &=\; \binom{N_\Sigma}{r} \cdot \sum_{i=0}^{N_\sigma-1} p_i^{r+1} \cdot (1-p_i)^{N_\Sigma - r} \end{aligned}$$

For clarity, define:

$$NW(r, N) \;=\; \binom{N}{r} \cdot \sum_{i=0}^{N_\sigma-1} p_i^r \cdot (1-p_i)^{N-r} \tag{19}$$

and note that:

$$\begin{aligned} NW(r{+}1, N_\Sigma{+}1) \;&=\; \binom{N_\Sigma{+}1}{r{+}1} \cdot \sum_{i=0}^{N_\sigma-1} p_i^{r+1} \cdot (1-p_i)^{N_\Sigma+1-r-1} \\ &=\; \frac{N_\Sigma{+}1}{r{+}1} \cdot P(\varphi_r) \end{aligned} \tag{20}$$

Now, from equations (18) and (19):

$$NW(r{+}1, N_\Sigma{+}1) \;=\; \mathcal{E}_{N_\Sigma+1}\{C_{r+1}\}$$

so that, using (20) it follows that:

$$\mathcal{E}_{N_\Sigma+1}\{C_{r+1}\} \;=\; \frac{N_\Sigma{+}1}{r+1} \cdot P(\varphi_r)$$

and finally substitution of this result into (17) leads us to:

$$q_r \;=\; \frac{(r{+}1) \cdot \mathcal{E}_{N_\Sigma+1}\{C_{r+1}\}}{(N_\Sigma{+}1) \cdot \mathcal{E}_{N_\Sigma}\{C_r\}}$$

Approximating the expectations by the counts:

$$\mathcal{E}_{N_\Sigma+1}\{C_{r+1}\} \approx C_{r+1} \quad \text{and} \quad \mathcal{E}_{N_\Sigma}\{C_r\} \approx C_r \tag{21}$$

we obtain Turing's formula [26] :

$$q_r = \frac{(r{+}1) \cdot C_{r+1}}{(N_\Sigma{+}1) \cdot C_r} \tag{22}$$

This is often approximated as [52] :

$$q_r = \frac{(r+1)\cdot C_{r+1}}{N_\Sigma \cdot C_r} \tag{23}$$

Section 2.3.2 discusses how the expression (22) may be shown to be optimal within a cross-validation framework [52]. From equation (23) the probability of an unseen event may be calculated easily :

$$\sum_{r\geq 0} C_r \cdot q_r = 1$$

$$\Rightarrow C_0 \cdot q_0 + \sum_{r>0} C_r \cdot q_r = 1$$

$$\Rightarrow C_0 \cdot q_0 + \frac{1}{N_\Sigma} \cdot \sum_{r>0} (r+1) \cdot C_{r+1} = 1 \quad \text{(using 23)}$$

$$\Rightarrow C_0 \cdot q_0 + \frac{1}{N_\Sigma} \cdot \sum_{r>0} r \cdot C_r - \frac{C_1}{N_\Sigma} = 1$$

so that, applying (16) we obtain the result:

$$C_0 \cdot q_0 = \frac{C_1}{N_\Sigma} \tag{24}$$

where the quantity $C_0 \cdot q_0$ is the probability of occurrence of any unseen event in Σ, while q_0 is the probability corresponding to any one particular unseen event. Therefore, adopting a relative frequency interpretation, we may view C_1 as an estimate of the number of unseen events in a sample of size N_Σ.

When the training set is sparse, the assumption (21) that the expectations may be approximated by the event counts may not be well founded. For example, it may occur that $C_r = 0$ for some r in which case the estimate (22) requires a division by zero. Good suggests smoothing the counts C_r before application of (22) to remedy this [26], while Katz suggests using (22) only for small r since sparse counts are more likely to occur for large r [41]. In particular, when r exceeds the threshold k, the use of the maximum likelihood estimate is advocated, judging the counts to be reliable in this case:

$$q_r = \frac{r}{N_\Sigma} \qquad \forall \;\; r > k$$

Requiring that the probability estimate of unseen events remain $\frac{C_1}{N_\Sigma}$, Katz [41] finds that a possible choice of q_r (others may be possible) is:

$$q_r \quad = \quad \left(\frac{\frac{(r+1)\cdot C_{r+1}}{r \cdot C_r} - \frac{(k+1)\cdot C_{k+1}}{C_1}}{1 - \frac{(k+1)\cdot C_{k+1}}{C_1}} \right) \cdot \frac{r}{N_\Sigma} \qquad \forall \;\; 1 \leq r \leq k \tag{25}$$

A typical value for k is 5. This approach may alleviate some of the problems associated with sparse C_r for large r. However, experiments with the LOB corpus have shown the above equation still to lack robustness and lead to unacceptable probability estimates.

Taking a different approach, the development in [55] describes methods by means of which constraints may be placed on the probability estimates, such as:

$$q_{r\text{–}1} \leq q_r$$

which requires the probability of more frequent events to equal or exceed that of less frequent ones, or:

$$\frac{r-1}{N_\Sigma} \leq q_r \leq \frac{r}{N_\Sigma}$$

which requires the estimates to lie close to the relative frequencies. Since other estimation methods have been found to be both simpler and more successful, these modifications will not be considered further.

2.3.2. Deleted estimation

The probabilities q_r may be estimated by employing the rotation method of cross-validation [35]. Define ϖ_r to be the sum of probabilities in $\bar{\Sigma}$ of all events occurring r times in Σ, and apply the symmetry requirement in order to write[9]:

$$\varpi_r \;=\; C_r \cdot q_r \qquad r = 1, 2, \ldots, R \tag{26}$$

Assume now that the training set has been partitioned into two sets termed the **retained** and **heldout** parts respectively. Let the counts C_r be determined from the former, so that the maximum likelihood estimate $\hat{\varpi}$ for ϖ with respect to the heldout set is :

$$\hat{\varpi}_r \;=\; \frac{Y_r}{\sum\limits_{k=1}^{R} Y_k}$$

where Y_r is the number of occurrences in the heldout part of all events appearing r times in the retained part, and R is now the highest frequency with which any event occurs in the retained part. Now assume the training corpus to be split into D disjoint sets, and the quantity Y_r for the case where the i_{th} such partition forms the heldout part to be denoted by Y_r^i. Let each of the D partitions form the heldout part in turn, the remaining $D-1$ constituting the retained part. The deleted estimate ϖ_r is defined to be that which maximises the product of the D heldout probabilities, and is found to be [52]:

$$\hat{\varpi}_r \;=\; \frac{\sum\limits_{i=1}^{D} Y_r^i}{\sum\limits_{i=1}^{D} \sum\limits_{k=1}^{R} Y_k^i} \tag{27}$$

When we choose $D = N_\Sigma$ so that the heldout part consists of exactly one sample[10], the result (27) reduces to the Good-Turing estimate [52], thus establishing the optimality of the latter within a cross-validation framework.

A comparison between the Good-Turing and deleted estimates has been made using approximately 22 million words of AP news text during training, and as much during testing. The results indicate the former method to be consistently superior [15], and thus no further emphasis will be given to the latter.

[9] By comparing equations (17) and (26), note that $\varpi_r \neq P\left(\varphi_r\right)$.

[10] This corresponds to the ”leaving-one-out” method of cross-validation [19].

2.3.3. Discounting methods

Katz interprets the process of probability estimation for unseen events as one in which the estimates for all seen events are decreased ("discounted") by a certain amount, and the resulting **discounted probability mass** then assigned to the unseen events [41]. This interpretation is used in [55] to postulate the form of the probability estimate as one which explicitly contains a **discounting factor** d_r :

$$q_r \;=\; \begin{cases} \dfrac{r - d_r}{N_\Sigma} & \text{if} \quad r > 0 \\ q_0 & \text{if} \quad r = 0 \end{cases} \tag{28}$$

As opposed to Turing's formula (22), which is sensitive to sparse C_r counts, the estimate (28) is very robust since it guarantees that $0 < q_r < 1$ as long as $d_r < r$, which is a condition under the control of the designer and independent of the counts obtained from the training set.

In order to obtain an expression for the discounted probability mass, recall that:

$$\sum_{r \geq 0} C_r \cdot q_r \;=\; 1$$

and solve for $C_0 \cdot q_0$ as follows:

$$C_0 \cdot q_0 + \sum_{r>0} C_r \cdot \left(\frac{r - d_r}{N_\Sigma} \right) \;=\; 1$$

$$\Rightarrow C_0 \cdot q_0 + \frac{1}{N_\Sigma} \cdot \sum_{r>0} C_r \cdot r - \frac{1}{N_\Sigma} \cdot \sum_{r>0} C_r \cdot d_r \;=\; 1$$

Finally apply equation (16) and obtain:

$$q_0 \cdot C_0 \;=\; \frac{1}{N_\Sigma} \cdot \sum_{r>0} C_r \cdot d_r \tag{29}$$

The probability of unseen events $q_0 \cdot C_0$ is in fact typically distributed among the possible unseen events by means of a more general distribution according to Katz's backing-off approach, which will be described in section 2.3.4. In both [53] and [55] two specific cases of discounting function are treated for the model (28), that of **absolute** discounting, and that of **linear** discounting.

2.3.3.1. Linear discounting

The linear discounting method chooses the discounting factor in equation (28) to be proportional to r :

$$d_r = a \cdot r \quad \text{with} \quad 0 < a < 1 \tag{30}$$

If we require the unseen probability to correspond to that delivered by the Good-Turing estimate (24), we obtain from (29) and (30) :

$$\frac{C_1}{N_\Sigma} \;=\; \frac{1}{N_\Sigma} \cdot \sum_{r>0} a \cdot r \cdot C_r$$

and then applying (16) we find :

$$a = \frac{C_1}{N_\Sigma} \tag{31}$$

This solution has been shown to optimise the leaving-one-out probability [55], and has been employed in the speech recognition system described in [36]. Katz states that the performance of the linear discounting estimator is not influenced strongly by the particular choice of discounting factor [41].

2.3.3.2. Absolute discounting

The method of absolute discounting chooses the discounting factor in equation (28) to be a positive constant b less than unity, i.e.:

$$d_r \;=\; b \quad \text{with} \quad 0 < b < 1 \tag{32}$$

We may once again require the unseen probability to equal that obtained with Turing's estimate so that, from equations (15), (29) and (32), we find:

$$\begin{aligned} q_0 \cdot C_0 \;&=\; \frac{b}{N_\Sigma} \cdot \sum_{r>0} C_r \\ &=\; \frac{M_\Sigma \cdot b}{N_\Sigma} \end{aligned}$$

and applying equation (24) it follows that:

$$b \;=\; \frac{C_1}{M_\Sigma} \tag{33}$$

As opposed to the Good-Turing estimator (22), this estimate is robust to the sample counts, since $C_1 < M_\Sigma$ for practical corpora (equality would require each event to be seen exactly once).

With the aim of finding an optimal value for the discounting factor, the leaving-one-out log probability is determined and subsequently differentiated with respect to b in [55]. The resulting equation can however not be solved exactly, although the following bound on the optimal was determined:

$$b \leq \frac{C_1}{C_1 + 2 \cdot C_2} < 1 \tag{34}$$

Empirical tests using *n*-gram language models show absolute discounting to give a 10% perplexity reduction over linear discounting for both the LOB as well as a 100,000-word German corpus [53].

2.3.4. Backing off to less refined distributions

The previous sections have addressed strategies useful in estimating the probability of an event σ_i in a sample Σ_j drawn from a population $\bar{\Sigma}_j$ when the sample is sparse. In each case the probability of encountering an unseen event may be computed, but the method of distributing this among the possible unseen evens is not specified. In the absence of further information, it seems reasonable to assume all events equally likely, and therefore to redistribute the probability mass $C_0 \cdot q_0$ uniformly. However, in language modelling applications, further information is often available in the form of a more general distribution obtained from a less specific definition of the history equivalence classes[11]. For example, when using a trigram, the bigram distribution may be considered.

Katz advocates redistributing the unseen probability among the possible unseen events, according to this more general distribution, in a process he terms **backing off** [41].

Consider two populations, $\bar{\Sigma}_j$ and $\bar{\Sigma}'_k$, the first based on a more specific and the second on a more general definition of the history equivalence class. Denote the sample we have from each population by Σ_j and Σ'_k respectively, and let the corresponding probability estimates be $P(\sigma_i \,|\, \Sigma_j)$ and $P(\sigma_i \mid \Sigma'_k)$. Finally, assume a method for calculating the probabilities of seen events to exist. The total probability of occurrence of any unseen event in the sample Σ_j is hence given by:

$$\begin{aligned} P_u(\Sigma_j) &= \sum_{\sigma_i : N\left(\sigma_i|\Sigma_j\right)=0} P\Big(\sigma_i \,|\, \Sigma_j\Big) \\ &= 1 - \sum_{\sigma_i : N\left(\sigma_i|\Sigma_j\right)>0} P\Big(\sigma_i \,|\, \Sigma_j\Big) \end{aligned} \tag{35}$$

Katz distributes this probability mass among the unseen events according to the lower-order probability $P(\sigma_i \,|\, \Sigma'_k)$ as follows:

$$P(\sigma_i \,|\, \Sigma_j) \overset{def}{=} \frac{P_u(\Sigma_j)}{\tilde{\alpha}(\Sigma_j)} \cdot P(\sigma_i \,|\, \Sigma'_k) \quad \forall \quad \sigma_i : N(\sigma_i \,|\, \Sigma_j) = 0 \tag{36}$$

It follows from (35) and (36) that:

$$P_u(\Sigma_j) = \frac{P_u(\Sigma_j)}{\tilde{\alpha}(\Sigma_j)} \cdot \sum_{\sigma_i : N\left(\sigma_i|\Sigma_j\right)=0} P(\sigma_i \,|\, \Sigma'_k)$$

$$\Rightarrow \tilde{\alpha}(\Sigma_j) = \sum_{\sigma_i : N\left(\sigma_i|\Sigma_j\right)=0} P(\sigma_i \,|\, \Sigma'_k)$$

It is more convenient to calculate :

$$\tilde{\alpha}(\Sigma_j) = 1 - \sum_{\sigma_i : N\left(\sigma_i|\Sigma_j\right)>0} P(\sigma_i \,|\, \Sigma'_k) \tag{37}$$

and for notational convenience define:

$$\alpha(\Sigma_j) \overset{def}{=} \frac{P_u(\Sigma_j)}{\tilde{\alpha}(\Sigma_j)} \tag{38}$$

so that from (35), (37) and (38) we find:

$$\alpha(\Sigma_j) = \frac{1 - \sum_{\sigma_i : N(\sigma_i|\Sigma_j)>0} P(\sigma_i \,|\, \Sigma_j)}{1 - \sum_{\sigma_i : N(\sigma_i|\Sigma_j)>0} P(\sigma_i \,|\, \Sigma'_k)} \tag{39}$$

[11] Refer to sections 2.2 and 2.3.

and the complete probability estimate for an event σ_i is given by :

$$P(\sigma_i \,|\, \Sigma_j, \Sigma'_k) \quad = \quad \begin{cases} P(\sigma_i \,|\, \Sigma_j) & \forall \quad N(\sigma_i \,|\, \Sigma_j) > 0 \\ \alpha(\Sigma_j) \cdot P(\sigma_i \,|\, \Sigma'_k) & \forall \quad N(\sigma_i \,|\, \Sigma_j) = 0 \end{cases} \tag{40}$$

Note that, if there are events unseen to both Σ_j and Σ'_k, the probability estimate $P(\sigma_i \mid \Sigma'_k)$ must itself employ a backoff to yet more general distribution, say $P(\sigma_i \mid \Sigma''_l)$. This recursion terminates once a distribution to which σ_i is no longer unseen is reached.

Note also that Katz's method is computationally efficient since the $\alpha(\Sigma_j)$ may be precomputed for each $j = \{0, 1, \ldots, N_H - 1\}$.

In principle any probability estimator that takes unseen events into account may be used in the backing off framework. In particular, Turing's estimate (22) may either be used directly or in its modified form (25). However, neither of these estimators is robust to strongly sparse C_r counts, and in practice the discounted estimates have been more successful. The linear discounted estimate (31) is employed in [41], while nonlinear discounting is advocated in [1].

2.3.5. Deleted interpolation

The backing-off approach is a means of combining probability estimates from two different populations $\bar{\Sigma}$ and $\bar{\Sigma}'$ in order to address the problem of unseen events. Assume now in general that we have N_p such populations, each of differing generality. Denote the sample we have from each population by $\Sigma^{(i)}$ where $i = \{0, 1, \ldots, N_p - 1\}$. The method of deleted interpolation combines the probability estimates obtained from these samples by means of a linear combination as follows:

$$P_{di}\Big(\sigma_i \,|\, \Sigma^{(N_p-1)}, \Sigma^{(1)}, \ldots, \Sigma^{(N_p-1)}\Big) \;=\; \sum_{j=0}^{N_p-1} \lambda_j\Big(\Sigma^{(N_p-1)}\Big) \cdot P\Big(\sigma_i \,|\, \Sigma^{(j)}\Big) \tag{41}$$

where $\Sigma^{(N_p-1)}$ is assumed to correspond to the most refined history equivalence class, and:

$$\lambda_j\Big(\Sigma^{(N_p-1)}\Big) > 0 \qquad \forall \quad j \in \{0, 1, \ldots, N_p - 1\} \quad \text{and} \quad \sum_{j=0}^{N_p-1} \lambda_j\Big(\Sigma^{(N_p-1)}\Big) = 1 \tag{42}$$

The smoothing parameters $\lambda_j(\Sigma^{(N_p-1)})$ are chosen to maximise the probability calculated using $P_{di}(\cdot \,|\, \cdot)$ of a cross-validation set, and hence the component probabilities of the right hand side of equation (41) will be weighted according to their utility with respect to the predictive quality of the language model.

In order to obtain the optimal smoothing parameters, an approach based on a hidden Markov model interpretation of equation (41) is proposed in [38]. The model has $N_p + 1$ states : one initial state s_{-1} and one corresponding to each of the terms on the right hand side of the equation, denoted here by $s_0, s_1, \ldots, s_{N_p-1}$. The $\lambda_j(\Sigma^{(N_p-1)})$ are interpreted as transition probabilities, and the $P(\sigma \mid \Sigma^{(\cdot)})$ as output probabilities. When the context $\Sigma^{(N_p-1)}$ is encountered, the model executes transitions from s_{-1} to each of the states $s_0, s_1, \ldots, s_{N_{p-1}}$ according to the transition probabilities, following which an event σ_i is emitted at each state according to the appropriate output probability. In this framework, the optimal $\lambda_j(\Sigma^{(N_p-1)})$ may be obtained by training the model using forward-backward reestimation, with some initial choice of $\lambda_j(\Sigma^{(N_p-1)})$ such that the constraints (42) are satisfied.

Although each distinct context generally has different smoothing parameters, the limited quantity of training data may require them to be tied appropriately. Alternatively, the need for forward-backward reestimation may be alleviated by using an empirically determined weighting function to calculate the smoothing parameters [64]. Results for a 70,000 word train information enquiry database, as well as the 1 million word Brown corpus, show perplexities to be increased only slightly by this simplification.

A performance comparison of the backing-off technique and the method of deleted interpolation[12] has shown both to give practically identical results [41], [51], and due to the higher implementational and computational complexity of the latter, it will not be pursued further in this work.

2.3.6. Modified absolute discounting

In [53] a variation of the discounting model (28) is presented. Instead of distributing the discounted probability mass only among unseen events in the backoff framework, it is used to smooth the probability estimate at all times. The following general discounting function is postulated to achieve this :

$$P(\sigma_i\,|\,\Sigma_j) \;\; = \;\; \frac{N(\sigma_i\,|\,\Sigma_j) - d(\sigma_i\,|\,\Sigma_j)}{N_{\Sigma_j}} + Q_d \cdot P(\sigma_i\,|\,\Sigma'_k) \tag{43}$$

where the discounting function $d(\sigma_i\,|\,\Sigma_j)$ is in general different for every event σ_i, and the samples Σ_j and Σ'_k are related as in section 2.3.4. Now from:

$$\sum_{i=0}^{N_\sigma - 1} P(\sigma_i\,|\,\Sigma_j) = 1$$

we have:

$$\sum_{i=0}^{N_\sigma - 1} \frac{N(\sigma_i\,|\,\Sigma_j) - d(\sigma_i\,|\,\Sigma_j)}{N_{\Sigma_j}} + \sum_{i=0}^{N_\sigma - 1} Q_d \cdot P(\sigma_i\,|\,\Sigma'_k) \;\; = \;\; 1$$

$$\Rightarrow Q_d \;\; = \;\; \frac{1}{N_{\Sigma_j}} \cdot \sum_{i=0}^{N_\sigma - 1} d(\sigma_i\,|\,\Sigma_j) \tag{44}$$

Now choose the discounting function:

$$d(\sigma_i\,|\,\Sigma_j) = \begin{cases} D & \text{if} \quad N(\sigma_i\,|\,\Sigma_j) > 0 \\ 0 & \text{if} \quad N(\sigma_i\,|\,\Sigma_j) = 0 \end{cases}$$

where $0 \leq D \leq 1$. With this choice, the probability mass Q_d may be found from (15) and (44) to be :

$$\begin{aligned} Q_d \;\; &= \;\; \frac{D}{N_{\Sigma_j}} \cdot \sum_{\forall \sigma_i : N\left(\sigma_i | \Sigma_j\right) > 0} 1 \\ &= \;\; \frac{D \cdot M_{\Sigma_j}}{N_{\Sigma_j}} \end{aligned}$$

[12] This comparison was carried out for both a bigram and a trigram language model using a 750 000 word corpus of office correspondence.

so that substitution into (43) yields [53] :

$$P(\sigma_i \,|\, \Sigma_j) \;=\; \frac{\max\Big[N(\sigma_i \,|\, \Sigma_j) - D, 0\Big]}{N_{\Sigma_j}} + D \cdot \frac{M_{\Sigma_j}}{N_{\Sigma_j}} \cdot P(\sigma_i \,|\, \Sigma_j) \tag{45}$$

This approach has the advantage of resulting in strong smoothing when the proportion of the number of different events to the total number of events occurring in the context is large, a condition that will hold when these counts are sparse.

Ideally the values of D should be chosen to maximise a cross validation function, but exact solution appears to be mathematically intractable [53]. Furthermore, D should strictly be different for every Σ, but it may be practically advantageous to tie its value in some way. Empirical tests were performed in [53] for the following choices:

1. Modelling D individually for each Σ, and determining the optimal values by iterative adjustment of the parameters with a fixed step size over their allowable ranges so as to maximise the leaving-one-out training set probability.
2. Modelling D to be the same for all Σ, thus pooling their counts, and determining the optimal value in a similar fashion to that used above.
3. Modelling D to be the same value for all Σ, this time approximating it by the upper bound (34).

It is found that all three approaches give practically identical results, indicating that the estimator is insensitive to the choice of D. The estimator (45) is employed successfully in [32], [33].

2.4. N-gram models

An *n*-gram is a sequence of n consecutively occurring items. In this section we will assume these items to be words, bearing in mind that the principles may easily be extended to other forms of *n*-gram[13]. For this choice, the language model estimates the probability of the following word $w(i)$ conditioned on the identity of the preceding $n-1$ words $\mathbf{w}(i-n+1, i-1)$, termed the $(n-1)$-gram **context**. From section 2.2 we see that the history equivalence classes are therefore defined by:

$$\mathbf{H} \;\equiv\; \Big\{h_0, h_1, \ldots, h_{N_H}\Big\} \;\equiv\; \Big\{\mathbf{w}(i-n+1, i-1) : N\Big(\mathbf{w}(i-n+1, i-1)\Big) > 0\Big\}$$

where N_H is here the number of distinct $(n-1)$-tuples in the training corpus. Furthermore, the mapping operator $H(\cdot)$ is many-to-one and defined by:

$$H(w(i)\,) \;\equiv\; \mathbf{w}(i-n+1, i-1) \tag{46}$$

For clarity define $\mathbf{w}_j^c$ to be the $(n-1)$-gram context corresponding to h_j, i.e.:

$$\mathbf{w}_j^c \;\equiv\; \mathbf{w}(i-n+1, i-1) \quad \text{when} \quad H(w(i)\,) = h_j$$

[13] Section 2.5, for example, introduces *n*-grams of word-categories.

Now consider a sample space $|\Sigma|$, with the words of the vocabulary as events, i.e.:

$$\sigma_i \equiv w_i \quad \forall \quad i = \{0, 1, \ldots, N_\sigma - 1\} \quad \text{with} \quad N_\sigma = N_w$$

With reference to the formalism introduced in section 2.3, we denote the population and the sample corresponding to each history classification by $\bar{\Sigma}_j$ and Σ_j respectively, so that :

$$\Sigma_j \;=\; \Big\{ w(i) : H(w(i)\,) = h_j \Big\} \tag{47}$$

while bearing in mind that all Σ_j continue to have the same sample space $|\Sigma|$. The population $\bar{\Sigma}_j$ corresponding to each such context $\mathbf{w}_j^c$ then comprises all words that have followed or will ever follow it in the language of interest, while a subset Σ_j of this population consists of all words that have in fact followed the context in the training corpus.

The number of occurrences of an event w_i in the sample Σ_j is given by the number of times the n-gram formed by the concatenation of the context $\mathbf{w}_j^c$ and the word w_i is seen in the training corpus:

$$N\Big(w_i \,|\, \Sigma_j\Big) \;=\; N\Big(\{\mathbf{w}_j^c, w_i\}\Big)$$

and the total number of events in the sample is given by the number of occurrences of the context itself:

$$N_{\Sigma_j} \;=\; \sum_{j=0}^{N_w} N\Big(w_j \,|\, \Sigma_j\Big) \;=\; N(\mathbf{w}_j^c)$$

As n increases, the potential number of distinct contexts N_H increases exponentially, and hence the number of words per sample decreases for a fixed training corpus. Even for small n the data in each sample are often sparse, and n must in practice be restricted to 2 (**bigrams**) or 3 (**trigrams**)[14]. Each technique presented in section 2.3 has been applied to *n*-gram probability estimation, but since the Good-Turing estimator was found to be consistently superior to the deleted estimate [15], and nonlinear discounting to deliver better performance than the Good-Turing estimator [1], the latter has been used in further work. In particular, absolute discounting was used in the backoff framework with the $(n–1)$-gram estimate as the more general distribution. Letting $p = n-1$, we find from (28), (33) and (40) :

$$P\Big(w(i)\,|\,\mathbf{w}(i-p, i-1)\Big) \;=\; \begin{cases} \dfrac{N\Big(\mathbf{w}(i-p,i)\Big) - b}{N\Big(\mathbf{w}(i-p,i-1)\Big)} & \text{if} \quad N\Big(\mathbf{w}(i-p,i)\Big) > 0 \\ \alpha\Big(\mathbf{w}(i-p,i-1)\Big)\cdot & \\ P\Big(w(i)\,|\,\mathbf{w}(i-p+1,i-1)\Big) & \text{if} \quad N\Big(\mathbf{w}(i-p,i)\Big) = 0 \end{cases} \tag{48}$$

and

$$b \;=\; \frac{C_1\Big(\mathbf{w}(i-p, i-1)\Big)}{M_\Sigma\Big(\mathbf{w}(i-p,i-1)\Big)} \tag{49}$$

[14] Results presented in [64] indicate small performance improvements when using larger n for a restricted-domain task (70,000-word train information enquiry database), but not for a larger and more diverse domain (the 1 million word Brown corpus). For larger tasks, 4-gram language models ($n = 4$) have been employed recently [91] but are still uncommon due to their extremely large size.

and from (39):

$$\alpha\Big(\mathbf{w}(i-p,i-1)\Big) \quad = \quad \frac{1 \; - \sum\limits_{w_i : N(w_i|\mathbf{w}(i-p,i-1))>0} P\Big(w(i)\,|\,\mathbf{w}(i-p,i-1)\Big)}{1 \; - \sum\limits_{w_i : N(w_i|\mathbf{w}(i-p,i-1))>0} P\Big(w(i)\,|\,\mathbf{w}(i-p+1,i-1)\Big)} \tag{50}$$

Backing off to the $(n-1)$-gram distribution is reasonable when the data are sparse, and convenient from an implementational point of view, but in some cases it may lead to overestimation of an unseen probability. In particular, the absence of an *n*-gram may be as a result of linguistic improbability rather than lack of data. This issue is addressed in [12], [78], in [44], and in [65], where experiments employ the WSJ 87-89 corpus in the first three cases, and a 9 million word corpus of newspaper text in the last. In each case perplexities are reduced, although word error rate reductions were only reported in [44].

2.5. Category-based language models

Instead of finding patterns among individual words, a language model may be designed to discover relationships between word groupings or **categories**. Taking this approach has the following advantages:

- Category-based models share statistics between words of the same category, and are therefore able to **generalise** to word patterns never encountered in the training corpus. This ability to sensibly process unseen events is termed language model **robustness**.
- Grouping words into categories can reduce the number of contexts in a model, and thereby counter **training set sparseness**.
- The reduction in the number of contexts leads to a more **compact** model employing fewer parameters, and therefore having more modest storage requirements which may be important from a practical standpoint.

The following sections introduce a formal notation for dealing with word categories and describe how these may be applied to the development of language models.

2.5.1. Word categories

A **category** will be taken to refer to any grouping of words. Let there be N_v such categories and denote them by:

$$\mathbf{V} = \{v_0, v_1, \ldots, v_{N_v}\}$$

Now define an operator $V(\cdot)$ that maps each word $w_i : i \in \{0, 1, \ldots, N_w\}$ to one or more categories $v_j : j \in \{0, 1, \ldots, N_v\}$, i.e.:

$$v_j = V(w_i) \quad j \in \{0, 1, \ldots, N_v - 1\} \quad \text{and} \quad i \in \{0, 1, \ldots, N_w - 1\} \tag{51}$$

where v_j is the category to which w_i is assigned by the operation $V(\,\cdot\,)$. When this mapping is many-to-one we will speak of **deterministic** category membership, while referring to **stochastic** membership when it is many-to-many.

Now assume that the probability of witnessing a word $w(i)$ is completely defined by a knowledge of the category to which it belongs, so that we may write:

$$P\Big(w(i)\,|\,\mathbf{w}(0,i{-}1)\Big) \;\approx\; P\Big(w(i)\,|\,v(i)\Big) \tag{52}$$

For stochastic category membership, this allows us to decompose the conditional probability estimates in the following way:

$$P\Big(w(i)\,|\,\mathbf{w}(0,i-1)\Big) \approx \sum_{\forall v: v\in V\big(w(i)\big)} P\Big(w(i)\,|\,v\Big)\cdot P\Big(v\,|\,\mathbf{w}(0,i{-}1)\Big) \tag{53}$$

Furthermore, classifying the history into equivalence classes, we find from equation (11) :

$$P\Big(v_j\,|\,\mathbf{w}(0,i{-}1)\Big) \approx \sum_{\forall h: h\in H\big(w(i)\big)} P\Big(v_j\,|\,h\Big)\cdot P\Big(h\,|\,\mathbf{w}(0,i{-}1)\Big) \tag{54}$$

In this framework a natural choice for the history equivalence class mapping is the identity of the most recent $n{-}1$ categories:

$$H(w(i)\,) \;=\; \Big\{v(i{-}n{+}1)\,, v(i{-}n{+}2)\,, \ldots, v(i{-}1)\,\Big\} \tag{55}$$

from which we obtain **category-based *n*-gram** language models. Note that equation (55) represents a many-to-many mapping when the operator $V(\cdot)$ is one-to-many. Equations (53), (54) and (55) with $i = 2$ are employed in [53] to construct a bigram language model based on linguistic **part-of-speech** word categories. Here each word is assigned to one or more categories by human experts according to its syntactic function. For the LOB corpus it is found that summing over all category and history equivalence class assignments (as opposed to choosing the most likely ones) reduces the perplexity by 20% . This agrees with the findings in [32], and indicates that multiple category membership improves modelling performance. Furthermore, the part-of-speech based model is found to have slightly lower perplexity than a word bigram, while employing fewer parameters. Larger improvements are obtained for a smaller German corpus of approximately 100,000 words, illustrating the improved generalisation of category-based models. A closed vocabulary was employed in both cases [43], [54].

When the history equivalence class mapping is many-to-one, equation (54) simplifies to:

$$P\Big(v(i)\,|\,\mathbf{w}(0,i{-}1)\Big) \;\approx\; P\Big(v_j\,|\,H(w(i)\,)\Big) \tag{56}$$

so that we may rewrite (53) as:

$$P\Big(w(i)\,|\,\mathbf{w}(0,i-1)\Big) \approx \sum_{\forall v: v\in V\big(w(i)\big)} P\Big(w(i)\,|\,V(w(i)\,)\Big)\cdot P\Big(V(w(i)\,)\,|\,H(w(i)\,)\Big) \tag{57}$$

Equation (57) has been used in the **synonym based** language model proposed in [38], which is very similar to the part-of-speech approach except in that the categories need not have strict grammatical definitions. Instead a *core vocabulary* $\mathcal{V}_{core}$ is defined, consisting a set of words that are assumed to exhibit all significant types of grammatical behaviour that may be encountered. Associated with each word w

in $\mathcal{V}_{core}$ is a *synonym list* $\mathcal{S}_w$, which is a list of words that display similar grammatical characteristics to w. The synonym lists are compiled automatically from the training corpus by identifying the word(s) in the core vocabulary with which the context of the new word agrees best. In the light of (57), $\mathcal{V}_{core}$ corresponds to the set of categories, and the synonym sets $\mathcal{S}_w$ to the category membership definitions.

Finally, when we restrict ourselves to deterministic membership, equation (57) simplifies to:

$$P\Big(w(i)\,|\,\mathbf{w}(0,i-1)\Big) \;\approx\; P\Big(w(i)\,|\,V(w(i)\,)\Big)\cdot P\Big(V(w(i)\,)\,|\,H(w(i)\,)\Big) \tag{58}$$

Furthermore, using the category *n*-gram of equation (55) with $n = 2$, we obtain from (58) :

$$P\Big(w(i)\,|\,\mathbf{w}(0,i{-}1)\Big) \;\approx\; P\Big(w(i)\,|\,V(w(i)\,)\Big)\cdot P\Big(V(w(i)\,)\,|\,V(w(i{-}1)\,)\Big) \tag{59}$$

which is the category-based bigram language model used in [30], [31], [32], [42] and [53] in conjunction with automatically-determined category membership, as will be described in the following section.

2.5.2. Automatic category membership determination

The word categories and history equivalence classes must be defined in some way before category-based language models can be used. Although some smaller corpora for which the words have been tagged with part-of-speech labels by linguistic experts are available, such hand-labelling is impractical for large amounts of text, and consequently methods allowing automatic grouping of words into categories have been investigated.

One option is to use a rule-based linguistic parser to tag the training corpus with linguistic part-of-speech and other information, and then to use these classifications as the word categories. However, since parsing is a very computationally intensive operation, it may be impractical to process the entire training corpus. In [90] a parser is used to tag an initial part of the training corpus, which is then used to initialise a statistical tagger, with which the remaining text is assigned with the most likely category labels.

Alternatively, some method of clustering the words into categories as part of an optimisation process may be employed. An optimisation algorithm that maximises the training set probability of the language model is described in [42] and [53]. A category-based bigram language model of the form (59) is used, so that each word may not belong to more than one category. Starting with some initial assignment, the algorithm evaluates the change in the training text log probability brought about by moving every word in the vocabulary from its current to every other possible category. The move resulting in the largest increase in the log probability is then selected, executed, and the process repeated. This continues until some convergence criterion is met.

In [53] the number of categories is assumed fixed for every optimisation run, this number being varied between 30 and 700 in repeated trials. It is found that the test set perplexity achieved after optimisation decreases initially as the number of categories increases, reaches a minimum, and then increases again, tending towards the word-bigram perplexity. Similar observations have been made in [69], where the category assignments were made by hand according to syntactic and semantic word functions. The following arguments account for this behaviour.

- **Overgeneralisation**: When there are few categories, the language model is not able to discriminate strongly among different patterns in the text.
- **Tradeoff** : As the number of categories increases, the extent to which the model must generalise diminishes, and the discrimination ability improves.
- **Overfitting**: When there are too many categories, the language model begins to reflect the peculiarities of the training set, and generalises less well to the test set.

However, since the bigram relative-frequencies are the maximum likelihood probability estimates, the training set log probability will continue to improve as the number of categories grows, until the number of categories equals the number of different words in the vocabulary and we are left with a word-based bigram language model.

Overfitting may be detected by employing cross-validation when calculating the training set log probability. In particular, the optimal number of categories may be determined automatically in this way. The leaving-one-out method of cross-validation [19] is employed in [42] to achieve this, and in conjunction with the described optimisation algorithm it has indeed been possible to estimate the best number of categories fairly accurately.

The algorithm in [42] has since been extended to allow clustering by maximising the category-based trigram log probability [48]. However, when employing large data sets, this extension leads to a large increase in required computation, and it is necessary to limit the number of categories to such an extent that it remains possible to build language models with better performance using the aforementioned bigram clustering.

The use of simulated annealing methods to find the optimal category membership assignments is advocated in [30], [31], [32], [33] and [88]. Once again the bigram model structure of equation (59) is employed, and the training set log probability is used as an optimisation criterion although now both the word as well as the category it is to be moved to are chosen by Monte Carlo selection. The decision as to whether the move is accepted or not is taken according to the Metropolis algorithm, which will occasionally allow the perplexity to increase. The conditions under which this may occur are governed by a control parameter which follows an annealing schedule such that moves leading to perplexity increases become increasingly unlikely with time. Monte Carlo minimisations were carried out with and without simulated annealing, and the 11% lower perplexities achieved in the former case demonstrate the existence of locally optimal category assignments.

Overall, the experimental results obtained with automatically-determined categories indicate that category-based models can improve on the performance of corresponding word-based models when the training corpus is small, but not when dealing with larger bodies of text [21], [48], [88].

Examples of the category membership after optimisation indicate that, while the most frequent words in each cluster often appear to be grouped by syntactic or semantic function, this is not always so. Indeed some categories are very difficult to justify intuitively [30], [31], [42], [53].

A greedy agglomerative algorithm for clustering words into word categories is presented in [8]. A bigram language model structure is assumed, and it is shown that for this model the training set log probability may be written as the sum of two terms: the unigram distribution entropy $\hbar(w)$ and the average **mutual information**[15] between adjacent categories $I_m\left(v_1, v_2\right)$.

$$LL \;=\; -\hbar(w) + I_m\left(v_1, v_2\right)$$

The optimal set of category assignments is that which maximises the mutual information component, since the entropy is independent of this choice. The greedy algorithm proposed in [8] initially assigns each word in the training corpus to its own category, and then at each iteration merges that category pair (not necessarily adjacent) which least decreases $I_m\left(v_1, v_2\right)$. This process continues until the desired number of categories has been reached. In this framework it is again assumed that each word may belong to only one category.

Due to the large number of potential merges that must be investigated at every step, considerable care is required during the implementation of the algorithm in order to ensure its computational feasibility. In particular, inherent redundancies in the calculation of the mutual information are taken advantage of in order to yield an efficient algorithm, which is ultimately used to cluster a very large corpus consisting of more than 365 million words. The word categories found in this way are used to build a category-based trigram language model in analogy to equation (59). This model is found to have a perplexity that is 11% higher than that of a a word-based trigram model. However, interpolation of the two models leads to a slight (3%) perplexity improvement.

2.5.3. Word groups and phrases

In order to improve the modelling of frequent word groups, it has been proposed that the language model vocabulary not be restricted to single words only, but that it be allowed also to contain frequently occurring phrases [38]. In particular, a technique relying on the concept of **mutual information** that allows these to be identified automatically from a text corpus is described.

The mutual information between two events x_i and x_j is given by:

$$I_m\Big(x_i, x_j\Big) \;=\; \log\left[\frac{P(x_i, x_j)}{P(x_i)\cdot P(x_j)}\right] \tag{60}$$

Now if the events are taken to be adjacently-occurring words, then $P(x_i, x_j)$ is the probability that x_j immediately follows x_i, and $P(x_i)$ and $P(x_j)$ are the unigram distributions of x_i and x_j respectively. Using relative frequency approximations, we may estimate these probabilities as :

$$P(x_i, x_j) \approx \frac{N(x_i, x_j)}{N\left(\cdot,\cdot\right)} \;\;;\;\; P(x_i) \approx \frac{N(x_i)}{N\left(\cdot\right)} \;\;;\;\; P(x_j) \approx \frac{N(x_j)}{N\left(\cdot\right)}$$

For a large corpus, $N\left(\cdot,\cdot\right) \approx N\left(\cdot\right) \equiv N$, so that we may estimate the mutual information as:

$$I_m\Big(x_i, x_j\Big) \;=\; \log\left[\frac{N(x_i, x_j)\cdot N}{N(x_i)\cdot N(x_j)}\right] \tag{61}$$

[15] Mutual information will be defined in section 2.5.3.

A high value of $I_m(x_i, x_j)$ indicates that x_j has followed x_i much more frequently in the text than would be expected were they to be generated independently. The procedure is then to find all consecutive word pairs with high mutual information and add these as single units to the vocabulary. Since the estimate in equation (60) may be unreliable for small counts, only word pairs occurring at least a threshold number of times are considered. In this way phrases such as "*nuclear magnetic resonance*" may be treated as single units by the language model. A similar approach[16] is taken in [8], where such associations are termed "**sticky pairs**". Treatment of such frequently occurring word groups as single lexicon entries has led to $3{-}8\%$ word error rate improvements relative to a word-based bigram model for a small task (35-45,000 words of training data) [88].

Instead of using the mutual information measure to identify word groups, the training set log probability can be maximised directly. This approach is taken in [25] and in [75], the first minimising the perplexity on a dedicated cross validation set, while the second minimising the training set leaving-one-out probability. Determination of the phrases is automatic, and proceeds by identifying the pair which would most decrease perplexity by merging, subsequent execution of this merge, and repetition. A 20% perplexity and a 12% word error rate reduction is achieved in [25] for experiments with a dialogue system having a training set size of 60,000 words, while results in [75] show a small (1.3% relative) improvement in word error rate for the Verbmobil task but no improvement for the larger Switchboard corpus. This difference is ascribed to the more constrained nature of the former corpus, and it is argued that fixed phrases are more useful in this case.

2.6. Longer-term dependencies

When based on *n*-grams, a language model is able to discriminate only among word histories that differ in the last $n-1$ words, where n very rarely exceeds 3. This may classify two histories as equivalent while they differ in an important respect due to some event that has taken place in the more distant past. Semantic relationships, for example, which may span large distances of text, affect the probabilities with which words may be expected to occur. For this reason methods of explicitly taking such longer range dependencies into account are relevant.

2.6.1. Pairwise dependencies

Certain long-range dependencies[17] may be taken into account by postulating that the probability of the next word $w(i)$ given the history $\mathbf{w}(0, i-1)$ may be decomposed into a set of independent pairwise probabilities $P(w(i) \,|\, w(j)\,)$, where $w(j) \in \mathbf{w}(0, i-1)$, i.e.:

$$P\Big(w(i) \,|\, \mathbf{w}(0, i-1)\Big) \;=\; F\Big(P\Big(w(i) \,|\, w(i-1)\Big), P\Big(w(i) \,|\, w(i-2)\Big), \ldots, P\Big(w(i) \,|\, w(0)\Big)\Big) \qquad (62)$$

where $F(\cdot)$ is an as yet undetermined function. By restricting ourselves to word pairs, the combinational explosion associated with *n*-grams for large n is avoided. However, since we are no longer limiting ourselves to consecutive words, the number of pairs is now, unlike in the bigram case, a significant fraction of N_w^2, where N_w is the vocabulary size [47]. For this reason it is necessary to filter the set of possible word pairs, retaining only those conveying a useful amount of information.

[16] No experimental results treating the incorporation of such pairs into a language model have been given in either source, however.
[17] Pairwise dependencies as described in this section are sometimes also referred to as **word associations** or **word triggers**.

A study of word association types discovered using a measure termed the **association ratio** $\psi\left(w_a, w_b\right)$, which is closely related to mutual information, is presented in [14].

$$\psi\left(w_a, w_b\right) \;=\; \frac{\dfrac{N(w_a, w_b, L_W)}{N_w \cdot (L_W - 1)}}{\left(\dfrac{N(w_a)}{N_w}\right) \cdot \left(\dfrac{N(w_b)}{N_w}\right)} \tag{63}$$

Here N_w is the size of the vocabulary, $N(w_a, w_b, L_W)$ the number of times w_b follows w_a in the corpus within a window of length L_W (typically 5 words), and $N(w_a)$ is the number of observations of w_a in the corpus. The association ratio differs from the mutual information in that it is not symmetric but encodes linear precedence. It is shown that, by using this measure as a selection criterion, many interesting word pair associations may be found. Particular attention is given to the range[18] and sequential order of the association. Ordering information is found to play a significant role for certain word combinations (e.g. “doctor” precedes “nurse” approximately 10 times more often than “nurse” precedes “doctor”). The association range is shown to vary widely according to the type of relationship the words have with each other, and the following four types have been identified:

- **Compound** : multiple-word expressions referring to single concepts, e.g. “Computer scientist”.
- **Fixed** : the words of interest are separated by a fixed number of words, e.g. “bread and butter”.
- **Lexical** : the words are related by syntactic factors.
- **Semantic** : the words are related by their meaning, e.g. “man“ and “woman”.

It is found that, for the first two types of association, the words usually co-occur within a narrow range. This is not so for the lexical and semantic associations, the latter in particular exhibiting a large variance of the association range. The following table, reproduced from [14], illustrates this:

			Separation (words)	
Relation	**Word *x***	**Word *y***	**Mean**	**Variance**
Compound	computer	scientist	1.12	0.10
	United	States	0.98	0.14
Fixed	bread	butter	2.00	0.00
	drink	drive	2.00	0.00
Lexical	refraining	from	1.11	0.20
	coming	from	0.83	2.89
	keeping	from	2.14	5.53
Semantic	man	woman	1.46	8.07
	man	women	-0.12	13.08

Table 2.1: Word relationships found using the association ration in [14].

A measure similar to (63) is used to identify associations employed in a long-distance bigram model [93]. This approach does not constrain the probability estimate to be conditioned on the preceding word,

[18] The number of words separating the word pair in question.

but allows it also to depend on a more informative word occurring further back in the sentence. Using this technique, perplexity improvements were reported for a constrained corpus of 1000 sentences.

Mutual information is used as a filter for word pairs also in [47]. The pairs found in this way are used in conjunction with a conventional trigram language model by means of the maximum entropy principle, which allows the combination of various knowledge sources while making minimal assumptions concerning the distributions. In doing so, a 12% reduction in perplexity over the conventional trigram model was achieved for 5 million words of Wall Street Journal training text. Most of this improvement was noted to be due to correlations of words with themselves (**self-triggers**).

Having selected word pairs with significant correlation, functional forms for the probability distribution $F(\cdot)$ in (62) may be postulated. It is suggested in [53], for example, that the probability estimate $P(w(i) \mid \mathbf{w}(0,i-1)\,)$ take on the form of a linear combination of the individual pairwise probabilities $P(w(i) \mid w(j)\,)$ over all words in the history, i.e. $j = \{1, 2, \ldots, p\}$:

$$P\Big(w(i)\,|\,\mathbf{w}(0,i-1)\Big) \;=\; \sum_{j=1}^{p} c_j \cdot P\Big(w(i)\,|\,w(i-j)\Big) \tag{64}$$

where

$$c_j \geq 0 \quad \text{and} \quad \sum_{j=1}^{p} c_j = 1 \tag{65}$$

The constraint (65) guarantees that the probabilities sum to unity. Note that, for this probabilistic model, the probabilities $P\,(w_i\,|\,w_k)$ depend only on the identities of w_i and w_k, and not on the positional distance between them. However, the weights c_j may explicitly depend on this distance, and can be interpreted as representing a type of window function over the most recent p words of text. Various choices are possible, including rectangular, triangular, Gaussian and Hamming-type functions. Alternatively, assuming the probabilities $P\,(w_i\,|\,w_k)$ to be known, the c_j may be chosen to maximise the probability of the training data. In order to find this maximum, the method of Lagrange multipliers is applied in [53]. Using the constraint equation:

$$\sum_{j=1}^{p} c_j - 1 = 0$$

and the likelihood function:

$$LL \;=\; \frac{1}{N} \cdot \sum_{i=1}^{N} \log \left[\sum_{j=1}^{p} c_j \cdot P(w(i)\,|\,w(i-j)\Big) \right]$$

the function to be maximised is:

$$F(c_1, c_2, \ldots, c_p) \;=\; \frac{1}{N} \cdot \sum_{i=1}^{N} \log \left[\sum_{j=1}^{p} c_j \cdot P\Big(w(i)\,|\,w(i-j)\Big) \right] - \lambda \cdot \left[\sum_{j=1}^{p} c_j - 1 \right]$$

Setting $\dfrac{\partial F(\cdot)}{\partial c_k} = 0$ it may be shown that:

$$c_k \;=\; \frac{1}{N} \cdot \sum_{i=1}^{N} \frac{c_k \cdot P\Big(w(i)\,|\,w(i-k)\Big)}{\sum\limits_{j=1}^{p} c_j \cdot P\Big(w(i)\,|\,w(i-j)\Big)}$$

This equation is used iteratively to determine the memory weights $c_k \quad \forall \quad k \in \{1, 2, \ldots, p\}$, with a uniform set of memory weights constituting a suitable initial condition [53].

Equation (64) was applied to the LOB corpus in [53]. The association probabilities $P(w_i\,|\,w_k)$ were estimated by counting the co-occurrences of words w_i and w_k within a window of length p. Bigrams and trigrams were disallowed in order to eliminate short-term dependencies. To avoid spurious co-occurrences, a minimum threshold of 3 was placed on the word counts of w_i and w_k. The weights c_j were found to be noncritical, and were made constant over the entire window. Various window sizes were investigated, and the optimum was found to lie at approximately $p = 100$. The resulting model was interpolated with a unigram language model, and resulted in a 15% perplexity reduction. However, no significant improvement was achieved when used in conjunction with a bigram language model.

2.6.2. Cache models

Language models usually employ probability estimates that have been chosen to perform well on average over the entire training corpus. This precludes adaptation to dynamic changes in the text characteristics, and therefore such models are described as *static*. The underlying philosophy of a cache is that, due to local text characteristics such as topic and author, words or word patterns that have occurred recently are more likely to recur in the immediate future than a static language model would predict. The cache addresses this by dynamically increasing the probability of such events.

A cache consists of a buffer of the most recent L words of text from which language model probabilities are calculated [39], [45]. The cache language model probabilities P_{cache} are combined with the static model probabilities P_s by linear interpolation:

$$\begin{aligned} P\Big(w(i)\,|\,\mathbf{w}(0,i-1)\Big) \;=\; & (1-\beta_{cache}) \cdot P_s\Big(w(i)\,|\,\mathbf{w}(0,i-1)\Big) \\ & + \beta_{cache} \cdot P_{cache}\Big(w(i)\,|\,\mathbf{w}(0,i-1)\Big) \end{aligned} \tag{66}$$

The interpolation weights β_{cache} are typically chosen to optimise performance on a development test set [24], [29], although deleted interpolation has been employed in [45]. The calculation of P_{cache} is frequently based on unigrams, but bigrams and trigrams have also been used [29], [70], [95]. Taking the unigram cache as an example [45]:

$$P_{cache}\Big(w_k\,|\,\mathbf{w}(0,i-1)\Big) \;=\; \frac{N_{cache}(w_k)}{L} \tag{67}$$

where $N_{cache}(w_k)$ is the number of occurrences of the word w_k in the cache of length L. As a matter of interest, equation (67) may be interpreted as a special case of the word-association model (64) by setting

$p = L,\ c_j = \frac{1}{L}\ \ \forall\ \ j \in \{1, 2, \ldots, p\},$ and choosing the association probabilities $P\left(w_i \,|\, w_k\right)$ to be [53]:

$$P\Big(w(i) \,|\, w(i{-}j)\Big) \;=\; \begin{cases} 1 & \text{if} \quad w(i) = w(i{-}j) \\ 0 & \text{otherwise} \end{cases}$$

When word classifications are available, the discrimination of the cache component may be enhanced by maintaining a separate buffer for each of the word categories. For example, a trigram language model based on part-of-speech categories is used in conjunction with a unigram cache in [45]. Let v_j denote a category hypothesised for $w(i)$, so that the trigram component of the language model computes:

$$P\Big(v_j \,|\, \mathbf{v}(i{-}2, i{-}1)\Big)$$

The probability of the word $w(i)$ given its category v_j is estimated from the relative frequency:

$$P_s\Big(w(i) \,|\, v_j\Big) \;=\; \frac{N\Big(w(i) \,|\, v_j\Big)}{N(v_j)} \tag{68}$$

and a cache is maintained for each individual category:

$$P_{cache(v_j)}\Big(w_k\Big) \;=\; \frac{N_{cache(v_j)}(w_k)}{L_j} \tag{69}$$

where $P_{cache(v_j)}\Big(w_k\Big)$ is the probability estimate from the cache for category v_j, $N_{cache(v_j)}(w_k)$ the number of occurrences of the word w_k in this cache, and L_j the cache size.

The combined probability estimate of $w(i)$, given that it belongs to v_j, is obtained by linear combination of (68) and (69) :

$$P\Big(w(i) \,|\, v_j\Big) \;=\; \beta_j \cdot P_s\Big(w(i) \,|\, v_j\Big) + (1 - \beta_j) \cdot P_{cache(v_j)}\Big(w(i)\Big)$$

The language model probability is calculated using equation (57) :

$$P\Big(w(i) \,|\, \mathbf{v}(i{-}2, i{-}1)\Big) \;=\; \sum_{v : v \in V\left(w(i)\right)} P\Big(w(i) \,|\, v\Big) \cdot P\Big(v \,|\, \mathbf{v}(i{-}2, i{-}1)\Big)$$

Since a cache models local variations in text character, it is usually reset at a point when this is known to change, for instance at article boundaries. Once this has occurred it is necessary to wait until a few entries are present in the cache before sensible probability estimates can be expected (for example, a threshold of 5 is employed in [45]).

In practice, the addition of a cache component to a language model has indeed led to significant perplexity reduction. For the part-of-speech trigram model in [45], the addition of a cache with $L = 200$ leads to a 14% drop in perplexity on the LOB corpus. Tests with word-based *n*-gram models show the addition of a unigram and bigram cache with $L = 1000$ to improve the perplexity by 17% on the WSJ 87-89 corpus [24] and lead to small (2% relative) word error rate improvements [29]. These figures are supported by similar results reported in [70] for tests on the 1994 and 1995 ARPA evaluation tasks [91] [92], where the addition of a unigram and bigram cache was found to lead to consistent improvements of 14% in perplexity and 2% (relative) in word error rate.

2.6.3. Stochastic decision tree based language models

As a fundamental alternative to the *n*-gram history equivalence class defined in equation (46), a stochastic decision tree has been used in [3] to perform the many-to-one mapping $H(w(i)\,)$ described in section 2.2.

A decision tree consists of a hierarchically nested set of binary questions, each associated with one node of the tree [5]. Excepting the leaf (terminal) nodes, each has two descendants, one for each possible outcome of the question. Denote a decision tree $\mathbf{T}$ as a set of T_n nodes, $\mathbf{T} = \{t_0, t_1, \ldots, t_{T_n-1}\}$, where t_0 refers to the root node. Furthermore, denote the l leaves of the tree by the set $\mathbf{L} = \{l_1, l_2, \ldots, l_l\}$, so that $\mathbf{L} \in \mathbf{T}$.

The development in [3] assumes that a set of Q "**predictor variables**" are extracted from the history. Denote these by **y**:

$$\mathbf{y} = \Big\{ y_1, y_2, \ldots, y_Q \Big\}$$

For the particular model studied in [3], these variables were chosen simply to be the most recent Q words, and hence we have:

$$\mathbf{y} = \Big\{ y_1, y_2, \ldots, q_Q \Big\} = \Big\{ w(i{-}Q)\,, w(i{-}Q{+}1)\,, \ldots, w(i{-}1)\, \Big\}$$

Many other choices are possible, however, such as syntactic word category identities and semantic labels for instance. The questions themselves concern the nature of the history $\mathbf{w}(0, i-1)$.

Denote a particular question by the symbol Θ_r, where the index signifies it to belong to node t_r. In [3] the Θ_r were initially assumed to be of the form :

$$\Theta_r : y_i \in \Lambda_r? \tag{70}$$

where Λ_r is a subset of the set of values y_i may take on. When y_i is a word, for example, Λ_r is a subset of the known vocabulary. Thus Θ_r ascertains whether one of the predictor variables is a member of a certain set, and returns either a TRUE or a FALSE result. Denote these two possible outcomes by Θ_r^+ and Θ_r^- respectively.

To obtain the classification of the history $\mathbf{w}(0, i-1)$, the tree is traversed from its root to one of the leaf nodes by answering in succession the question Θ_r posed at each node t_r on this path. The leaves of the tree correspond to the history equivalence classifications $l_j \equiv h_j = H(w(i)\,)$. Each has associated with it an appropriate distribution $P(w_k\,|\,l_j) \quad \forall \quad k \in \{0, 1, \ldots, N_w - 1\}$, reflecting the probability of the next word $w(i)$ given the equivalence class l_j. Note that the number of indicator variables Q may be held large without encountering the combinational explosion experienced with *n*-gram models, since for a given equivalence class generally only a subset of the indicator variables need to be examined.

Starting with only the root node (i.e. $\mathbf{T} = \{t_o\}$ and $\mathbf{L} = \emptyset$), the tree structure is grown incrementally from the training corpus. Growth occurs at the leaves only, non-terminal nodes remaining fixed. For a particular leaf $l_j \equiv t_r$ involves the determination and assignment of a suitable new question Θ_r and the corresponding addition of two descendant nodes (which become the new leaf nodes).

In order to control this process, a criterion indicating how the growth of a terminal node affects the performance of the tree is needed. A common choice, as also employed in [3], is the average entropy of the leaf probability distributions $P(w_k \,|\, l_j)$. Minimising this figure directly minimises the overall uncertainty associated with the prediction of the next word. The entropy at leaf l_j is:

$$\hbar_j\,(W) \;=\; -\sum_{k=0}^{N_w-1} P(w_k \,|\, l_j)\cdot\log\Big(P(w_k \,|\, l_j)\Big)$$

so that the average entropy of the tree is:

$$\bar{\hbar}_{tree}\,(W) \;=\; \sum_{j=1}^{L} \hbar_j\,(W)\cdot P(l_j) \tag{71}$$

where $P\,(l_j)$ is the probability of visiting leaf l_j and the argument W signifies that the entropy is calculated for the word probability distribution.

For questions of type (70), Θ_r is uniquely defined by the choice of predictor variable y_i and the choice of the subset Λ_r. For a certain y_j, the process of finding an optimal Θ_r therefore reduces to the task of determining the optimal partition of the range of y_j into the subset Λ_r and its complement $\bar{\Lambda}_r$. Since the calculation of the globally optimal solution to this problem is extremely computationally expensive, locally optimal greedy hill-climbing algorithms are employed. For example, a variant of k-means clustering is employed in [89]. A considerable fraction of the computation required during tree-growing is devoted to set construction.

Note that both $P(l_j)$ and $P(w_k \,|\, l_j)$ are assumed to be the true probabilities, valid for the language in general. In practice they must be replaced by their estimates, which are derived from the training corpus. Being of limited size, this corpus will never be truly representative of the entire language, and care must therefore be taken to avoid overfitting by the model. This may be accomplished by employing some form of cross-validation, for instance the use of a heldout set [3].

The form of the questions (70) is simple and conveniently illustrates the concept of the tree-based language model, but it may be argued that these are also unacceptably restrictive [3]. If the optimal question were to be of the form "$\Theta_r \,:\, y_i \in \Lambda_i \cap y_j \in \Lambda_j$?", for example, it would be split across two nodes, one with $\Theta_r \,:\, y_i \in \Lambda_i$? and one with $\Theta_r \,:\, y_j \in \Lambda_j$?. This would lead to unnecessarily large trees and fragment the already limited training data.

An alternative form of question employing a directed acyclical graph structure termed a **trellis** is presented in [89]. The trellis structure allows nodes to have more than two children and children to have multiple parents, thus allowing sum-of-products Boolean expressions to be encoded while preserving the use of elementary questions at the nodes themselves.

The tree-based language model has been compared with a trigram language model for a 5,000 word vocabulary task using a 30 million word corpus [3]. The tree-based model contained 10,000 leaves and exhibited a perplexity of 90.7. This was a slight improvement in relation to the trigram, which had a perplexity of 94.9. Although the tree had only 10,015 distinct probability distributions as opposed to the 796,000 of the trigram, the storage required by both was approximately equal since the distributions of the latter generally have only a few nonzero entries while this is not so for the former.

2.7. Domain adaptation

Adaptivity in a language model concerns its capacity to alter the probability estimate $P(w(i)\,|\,\mathbf{w}(0,i-1))$ in accordance with the particular nature of the text. Such text **domains** may be distinguished by attributes such as, for example, the topic of discussion, style of writing, and the time of writing. Cache-based models, as described in section 2.6.2, are an example of how this adaptation may be achieved dynamically. However if we know the domain(s) of application *a-priori*, we may produce a static model already adapted to the particular task.

2.7.1. Specialisation to a target domain

Language models trained on large quantities of text covering many subject areas and styles of writing display good average performance, but are not able to take advantage of the particularities of the domains to which they are applied. Moreover, often there are insufficient data from the specialised domain to allow specialised models to be built directly. In such cases it is desirable to adapt the parameters of a general language model given a relatively small amount of material from the target domain.

Denote the general sample by Σ^G and the sample obtained from the target domain by Σ^D. Let the parameters of the probability distribution defined on the sample space $|\Sigma|$ be χ. Then, starting with the general sample $P(\chi\,|\,\Sigma^G)$ as a prior distribution, we may write the posterior distribution of χ as :

$$P(\chi\,|\,\Sigma^D,\Sigma^G) \quad = \quad \frac{P(\Sigma^D\,|\,\chi,\Sigma^G)\cdot P(\chi\,|\,\Sigma^G)}{\int_\chi P(\Sigma^D\,|\,\chi,\Sigma^G)\cdot P(\chi\,|\,\Sigma^G)}$$

The study in [22] compares the **maximum a-posteriori** (MAP) estimate:

$$\chi^{MAP} \quad = \quad \arg\max_{\chi} P(\chi\,|\,\Sigma^D,\Sigma^G)$$

with the **classical Bayes** estimate:

$$\chi^{Bayes} \quad = \quad \mathcal{E}\Big\{\chi\,|\,\Sigma^D,\Sigma^G\Big\} \quad = \quad \int_\chi \chi\cdot P(\chi\,|\,\Sigma^D,\Sigma^G)\cdot d\chi$$

as competing techniques for adapting the parameters of a probability distribution to new data. Experiments encompass the adaptation of a language model obtained from a 1.9 million word corpus of radiological reports from a particular hospital, to a smaller set of reports obtained from another. It is found that MAP outperforms Bayes adaptation, but also that even better results are obtained from the optimal linear interpolation of language models built using the general and domain-specific samples directly. Word error rate improvements[19] of 5 and 16 % for MAP, and 11% and 25% for optimal interpolation, were achieved after considering 1,000 and 5,000 words of text from the new domain respectively. No figures were given for a single model obtained when pooling the training data from the two samples.

A related approach is taken in [73], where the general language model is adapted to a particular domain using the **minimum discrimination information** method, in which the adapted model is required to be as close as possible to the general distribution in the Kullback-Liebler sense, while satisfying a set of constraints that are imposed by the new domain. These constraints include, for instance, unigram and bigram probabilities. The technique was applied to the Switchboard corpus, and performance was

[19] These reductions are relative to a language model trained exclusively on the general sample.

measured relative to a language model constructed from the general sample only, as well as a model constructed from the pooled data. For these two cases, word error rate improvements of between 4.6 and 6.0% , and between 3.4 and 4.3% were achieved respectively. In contrast with the results reported in [22], similar word error rate improvements were achieved using an optimal linear interpolation between general and domain-specific models.

2.7.2. Mixtures of topic-specific language models

In order to account for a set number of distinct themes appearing in a corpus, the text may be divided into partitions corresponding to common subject matter, termed *topics*, following which a trigram language model is built for each individual topic [28]. The resulting models are then combined linearly to obtain an overall probability estimate:

$$P\Big(w(i)\,|\,\mathbf{w}(i-2,i-1)\Big) \;=\; \sum_{j=1}^{N_{topic}} \lambda_j\cdot P_j\Big(w(i)\,|\,\mathbf{w}(i-2,i-1)\Big) \tag{72}$$

with

$$\sum_{j=1}^{N_{topic}} \lambda_j = 1$$

where N_{topic} is the number of topics, $P_j\,(w(i)\,|\,\mathbf{w}(i-2,i-1))$ the trigram estimate for the j_{th} topic, and λ_j the weighting given to the contribution of the j_{th} topic. Since subdivision of the training corpus may lead to undertrained topic models, equation (72) was linearly interpolated with a **general** trigram $P_G\,(\cdot)$ trained on the entire training set.

$$P(w(i)\,|\,\mathbf{w}(i-2,i-1)\Big)=\sum_{j=1}^{q}\lambda_j\cdot\Big[\theta_j\cdot P_j\Big(w(i)\,|\,\mathbf{w}(i-2,i-1)\Big)\;+\;(1-\theta_j)\cdot P_G\Big(w(i)\,|\,\mathbf{w}(i-2,i-1)\Big)\Big]$$

The division of the corpus into the topic partitions themselves proceeds by agglomerative clustering, assuming each paragraph of text to be concerned with only one particular topic. Assigning each paragraph in the corpus to an individual topic cluster initially, the most similar pairs of clusters are merged successively until the desired number of topics is reached. The normalised number of content words common to both clusters was used as a similarity measure, and further cluster refinement is achieved by reassigning member sentences according to a greedy algorithm so as to maximise the training set probability. Note that the component trigram models $P_j\,(w(i)\,|\,w\,(i-2,i-1))$ must be recomputed after every reassignment using the new partition of the training corpus.

Although in a truly adaptive language model the values of λ_j should adjust to the characteristics of the current text, fixed values were employed and optimal values for λ_j as well as θ_j were obtained using a Viterbi-type reestimation approach [28]. By using a model with 5 topic classes, a 4% relative reduction in overall recogniser error rate was achieved using the NAB1 corpus [29].

2.8. Language models for other applications

Although this thesis focuses on their application to speech-recognition systems, language models are important components also in other fields, some of which are introduced briefly in the following sections.

2.8.1. Character and handwriting recognition

The recognition of printed or handwritten text is a search for the most likely sequence of words with respect to given graphic written evidence, and it is not surprising that techniques very similar to those used by a speech-recogniser may be applied [76], [87]. In particular, instead of an acoustic model, a model for the geometry or trajectory of the writing is employed. Let $\mathbf{x}(0, T-1)$ be a parameterisation of this written evidence, so that we may formulate the recognition problem as the solution of:

$$\underset{w(0),w(1),\ldots,w(K)}{\text{arg max}} \left\{ P\Big(\mathbf{x}(0,T)\,|\,\mathbf{w}(0,K)\Big) \cdot P\Big(\mathbf{w}(0,K)\Big) \right\}$$

Here again the language model $P(\mathbf{w}(0,K))$ supplies an estimate of the grammatical plausibility of the hypothesised sequence of words $\mathbf{w}(0,K)$, and has been shown to have great impact on recognition accuracy [27], [83], [87].

2.8.2. Machine translation

The greater availability of aligned bilingual corpora has prompted renewed research into statistical methods of automatic translation [4], [6], [7], [86]. In particular, the work in [4] and [7] considers the problem of translating a sentence $\mathbf{w}_s(0, N_{sl})$ from the **source language** (French) into a corresponding sentence $\mathbf{w}_t(0, N_{tl})$ in the **target language** (English) according the following probabilistic model:

$$\begin{aligned}\mathbf{w}_t(0,N_{tl}) &= \underset{\forall \tilde{\mathbf{w}}_t(0,N_{tl})}{\text{arg max}} \left[P\Big(\mathbf{w}_s(0,N_{sl}), \tilde{\mathbf{w}}_t(0,N_{tl})\Big)\right] \\ &= \underset{\forall \tilde{\mathbf{w}}_t(0,N_{tl})}{\text{arg max}} \left[P\Big(\mathbf{w}_s(0,N_{sl})\,|\,\tilde{\mathbf{w}}_t(0,N_{tl})\Big) \cdot P\Big(\tilde{\mathbf{w}}_t(0,N_{tl})\Big)\right]\end{aligned}$$

Here $P(\mathbf{w}_s(0,N_{sl}) \mid \tilde{\mathbf{w}}_t(0,N_{tl}))$ is the **translation model**, reflecting the extent to which the English words express the ideas presented in French, and $P(\tilde{\mathbf{w}}_t(0,N_{tl}))$ the English language model, which reflects the extent to which the English hypothesis is grammatical. Within this formalism, the translation model is the analogy of the acoustic model in the speech recognition framework, although the internal mechanisms of each are of course different. Finding the best English translation of the French sentence is a search for the English hypothesis maximising $P(\tilde{\mathbf{w}}_t(0,N_{tl})\,|\,\mathbf{w}_s(0,N_{sl}))$.

Since human translators have been observed to translate almost four times more quickly when dictating than when having to write or type their translations, the integration of such a translation system into a speech-recogniser has been investigated in [9]. Since the text of the source language is already available, and recognition occurs in the target language, the speech-recognition problem is reformulated as:

$$\mathbf{w}_t(0,N_{tl}) = \underset{\forall \tilde{\mathbf{w}}_t(0,N_{tl})}{\text{arg max}} \left[P\Big(\mathbf{x}_t\,(0,T)\,, \mathbf{w}_s(0,N_{sl}), \tilde{\mathbf{w}}_t(0,N_{tl})\Big)\right]$$

where $\mathbf{x}_t$ is the acoustic observation in the target language, obtained from the speech of the dictation.

One may assume that $\mathbf{x}_t$ is independent of the text of the source language, i.e.:

$$P\Big(\mathbf{x}_t \,|\, \mathbf{w}_s(0, N_{sl}), \tilde{\mathbf{w}}_t(0, N_{tl})\Big) \;\approx\; P\Big(\mathbf{x}_t \,|\, \tilde{\mathbf{w}}_t(0, N_{tl})\Big)$$

and hence obtain:

$$\mathbf{w}_t(0, N_{tl}) \;=\; \underset{\forall \tilde{\mathbf{w}}_t(0, N_{tl})}{\text{arg max}} \left[P\Big(\mathbf{x}_t\,(0, T) \,|\, \tilde{\mathbf{w}}_t(0, N_{tl})\Big) \cdot P\Big(\mathbf{w}_s(0, N_{sl}) \,|\, \tilde{\mathbf{w}}_t(0, N_{tl})\Big) \cdot P\Big(\tilde{\mathbf{w}}_t(0, N_{tl})\Big) \right]$$

Results indicate improved recognition performance when the additional knowledge of $\mathbf{w}_s(0, N_{sl})$ incorporated in this way.

2.8.3. Spelling correction

Automatic methods for correcting spelling errors are important in a number of areas, including document preparation, database interaction and text-to-speech systems [46]. Two types of error may be distinguished: firstly **nonword errors**, where the misspelling results in a word no longer valid in the language of interest (for example “than” misspelled as “tahn”), and secondly **real-word errors**, where the error results in a different but valid word with alternate meaning (for example “from” misspelled as “form”). Language models have been applied successfully to the correction of both types of error. In particular, they have led to improvements relative to *isolated-word* methods, which treat the misspelling without taking into account the surrounding words and are hence unable to address real-word errors.

When a nonword error is detected, a number of similar valid spellings are determined, generally by means of a database of valid words or subword units (such as character n-grams) as well as suitable distance measures indicating similarity between strings. Language models have been employed successfully to the ranking of these alternatives, the best then constituting the result of the automatic correction. In the case of real-word errors, the language model was used both to detect the misspelled words by sensing the associated low language model probability, as well as to choose the most likely from the list of subsequently generated alternatives [46].

2.8.4. Tagging

Words may be classified into groups according to their grammatical function (or **part-of-speech**) within the sentence. Such analyses are often important as the first steps in discovering higher-level linguistic structure, for instance the identification of noun-phrases. Since words often have more than one possible part-of-speech assignment (for instance “light”, which may act as adjective, verb or noun), this classification may be ambiguous when considering only the lexical identity of the word. Language models based on part-of-speech information and bigram or trigram dependencies have been used with much success for the automatic annotation of unlabelled text with part-of-speech information, a process referred to as **part-of-speech tagging** [13], [17] ,[20].

Chapter 3

Variable-length category-based *n*-grams

3.1. Introduction and overview

In this chapter we develop a language model designed to capture syntactic patterns in English text. The part-of-speech classification[20] of each word in the training corpus is assumed to be known and constitutes the *a-priori* grammatical information that will be exploited by the statistical model.

The model employs *n*-grams of part-of-speech word-categories to capture sequential grammatical dependencies [56], [57]. Since there are far fewer parts-of-speech than there are words in a typical vocabulary, the number of different *n*-grams is much smaller for a given value of n than for a word-based *n*-gram model. This reduces the problem of data sparseness, and it becomes possible to increase the value of n both from a statistical as well as a storage viewpoint. Furthermore, *n*-grams based on categories are intrinsically able to generalise to word *n*-tuples never witnessed during training. Since these categories embed syntactic information, this generalisation proceeds according to the measure of grammatical correctness assigned to the unseen sequence by the model. Finally, the length of each individual *n*-gram is optimised by allowing it to increase to the point at which no further improvement in predictive capability is detected, thereby allowing model size to be traded for model performance in a well-defined way.

3.2. Structure of the language model

Drawing on the exposition of section 2.5.1, denote the N_v different part-of-speech categories by:

$$\mathbf{V}=\left\{v_0, v_1, \ldots v_{N_v-1}\right\}$$

Because a word may have multiple grammatical functions, the mapping operator $V(\cdot)$ is one-to-many and hence $V(w_i)$ is the set of possible part-of-speech classifications for w_i. With reference to the definitions in section 2.2, we now define a history equivalence class to be an *n*-gram of categories. Let there be N_H such *n*-grams, denoting these collectively by $\mathbf{H}$:

$$\mathbf{H}=\left\{h_0, h_1, \ldots, h_{N_H-1}\right\}$$

Let the length of the category *n*-gram associated with the particular history equivalence class h_i be given

[20] This work employs the definitions used for the LOB corpus, a listing of which appears in appendix E.

by $L_H(h_i)$, so that the *n*-gram itself may be denoted by:

$$h_i \;\equiv\; \mathbf{v}_{h_i}\Big(0, L_H(h_i) - 1\Big) \tag{73}$$

Finally, since $V(\,\cdot\,)$ is one-to-many, a particular category *n*-gram may describe multiple word *n*-grams. Hence the mapping $H(\,\cdot\,)$ is many-to-many, returning for each particular word sequence $\mathbf{w}(0, i-1)$ the set of history equivalence classes $H(\mathbf{w}(0, i-1)\,)$ for which the category and word *n*-grams match:

$$H(\mathbf{w}(0,i-1)\,) \;=\; \Big\{ h_a : v_{h_a}(j) \in V\Big(w(j+i-L_H(h_a))\Big) \,,\; a \in \mathcal{R}_H,\; j \in \mathcal{R}_{L_a},\; i \geq L_H(h_a) \Big\} \tag{74}$$

where $\mathcal{R}_H = \{0, 1, \ldots, N_H - 1\}$ and $\mathcal{R}_{L_a} = \{0, 1, \ldots, L_H(h_a) - 1\}$. Assume as in equation (52) that the word occurrence probability depends only on its believed category, and hence employ (53) to write:

$$P\Big(w(i)\,|\,\mathbf{w}(0, i-1)\Big) \;\approx\; \sum_{\forall v : v \in V\left(w(i)\right)} P\Big(w(i)\,|\,v\Big) \cdot P\Big(v\,|\,\mathbf{w}(0, i-1)\,\Big) \tag{75}$$

Furthermore, employing the decomposition (54) and the *n*-gram assumption (73), we may decompose the second term on the right hand side of equation (75) as follows:

$$P\Big(v_j\,|\,\mathbf{w}(0, i-1)\Big) \;\approx\; \sum_{\forall h : h \in H\left(w(i)\right)} P(v_j\,|\,h) \cdot P\Big(h\,|\,\mathbf{w}(0, i-1)\,\Big) \tag{76}$$

where h is the category *n*-gram context. The summation in (75) accounts for all part-of-speech categories that may be assigned to $w(i)$, and the summation in (76) for all the history equivalence classes matching the word history $\mathbf{w}(0, i-1)$. The interrelation of the probability functions in these equations is illustrated in figure 3.1, and the subsequent sections treat the estimation of each individually.

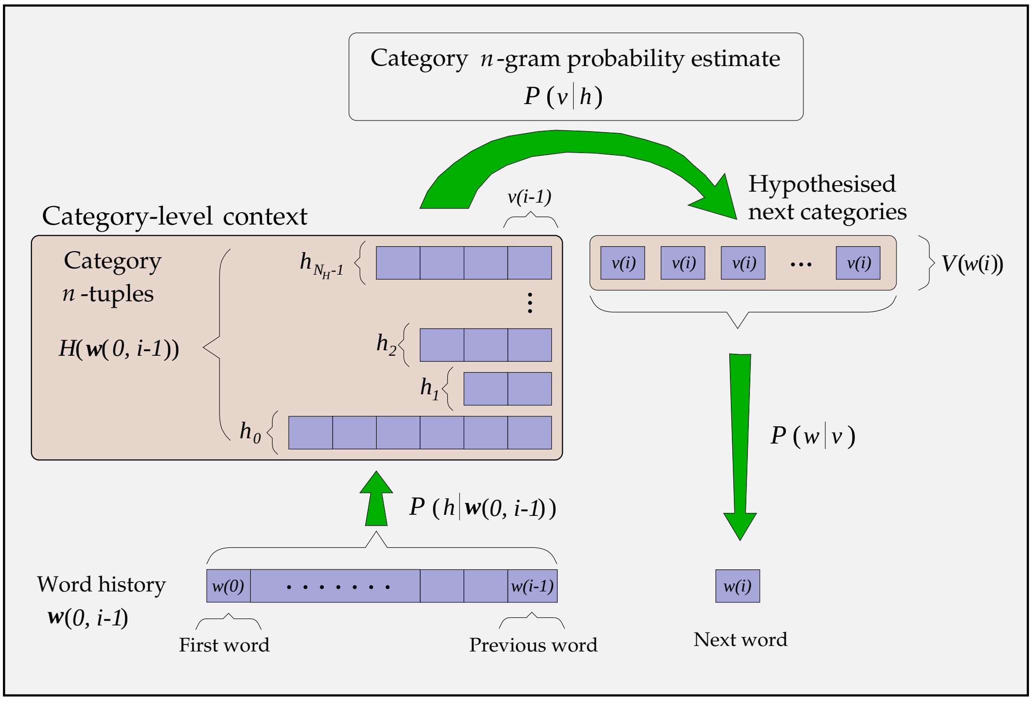

Figure 3.1: Operation of the category-based language model.

3.2.1. Estimating $P(v_j|h_m)$

For compact storage of category *n*-grams a tree data structure is employed, associating each node with a particular word category so that paths originating at the root correspond to category *n*-grams. In this way each node represents a distinct history equivalence class h_m, and has associated with it a conditional probability distribution function $P\,(v|h_m)$, while the set of all nodes corresponds to the set of all history equivalence classes $\mathbf{H}$. By not restricting the length of the individual paths through the tree, contexts of arbitrary depth are catered for. The following figure illustrates this structure by means of an example. Nodes are labelled both with the specific history equivalence class h_m they represent, as well as the category defining the *n*-gram with respect to the parent node. In particular, for this example, the history equivalence class h_1 corresponds to the bigram context $\mathbf{v}(i-1, i-1) = \{v_1\}$ and the history equivalence class h_5 to the trigram context $\mathbf{v}(i-2, i-1) = \{v_2, v_8\}$.

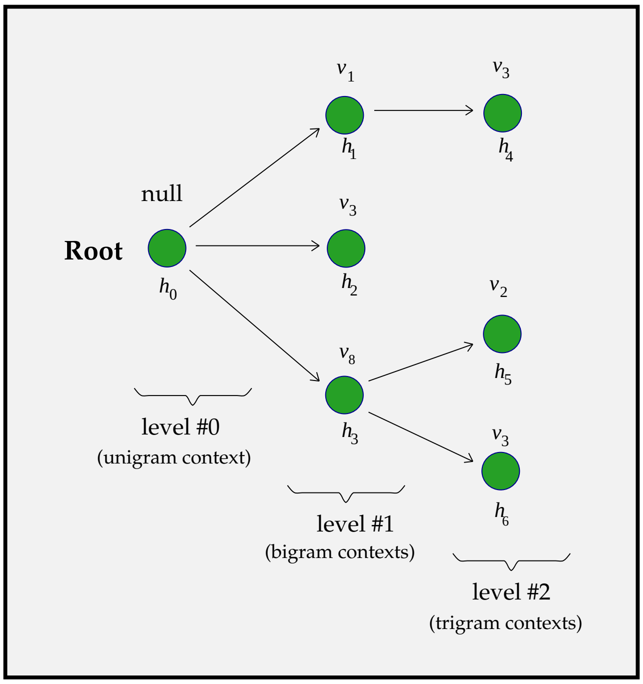

Figure 3.2: Organisation of the category *n*-gram tree.

Probabilities of the form $P(v(i)\,|\,\mathbf{v}(0, i-1)\,)$ are calculated from the tree by first determining the history equivalence class corresponding to the category context $\mathbf{v}(0, i-1)$ and then applying an *n*-gram probability estimator. In particular, define the operator $\mathcal{F}_t(\,\cdot\,)$ which maps category *n*-grams to history equivalence classes, i.e.:

$$h_m = \mathcal{F}_t(\mathbf{v}(0, i-1)\,) \tag{77}$$

where h_m is the history equivalence class corresponding to the deepest match of the category sequence $\mathbf{v}(0, i-1)$ within the *n*-gram tree. The probabilities $P(v|h_m)$ are estimated by application of Katz's back-off in conjunction with nonlinear discounting according to equations (48), (49) and (50). Model construction proceeds via the following level-by-level tree growing strategy, which retains only contexts that improve a performance quality estimate. This allows model compactness to be maintained while employing longer *n*-grams where they benefit performance.

1. **Initialisation** : $L = -1$
2. $L = L + 1$
3. **Grow** : Add level $\#L$ to level $\#(L-1)$ by adding all the $(L+1)$-grams occurring in the training set for which the L-grams already exist in the tree.
4. **Prune** : For every (newly created) leaf in level $\#L$, apply a quality criterion and discard the leaf if it fails.
5. **Termination** : If there are a nonzero number of leaves remaining in level $\#L$, goto step 2.

The quality criterion is based on the training set probability delivered by the model to reflect its predictive performance. In order to avoid overfitting, leaving-one-out cross validation is employed in calculating this probability [19]. For the category *n*-gram model, this method has the particular advantage of being computationally efficient since the *n*-gram counts are available in memory. In particular, referring to appendix B, the leaving-one-out log probability may be expressed as:

$$LL_{\text{cum}}(\Omega^{tot}) \;=\; \sum_{i=0}^{N_c-1} \log\left[P\Big(v(i) \mid \mathbf{v}(0,i-1)\,,\Omega_i^{\text{RT}}\Big)\right]$$

where N_c is the number of words in the training corpus Ω^{tot}, and $P(v(i) \mid \mathbf{v}(0,i-1)\,,\Omega_i^{\text{RT}})$ is the probability estimated by the *n*-gram model obtained from Ω^{RT}, the retained-part formed by removal of the heldout part Ω^{HO} from Ω^{tot}. Using definition (77), this log probability may be rewritten as the sum of contributions of each node:

$$\begin{aligned} LL_{\text{cum}}\Big(\Omega^{\text{tot}}\Big) &= \sum_{n=0}^{N_H-1}\left(\sum_{v(j):\mathcal{F}_t\big(\mathbf{v}(0,j-1)\big)=h_n} \log\left[P\Big(v(j) \mid h_n, \Omega_j^{\text{RT}}\Big)\right]\right) \\ &= \sum_{n=0}^{N_H-1} LL_{\text{cum}}^{h_n} \end{aligned}$$

where

$$LL_{\text{cum}}^{h_n} \;=\; \sum_{k=0}^{N_v-1} N_{h_n}(v_k)\cdot\log\left[P\left(v_k \mid h_n, \Omega_k^{\text{RT}}\right)\right]$$

and where LL_{cum} is the log probability of the entire training corpus Ω^{tot}, $LL_{\text{cum}}^{h_n}$ the log probability of all events occurring in context h_n, $N_{h_n}(v_k)$ the total number of times v_k was seen in context h_n in Ω^{tot}, and $P\left(v_k|h_n,\Omega_k^{\text{RT}}\right)$ the probability of v_k occurring in context h_n based on the retained part of the training set Ω_k^{RT} formed when v_k constitutes the heldout part.

Now assume that node h_n is a leaf, and that the change in training set log probability resulting from the addition of a child $h_{n+\epsilon}$ must be calculated. While h_n refers to the original parent node, $\acute{h}_n$ is used to denote this node after the addition of the child. The change in log probability is then given by:

$$\Delta LL_{\text{cum}}^{h_n} \;=\; LL_{\text{cum}}^{\acute{h}_n} + LL_{\text{cum}}^{h_{n+\epsilon}} - LL_{\text{cum}}^{h_n}$$

In terms of these quantities, the pruning criterion is:

$$\Delta LL_{\text{cum}}^{h_n} > -\lambda_{ct} \cdot LL_{\text{cum}} \left(\Omega^{\text{tot}}\right) \tag{78}$$

This requires the new node to lead to an increase of at least a threshold defined as a fraction λ_{ct} of the total log probability, so as to make the choice of the threshold fairly problem independent.

The probability $P(v_j|h_m)$ may be used to calculate a perplexity indicating the confidence with which the tree predicts the following category. This is used later in language model evaluation, and will be referred to as the **category perplexity**.

3.2.2. Estimating $P(w_i|v_j)$

Assuming each category to have a sufficiently large membership, we apply the relative frequency:

$$P(w_i|v_j) = \frac{N(w_i|v_j)}{N(v_j)}$$

Since the language model must hypothesise categories for out-of-vocabulary (OOV) words, the probability with which these occur within each category must be estimated. Accordingly a word named[21] "UW" is added to each category, and its count N_{uw} estimated by the leaving-one-out method:

$$P(\texttt{UW}|v_j) = \frac{N_1(v_j)}{N(v_j) + \eta} \tag{79}$$

and

$$N_{uw}(v_j) = \frac{P(\texttt{UW}|v_j) \cdot N(v_j)}{1 - P(\texttt{UW}|v_j)}$$

where $N_1(v_j)$ is the number of words seen exactly once in both v_j and the training set, $N(v_j)$ the total number of words in v_j, $N_{uw}(v_j)$ the estimated count for UW in v_j. Finally, $\eta > 0$ is a small constant introduced heuristically to ensure that the denominator of (79) always exceeds the numerator. Its effect is significant only for sparsely trained categories with consequently small $N(v_j)$. The effect of η on performance was seen empirically to be weak, and $\eta \approx 5 \ldots 10$ yields satisfactory results for the LOB corpus. More complete details are given in appendix C.

3.2.3. Estimating $P(h_m\,|\,\mathbf{w}(0, i-1))$

This language model component estimates the probability that a particular word history $\mathbf{w}(0, i-1)$ corresponds to the category *n*-gram context associated with h_m. Since a word may have multiple part-of-speech classifications, there are in general many possible contexts to which $\mathbf{w}(0, i-1)$ could belong. The set of such contexts as well as the probabilities associated with each may be calculated using a recursive approach which we will develop by first assuming these contexts and probabilities to be known for $\mathbf{w}(0, i-1)$, and then deriving the corresponding results for $\mathbf{w}(0, i)$. To begin, we define:

$\mathbf{v}_j^{hyp}(a, b)$: A possible category sequence for the words $\mathbf{w}(a, b)$, termed a **hypothesis** hereafter, individual hypotheses being distinguished by the index j.

$N_{hyp}(a, b)$: The number of hypotheses for the word sequence $\mathbf{w}(a, b)$.

[21] The string "UW" is an abbreviation for "unknown word". Any other word that is not in the vocabulary could of course be used instead.

For each possible hypothesis we require an indication of the likelihood that it is the correct classification of the word string. Denote this probability by $P(\mathbf{v}_j^{hyp}(a,b)\,|\,\mathbf{w}(a,b)\,)$, so that:

$$\sum_{j=0}^{N_{hyp}(a,b)-1} P\Big(\mathbf{v}_j^{hyp}(a,b)\,|\,\mathbf{w}(a,b)\Big) \;=\; 1 \tag{80}$$

In the following we determine expressions for $P(\mathbf{v}_j^{hyp}(0,i)\,|\,\mathbf{w}(0,i)\,)$, from which the desired probability of the history equivalence class may be obtained as follows:

$$P\Big(h_m\,|\,\mathbf{w}(0,i)\Big) \;=\; \sum_{\forall j:\mathcal{F}_t\left(\mathbf{v}_j^{hyp}(0,i)\right)=h_m} P\Big(\mathbf{v}_j^{hyp}(0,i)\,|\,\mathbf{w}(0,i)\Big) \tag{81}$$

Explicit maintenance of the hypotheses is necessary (as opposed to simply keeping a record of the history equivalence classes) due to the varying lengths of the *n*-grams. In particular, it may occur that:

$$L_H(\mathcal{F}_t(\mathbf{v}(0,i{-}1)\,))\;<\;L_H(\mathcal{F}_t(\mathbf{v}(0,i)\,))-1$$

in which case the *n*-gram probability estimate based on $\mathbf{v}(0,i)$ makes use of more contextual information than is implicit in the history equivalence class $\mathcal{F}_t(\mathbf{v}(0,i{-}1)\,)$. In practice it is only necessary to maintain a set of hypotheses $\mathbf{v}(i-D,i-1)$ of depth D such that D equals or exceeds the maximum length of any *n*-gram stored in the tree. This guarantees that the hypotheses are always at least as deep as any path through the tree. Hypotheses arising during the recursive calculations that differ only in elements $v(i-j)$ for $j>D$, may be merged by summing their probabilities.

Given a set of existing hypotheses $\{\mathbf{v}_j^{hyp}(0,i{-}1)\}$, the set of new hypotheses is $\{\mathbf{v}_j^{hyp}(0,i{-}1)\,,v_k\}$ for all (j,k) such that $j=\{0,1,\ldots,N_{hyp}(0,i{-}1)-1\}$ and $k=\{0,1,\ldots,N_v{-}1\}$ where N_v is the number of different categories. Consider now the particular postulate $\mathbf{v}_{j'}^{hyp}(0,i) \;=\; \{\mathbf{v}_j^{hyp}(0,i{-}1)\,,v_k\}$, the prime over the index indicating that there is in general no fixed relation between the ordering of the two sets of hypotheses. Using Bayes rule, we may write:

$$\begin{aligned}P\left(\mathbf{v}_{j'}^{hyp}(0,i)\,|\,\mathbf{w}(0,i)\right) &= \frac{P\left(\mathbf{w}(0,i)\,|\,\mathbf{v}_{j'}^{hyp}(0,i)\right)\cdot P\left(\mathbf{v}_{j'}^{hyp}(0,i)\right)}{P\Big(\mathbf{w}(0,i)\Big)}\\ &= \frac{P\left(\mathbf{w}(0,i)\,,\mathbf{v}_{j'}^{hyp}(0,i)\right)}{P\Big(\mathbf{w}(0,i)\Big)}\end{aligned} \tag{82}$$

but, recalling assumption (52) it follows that:

$$\begin{aligned}P\left(\mathbf{w}(0,i)\mid\mathbf{v}_{j'}^{hyp}(0,i)\right) &= \prod_{k=0}^{i} P\left(w(k)\mid v_{j'}^{hyp}(k)\right)\\ &= P\left(w(i)\,|\,v_{j'}^{hyp}(i)\right)\cdot P\left(\mathbf{w}(0,i{-}1)\mid\mathbf{v}_{j'}^{hyp}(0,i{-}1)\right)\end{aligned} \tag{83}$$

and applying equation (76) as well as the *n*-gram assumption (73), we may write:

$$\begin{aligned} P\left(\mathbf{v}_{j'}^{hyp}(0,i)\right) &= \prod_{k=0}^{i} P\left(v_{j'}^{hyp}(k) \,|\, \mathcal{F}_t(\mathbf{v}_{j'}^{hyp}(0,k\!-\!1)\,)\right) \\ &= P\left(v_{j'}^{hyp}(i) \,|\, \mathcal{F}_t(\mathbf{v}_{j'}^{hyp}(0,i\!-\!1)\,)\right) \cdot P\left(\mathbf{v}_{j'}^{hyp}(0,i\!-\!1)\right) \qquad (84) \end{aligned}$$

where $\mathbf{v}_{j'}^{hyp}(0,–1)$ is the single initial empty hypothesis and $\mathcal{F}_t(\mathbf{v}_{j'}^{hyp}(0,–1)\,)$ the associated unigram context, so that $P\left(\mathbf{v}_{j'}^{hyp}(0,–1)\right) = 1$. From (82), (83) and (84) it follows that:

$$\begin{aligned} P\left(\mathbf{w}(0,i)\,,\mathbf{v}_{j'}^{hyp}(0,i)\right) &= P\left(w(i)\,|v_{j'}^{hyp}(i)\right) \cdot P\left(v_{j'}^{hyp}(i) \,|\, \mathcal{F}_t(\mathbf{v}_{j'}^{hyp}(0,i\!-\!1)\,)\right) \cdot \\ &\quad\; P\left(\mathbf{w}(0,i\!-\!1)\,,\mathbf{v}_{j'}^{hyp}(0,i\!-\!1)\right) \qquad (85) \end{aligned}$$

Finally, note that:

$$P\Big(\mathbf{w}(0,i)\Big) \;=\; \sum_{j'=0}^{N_{hyp}(0,i)} P\left(\mathbf{w}(0,i)\,,\mathbf{v}_{j'}^{hyp}(0,i)\right) \qquad (86)$$

At any given instant, the most likely postulate is that for which $P\left(\mathbf{v}_{j'}^{hyp}(0,i)\,|\mathbf{w}(0,i)\right)$ is a maximum. Since the number of possible hypotheses becomes extremely large as i increases, it is necessary to restrict storage to the N_{hyp}^{max} most likely candidates by choosing those for which this probability is greatest. This implies that valid hypotheses may be discarded, and hence equation (80) will no longer be satisfied:

$$\sum_{q=0}^{N_{hyp}^{max}} P\left(\mathbf{v}_{q}^{hyp}(0,i)\,|\,\mathbf{w}(0,i)\right) \;<\; 1 \quad \text{when} \quad N_{hyp}^{max} < N_{hyp}(0,i)$$

and where $\mathbf{v}_q^{hyp}(0,i)$ is taken in this case to refer to the q_{th} most likely hypothesis. However equation (76) requires these probabilities to sum to unity. By replacing equation (86) with:

$$P\Big(\mathbf{w}(0,i)\Big) \;\overset{def}{=}\; \sum_{q=0}^{N_{hyp}^{max}} P\left(\mathbf{w}(0,i)\,,\mathbf{v}_{q}^{hyp}(0,i)\right) \qquad (87)$$

these conditional probabilities are renormalised on application of equation (82). In effect the probability mass associated with the discarded hypotheses is distributed proportionally among those which are retained. Note that, since according to equation (82) the quantity $P\left(\mathbf{w}(0,i)\right)$ is common to all new hypotheses, the choice of the N_{hyp}^{max} best candidates can be made by considering the joint probabilities $P\left(\mathbf{w}(0,i)\,,\mathbf{v}_{j'}^{hyp}(0,i)\right)$ instead of the conditional probabilities $P\left(\mathbf{v}_{j'}^{hyp}(0,i) \mid \mathbf{w}(0,i)\right)$.

The complete recursive procedure is summarised in the following. It is assumed that the set of $N_{hyp}^{old} \le N_{hyp}^{max}$ best previous hypotheses $\mathbf{v}_j^{hyp}(0,i\!-\!1)$ as well as the corresponding probabilities $P\left(\mathbf{w}(0,i)\,,\mathbf{v}_j^{hyp}(0,i\!-\!1)\right)$ are available in arrays collectively referred to as $\mathbf{H}^{old}$. Similarly the set of $N_{hyp}^{new} \le N_{hyp}^{max}$ updated context hypotheses with their corresponding probabilities $P\left(\mathbf{w}(0,i)\,,\mathbf{v}_j^{hyp}\,(0,i)\right)$ and $P\left(\mathbf{v}^{hyp}(0,i)\,|\,\mathbf{w}(0,i)\right)$ will be stored in $\mathbf{H}^{new}$. Initialisation is accomplished by setting $i = -1$ and placing a single empty hypothesis in $\mathbf{H}^{new}$, i.e. $v_0^{hyp} = \{\}$ and $N_{hyp}^{new} = 1$.

```
1.  Copy all $\mathbf{v}^{hyp}(0,i)$ and corresponding $P\Big(\mathbf{w}(0,i)\,,\mathbf{v}^{hyp}(0,i)\Big)$ in $\mathbf{H}^{new}$ to $\mathbf{H}^{old}$.
2.  Clear $\mathbf{H}^{new}$.
3.  $i = i + 1$.
4.  For each hypothesis $\mathbf{v}_j^{hyp}$ in $\mathbf{H}^{old}$ where $j = \Big\{0, 1, \ldots, N_{hyp}^{old} - 1\Big\}$ :
5.      For each category $v_k$ such that $v_k \in V(w(i)\,)$:
6.          $\mathbf{v}^{hyp}(0,i) = \Big\{\mathbf{v}_j^{hyp}, v_k\Big\}$.
7.          Calculate $P\Big(\mathbf{w}(0,i)\,,\mathbf{v}^{hyp}(0,i)\Big)$ using (85) .
8.          If $P\Big(\mathbf{w}(0,i)\,,\mathbf{v}^{hyp}(0,i)\Big)$ exceeds an entry in $\mathbf{H}^{new}$, insert $P\Big(\mathbf{w}(0,i)\,,\mathbf{v}^{hyp}(0,i)\Big)$
            and $\mathbf{v}^{hyp}(0,i)$ into $\mathbf{H}^{new}$, possibly overwriting the smallest entry in the process.
9.  Calculate $P(\mathbf{w}(0,i))$ using (87) .
10. Calculate $P\Big(\mathbf{v}^{hyp}(0,i)\,|\,\mathbf{w}(0,i)\Big)$ for each $q = \Big\{0, 1, \ldots, N_{hyp}^{new} - 1\Big\}$ in $\mathbf{H}^{new}$ using (82) .
11. $\mathbf{H}^{new}$ now contains the set of best new hypotheses as well as the corresponding probabilities
    $P\Big(\mathbf{w}(0,i)\,,\mathbf{v}^{hyp}(0,i)\Big)$ and $P\Big(\mathbf{v}^{hyp}(0,i)\,|\,\mathbf{w}(0,i)\Big)$ . Use (81) to calculate $P\left(h_m \mid \mathbf{w}(0,i)\right)$.
```

3.2.4. Beam-pruning

The procedure described in the previous section maintains a fixed maximum number of hypotheses for the word history $\mathbf{w}(0,i)$. Often many of these have very low associated $P\left(\mathbf{v}_q^{hyp}(0,i)\,|\,\mathbf{w}(0,i)\right)$, and by discarding such unlikely hypotheses, computational efficiency may be improved considerably. Beam-pruning maintains only those hypotheses with associated probabilities that are at least a certain fraction of the most likely hypothesis. In particular, letting $P_{\mathbf{w},\mathbf{v}}^{max}\,(i)$ denote the maximum $P\Big(\mathbf{w}(0,i)\,,\mathbf{v}^{hyp}(0,i)\Big)$ entry in $\mathbf{H}^{new}$, this condition is met when:

$$P\Big(\mathbf{w}(0,i)\,,\mathbf{v}^{hyp}(0,i)\Big) \;\geq\; \delta \cdot P_{\mathbf{w},\mathbf{v}}^{max}\,(i) \tag{88}$$

In practice this means that step 8 must be reformulated as follows:

```
8a. If (88) is satisfied by the new hypothesis, insert $P\Big(\mathbf{w}(0,i)\,,\mathbf{v}^{hyp}(0,i)\Big)$ and $\mathbf{v}^{hyp}(0,i)$ into $\mathbf{H}^{new}$,
    possibly overwriting the smallest entry in the process.
8b. remove from $\mathbf{H}^{new}$ any hypothesis for which (88) fails.
```

The second step discards hypotheses which have moved outside the beam. Incorporation of the beam-pruning allows accuracy to be traded for computational efficiency, an issue that is particularly important when the language model is used to tag large quantities of text.

3.2.5. Employing the language model as a tagger

A knowledge of the correct category assignment for each word in the training corpus is assumed when constructing the language model. Since this information is not normally available, an automatic means of annotating large volumes of text with grammatical word classifications is sought.

Assigning the most likely category to each word in a sentence is a process referred to as **tagging**[22]. Denoting the sentence by $\mathbf{w}(0, N-1)$, this corresponds to finding the sequence $\mathbf{v}(0, N-1)$ for which the probability

$$P\Big(\mathbf{v}(0, N-1) \mid \mathbf{w}(0, N-1)\Big)$$

is a maximum. Recall now that a list of these probabilities as well as the corresponding category assignments is maintained by the procedure described in section 3.2.3, and hence the calculation of the probability $P\left(h \mid \mathbf{w}(0,i)\right)$ implicitly involves a tagging operation. In particular, it maintains a list of the most likely category sequences for the words $\mathbf{w}(0,i)$ with respect to the language model statistics.

With reference to the procedure on page 47, sentences may be tagged one at a time as follows:

1. Initialise $\mathbf{H}^{new}$ and set $i = -1$.
2. Execute steps 1 - 11 for each word in the current sentence in turn.
3. The hypothesis $\mathbf{v}(0, N-1)$ with the highest $P\Big(\mathbf{v}(0, N-1) \mid \mathbf{w}(0, N-1)\Big)$ is the most likely sequence of tags for the sentence.

Using this technique, a language model constructed from a tagged corpus may be used to assign tags to an untagged corpus, making it possible to build further language models from the result. Since the untagged corpus may contain many millions of words, the beam-pruning technique is of particular importance during tagging.

3.3. Performance evaluation

In the following we evaluate the language model construction and application techniques described in the first part of this chapter. Experiments are carried out for the LOB, Switchboard and WSJ corpora, and because the last two do not contain part-of-speech information, application of the language model as a tagger is treated first. Details regarding the corpora and their division into test and training sets is presented in appendix D.

3.3.1. Tagging accuracy

A category-based language model built using the LOB training set was used to tag the test set by means of the procedure described in section 3.2.5, and the result compared with the actual tags in order to determine the tagging accuracy. As a benchmark, the same experiment was carried out using the ACQUILEX tagger [20], and are shown in the following table.

	ACQUILEX	**Category model**
Overall tagging accuracy	94.03%	95.13%
Tagging accuracy of known words	95.77%	96.31%
Tagging accuracy of OOV words (2.51%)	31.17%	49.30%

Table 3.1: Tagging accuracies for the LOB test set.

[22] The particular category assignment for each word is referred to as the **tag**.

When tagging words falling within the vocabulary defined by the training corpus, the language model achieves a 13% reduction in tagging error over the benchmark, and when tagging OOV words, this figure rises to 26% . The improvements are attributed to the longer *n*-gram contexts used by the category model, as well as the method used to calculate probabilities for unknown words, described briefly in section 3.2.2 and in more detail in appendix C.

3.3.1.1. Lexicon augmentation

The results in table 3.1 show the tagging accuracy for OOV words to be significantly lower than for words seen during training, and for this reason the effect of employing various additional information sources to augment the lexicon was investigated. In particular, the following sources were used:

1. Word spellings and part-of-speech assignments from the Oxford Advanced Learner's Dictionary (available electronically [50]). The mapping used to convert the dictionary's grammatical classifications to those employed by the language model is described in appendix F.
2. A list of 5000 frequent names and surnames. These were included since OOV words were seen to include a high proportion of proper nouns (approximately 70%).
3. Genitive cases of words already present in the lexicon only in their standard form.

The following table shows tagging accuracies when using the augmented lexicon. Note that the OOV rate has more than halved, and that the overall tagging error has been reduced by 14.3% .

	No augmentation	**With augmentation**
OOV rate	2.51%	1.05%
Overall tagging accuracy	95.13%	95.82%
Tagging accuracy of known words	96.31%	96.26%
Tagging accuracy of OOV words	49.30%	54.55%

Table 3.2: Tagging accuracies with augmented lexicon.

3.3.1.2. Beam pruning

Since the text corpora that are to be tagged may be very large, the required computational effort is an important practical issue, since it determines the required processing time. Figure 3.3 shows how the tagging rate and accuracy vary as functions of the beam pruning parameter. Note that tagging may be accelerated by a factor of 2-3 with only a slight decrease in accuracy.

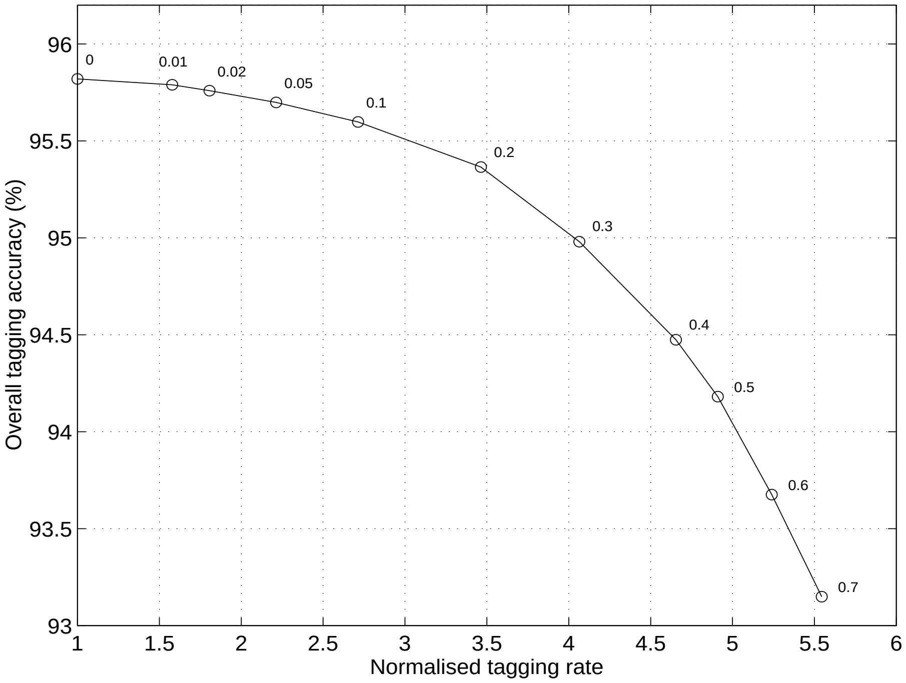

Figure 3.3: Tagging rate and accuracy as a function of the beam pruning parameter.

3.3.1.3. Tagging the Switchboard and WSJ corpora

Since the Switchboard and Wall Street Journal (WSJ) corpora do not contain part-of-speech classifications, they must be tagged before category-based models can be constructed. To accomplish this, a language model was built using the entire LOB corpus, after which its lexicon was augmented as in section 3.3.1.1. Finally, the most frequent OOV words in the respective corpora were determined, tagged by hand, and also added to the lexicon. Table 3.3 summarises this process.

	Switchboard	**WSJ**
Words in LOB corpus lexicon	42,258	42,258
Words added from dictionary	34,509	34,509
Additional proper nouns	5,037	5,037
Hand-tagged OOV words	452	1,888
Genitives	46,209	46,061
TOTAL	128,465	129,753
OOV-rate on training corpus	0.26%	1.19%

Table 3.3: Constitution of the augmented lexica for tagging Switchbaord and WSJ.

The category-based language model was used in conjunction with the corresponding augmented lexicon to produce tagged versions of both the Switchboard and the WSJ corpora.

3.3.2. Constructing category trees

Using the procedure detailed in section 3.2.1, category *n*-gram language model trees were constructed for the LOB, Switchboard and WSJ corpora. The proposed tree pruning technique based on the leaving-one-out probability, termed **likelihood pruning** in the following, was evaluated by comparing it with two others: *context-count pruning* and *n-gram-count pruning*. The first discards all *n*-grams for which the $(n-1)$-gram context has been seen fewer than a threshold number of times. The second eliminates all *n*-grams in the tree which have themselves been seen less often than a threshold, regarding them as part of the unseen mass in probability calculations. Note that the latter approach is often used to make word-based *n*-gram models more compact.

To illustrate the models obtained for different pruning thresholds, figure 3.4 graphs model size (total number of *n*-grams in the tree) against the resulting category-perplexity (introduced at the end of section 3.2.1) for the LOB corpus. Models obtained by *n*-gram-count pruning are not shown here since their perplexities are off-scale, lying above 17.5.

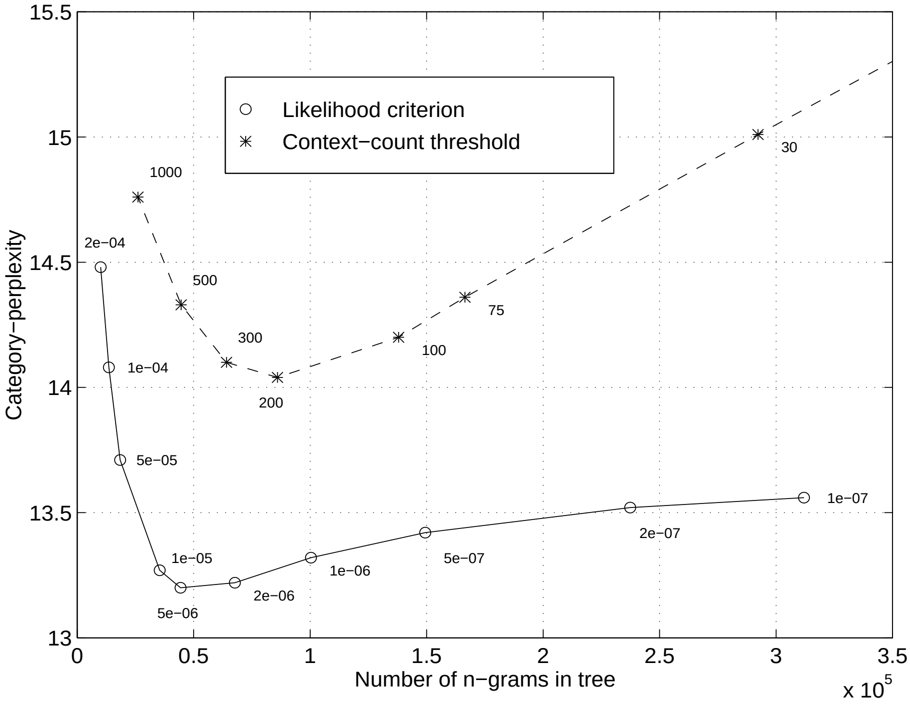

Figure 3.4: Model size versus performance for context-count and likelihood pruning on the LOB corpus.

From figure 3.4 we see that, as the complexity of the tree increases, the test set perplexity moves through a minimum. The initial decrease may be ascribed to underfitting of the data due to an insufficient number of parameters, and the subsequent increase to overfitting of the data due to an excessive number of parameters in the language model tree. Although its effect has been significantly reduced in comparison with context-count pruning, overfitting occurs despite the use of leaving-one-out cross-validation, which remains an approximate way of modelling the test set. For all values of the pruning threshold, trees constructed using likelihood-pruning exhibit a better size versus performance characteristic than those obtained by thresholding context counts. Similar behaviour has been seen for the Switchboard and Wall Street Journal corpora, except that in the last case the perplexity increase due to overfitting is smaller, an effect which is ascribed to the larger amount of training material.

In order to determine how longer *n*-grams affect model performance, the category perplexity was measured while limiting the maximum *n*-gram length in the tree to various values[23]. In particular, the trees obtained from the LOB training set by likelihood pruning (threshold $\lambda_{ct} = 5e - 6$), context-count pruning (threshold 200) and *n*-gram-count pruning (threshold 1) were used for these experiments, and the results graphed in figure 3.5. We conclude that context-count pruning performs better than *n*-gram-count pruning[24], although the difference is negligible for $n < 4$. This is due to the increasing sparseness of *n*-grams and their contexts at higher n. In particular, individual *n*-grams occurring more often than the threshold will cause *n*-gram-pruning to maintain a context which may deliver bad generalisation due to a high proportion of unseen events in the test set. However, pruning based on the log probability outperforms both approaches based on count thresholds.

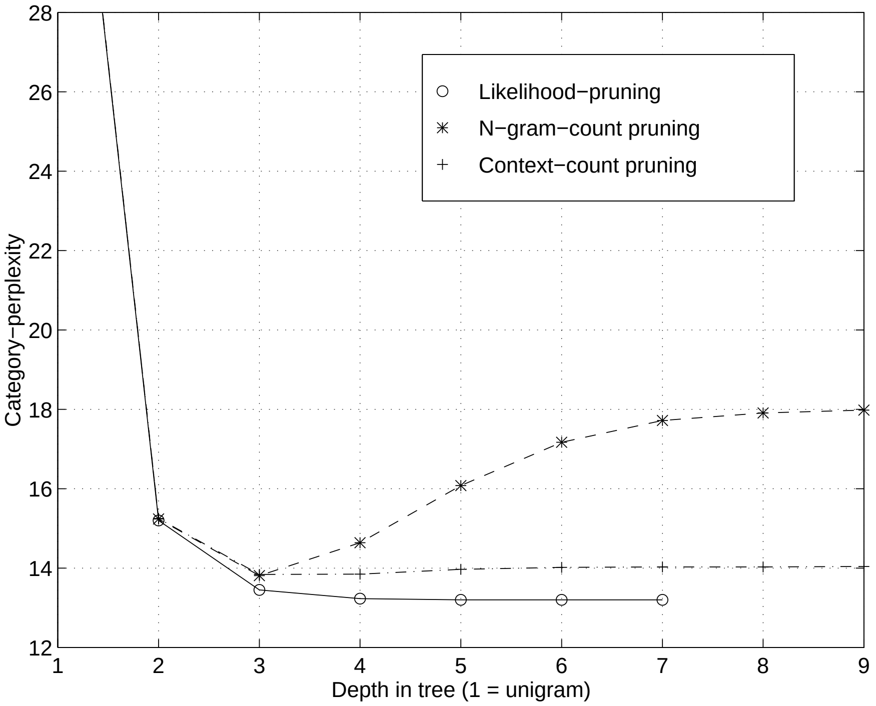

Figure 3.5: Evolution of the category-perplexity as a function of maximum *n*-gram length.

Consider now for each of the three corpora the tree obtained by likelihood pruning with $\lambda_{ct} = 5e - 6$. Figure 3.6 analyses the proportion of *n*-grams in each model for each n, and from it we conclude that:

- For small n (unigrams and bigrams), the number of possible *n*-grams is small and so the number of *n*-grams in the model is small.
- As n becomes large, the data become sparse, and consequently the cross-validated log probability permits fewer *n*-grams to be added to the model.
- The WSJ corpus, which is much larger than the other two and therefore less sparse, has a larger proportion of *n*-grams with larger n.

[23] Although the results are not shown here, similar behaviour was observed for the word perplexity.

[24] It may be possible to achieve better results by varying the threshold counts as a function of the *n*-gram length, but this was not investigated.

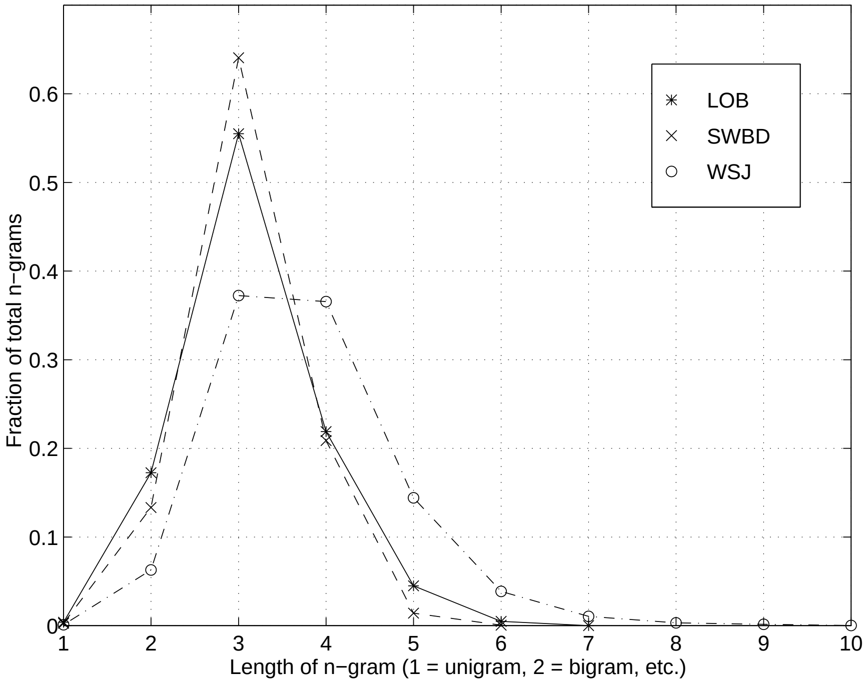

Figure 3.6: Proportion of *n*-grams in the tree as a function of n.

Furthermore, when considering the evolution of the category-perplexities for each model with the addition of each level to the tree, we find the behaviour illustrated in figure 3.7. Here each curve has been normalised with respect to its bigram perplexity in order to facilitate the comparison. The greater size of the WSJ corpus allows longer *n*-grams to be of more benefit, and this is reflected by the larger relative decreases in perplexity with the addition of 4-grams and higher. In contrast, the Switchboard corpus is too sparse to merit the addition of 7-grams.

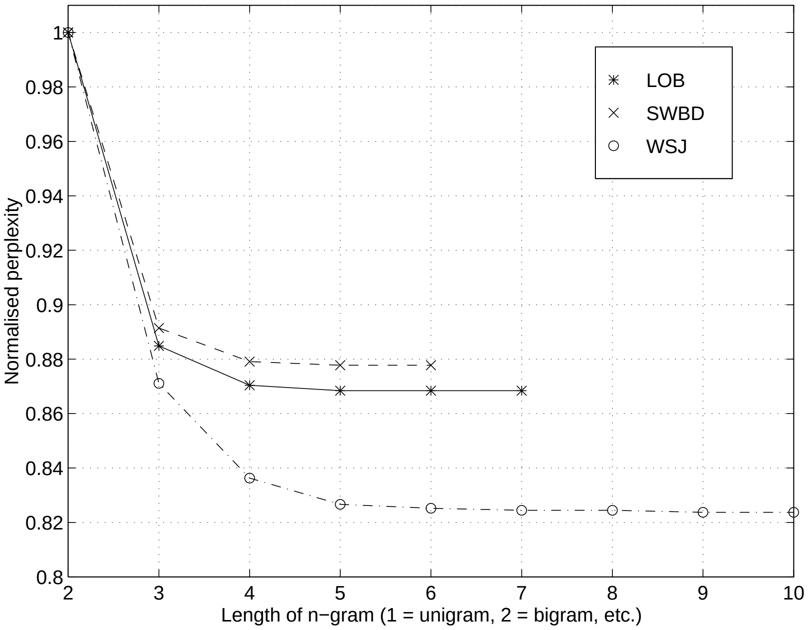

Figure 3.7: Perplexity as a function of maximum tree depth.

3.3.3. Word-perplexity

While the previous section has investigated the behaviour of the variable *n*-gram model using the category perplexity, here we consider the perplexities measured at the word level when implementing the language model described in section 3.2. For all three corpora, category trees produced using a likelihood pruning threshold of $\lambda_{ct} = 5e-6$ in equation (78) are employed. The effect which limiting the number of maintained history postulates to N_{hyp}^{max} has on performance is shown in the following table[25]. Beam-pruning was not employed in this experiment.

	Number of hypotheses (N_{hyp}^{max})				
	1	2	4	10	25
LOB	521.2	455.4	444.0	441.7	441.1
Switchboard	152.5	144.9	144.0	143.9	143.9
WSJ	522.7	483.2	477.1	476.2	475.8

Table 3.4: Perplexities when varying the maximum number of history postulates N_{hyp}^{max}.

The word perplexities decrease monotonically as the number of hypotheses is increased, demonstrating that the history equivalence class ambiguity has a significant effect on the language model performance. The largest decrease occurs as N_{hyp}^{max} is increased from 1 to 2, further increments leading to smaller reductions. The figures in the table indicate that a value of approximately 10 will yield near-optimal results, and therefore $N_{hyp}^{max} = 10$ will be used henceforth. Having fixed N_{hyp}^{max}, it remains only to decide on the value of the beam pruning parameter. Figure 3.8 shows its effect on perplexity for the LOB corpus, and we conclude that a value less than 10^{-1} should be chosen. Since a wider beam increases the computational effort, there is a tradeoff between the model accuracy and computational cost. Unless otherwise specified, all experiments have been carried out with at value of 10^{-2}.

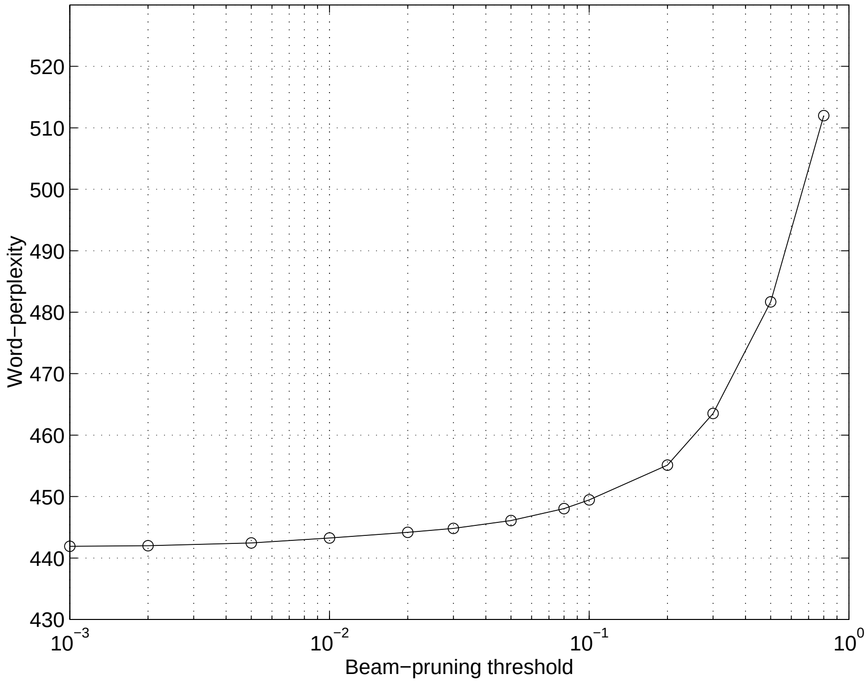

Figure 3.8: The effect of the beam pruning parameter on perplexity for the LOB corpus.

[25] For comparison, the perplexities of word-based *n*-gram language models for each corpus are given in appendix D

3.4. Comparing word- and category-based models

This chapter has introduced a language model based on variable-length part-of-speech *n*-grams. Word-based bigram and trigram models[26] often remain the most successful and popular choice, however. In view of this, the following sections present a detailed investigation into the relative performance of these two approaches, in an attempt to determine the strengths and weaknesses of each [59]. All tests employ language models trained on the WSJ corpus, the category-based model having been produced with a pruning threshold of $\lambda_{ct} = 5e - 6$ as defined in equation (78).

3.4.1. Overall probability estimates

Firstly we investigate the differences in the values of the probability estimates delivered by each model. The histograms[27] in figure 3.9 shows the number of words in the test set predicted with various log probabilities by each model. Note in particular the following:

- The category-based model assigns probabilities less than 10^{-6} to a smaller proportion of words.
- A larger number of words is predicted with high probability (between 1.0 and 0.1) by the word-based than by the category-based model.

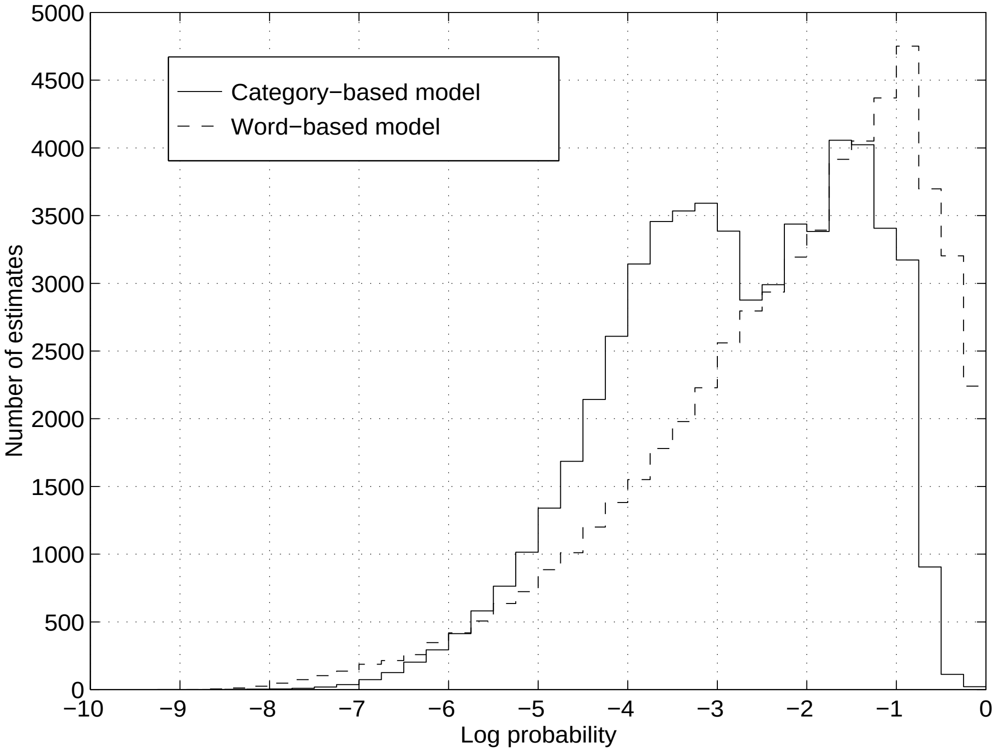

Figure 3.9: Overall distribution of log probabilities produced by word- and category-based models.

We conclude that the ability of the category-model to generalise to unseen word sequences leads to the smaller number of very low probabilities. However this same characteristic does not allow it to capture word-specific relations, and the strongest of these are responsible for the high-probability estimates delivered by the word-based model.

[26] As presented in section 2.4.

[27] A bin size of 0.25 has been used.

3.4.2. The effect of backoffs

The objective here is to investigate the probability estimates made by the category-based model and the word-based trigram when the latter does not back-off, or backs-off to various degrees. Table 3.5 presents the perplexities calculated according to the type of word-model *n*-gram request and backoff. The category- and word-based models are denoted by "CBM" and "WBM" respectively.

Type of *n*-gram request	% of test set	CBM perplexity	WBM perplexity
Unigram	1.1 %	564.6	1,027.6
Bigram (found)	19.3 %	321.6	95.3
Bigram (backed-off to unigram)	2.3 %	15,110.3	73,541.4
Trigram (found)	59.7 %	232.8	30.8
Trigram (backed-off to bigram)	13.4 %	3,418.9	3,530.2
Trigram (backed-off to unigram)	4.2 %	31,202.2	288,237.0

Table 3.5: Perplexities for various types of word *n*-gram requests.

From these results we may conclude that, when the word-based model does not back-off (a situation true for approximately 80% of the words in the test set), it performs significantly better on average than the category-based model. When backoffs do occur this is no longer true, and the intrinsic ability to generalise to unseen word *n*-tuples allows the category-based model often to deliver better probability estimates. It is interesting to note that a significant proportion (approximately 22%) of the predictions made by the category-based model are as good or better than those of the word-model, even when backing-off does nor occur.

3.4.3. Per-category analysis

Here the objective is to analyse the contribution to the overall test set log probability made by individual categories, as well as the average log probability associated with each, for non backed-off word *n*-grams.

The category is taken to be that assigned to the word by the same tagger used to tag the training corpus. Figures 3.10 and 3.11 illustrate[28] the absolute fraction of the total log probability each category is accountable for, while figure 3.12 shows the per-category average log probability. The horizontal axis indicates the frequency with which each such category occurs in the test set. From these figures we conclude that:

- As found in the previous section, when backing-off does not occur, the word-based model performs better than the category-based model.
- There is no clear relationship between the frequency of occurrence of a category and its perplexity, except that the categories with lowest average log probability also have been seen a very small number of times (this is not visible in figure 3.12, but becomes evident on expansion of the horizontal axis by several orders of magnitude). These categories are sparsely trained as they have been seen only a small number of times in the training set.

[28] The following most significant grammatical categories have been labelled in these figures: common noun (NN), plural common noun (NNS), adjective (JJ), proper noun (NP), verb base form (VB), past tense of verb (VBD), past participle (VBN), present participle (VBG), third-person singular verb (VBZ), adverb (RB), preposition (IN), cardinal (CD), singular or plural article (ATI), singular article (AT), coordinating conjunction (CC), subordinating conjunction (CS), letter of the alphabet (ZZ), end-of-sentence marker (SE), infinitival "to" (TO), unit of measurement (NNU).

- Although some categories have low average log probabilities, they are infrequent and therefore make a minor contribution overall.
- Common nouns (NN) are the most significant contributers to overall log probability, followed by plural common nouns (NNS), adjectives (JJ) and proper nouns (NP).
- In general it seems that words with semantic content, such as nouns and adjectives, are harder to predict (have lower probability) than syntactic function words (prepositions, articles, conjunctions and personal pronouns). This is true for both word- and category-based models. The two vertically separated groupings in figure 3.12 illustrate this point.
- There appears to be an approximately linear relationship between the frequency with which a category occurs and its contribution to the overall log probability. The constant of proportionality is different for words with significant semantic content and syntactic function words respectively. This is emphasised by the two elongated groupings in figures 3.10 and 3.11.

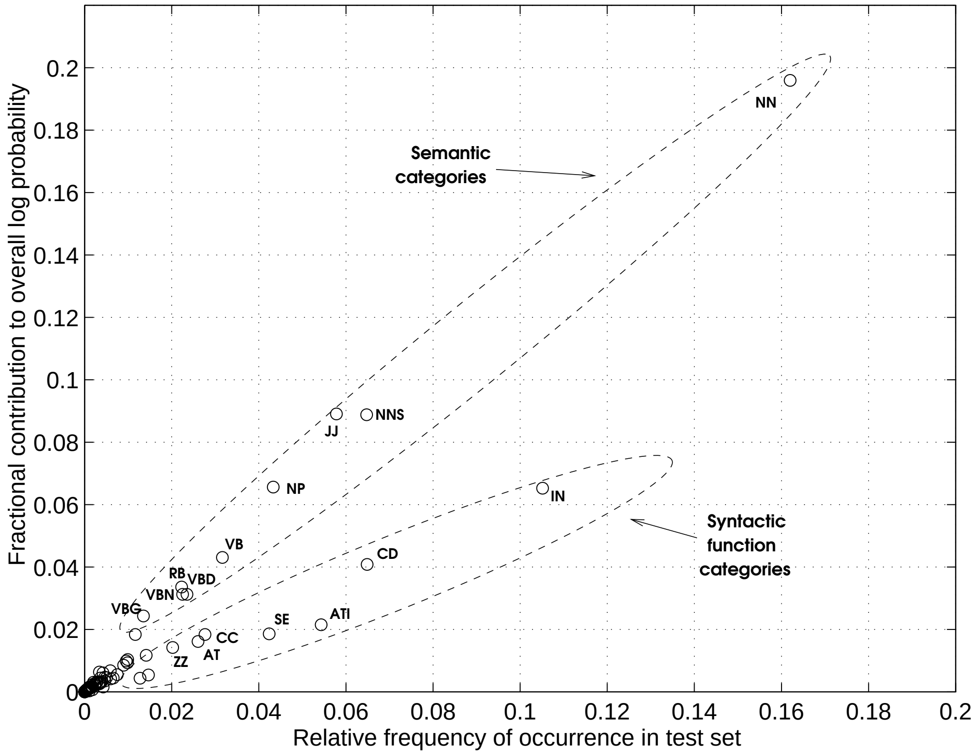

Figure 3.10: The contribution of each category to the overall test set log probability (word-based model).

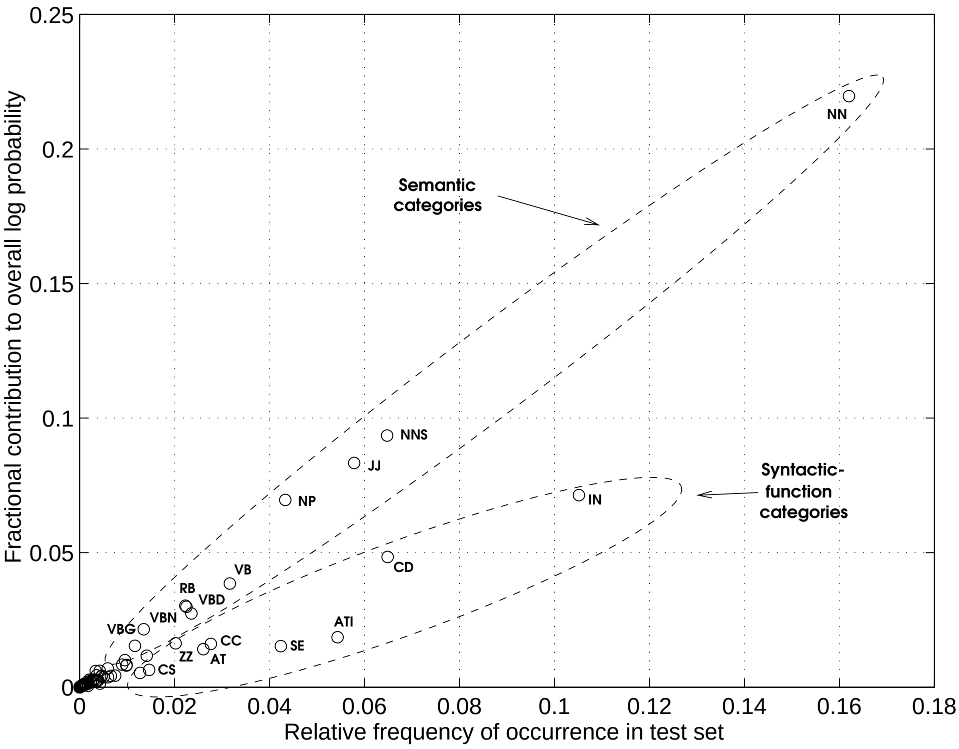

Figure 3.11: The contribution of each category to the overall test set log probability (category-based model).

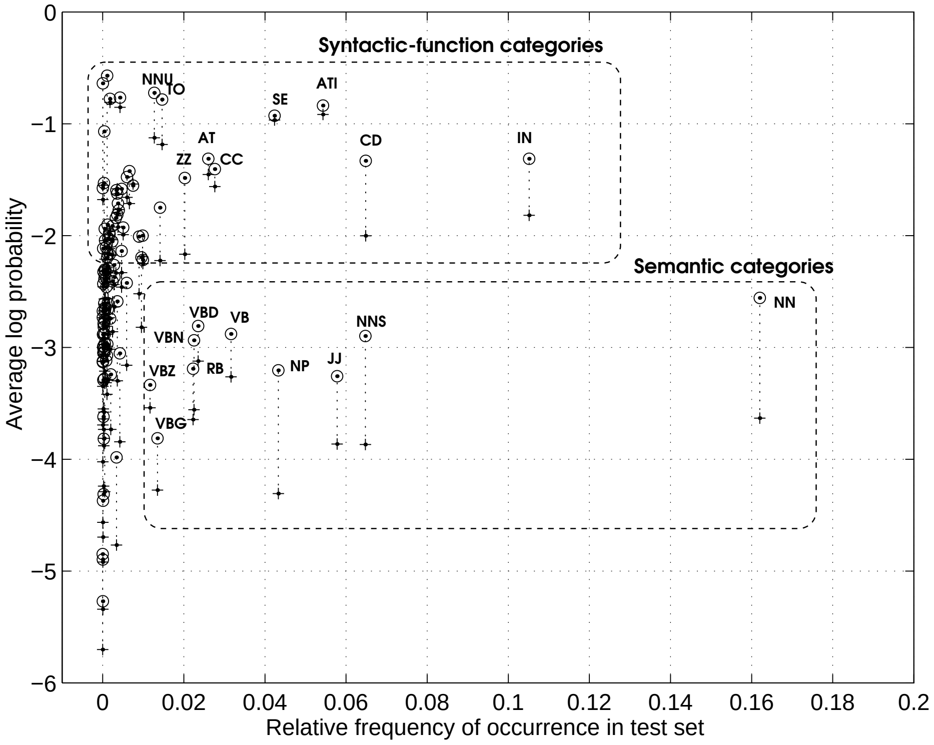

Figure 3.12: Average log probability per category for both word-based (⊙) and category-based (+) models.

3.4.4. Per-*n*-gram analysis

The findings of the preceding two sections have made it clear that word-based *n*-grams carry a significant amount of information that cannot be captured by their category-based counterparts. The objective of this section is to investigate what proportion of word *n*-grams play a significant role in improving upon the category-based model. In order to do this, the contribution to the difference in total log probability between the word- and category-based model made by each distinct word-trigram in a non-backoff situation is calculated. These contributions are subsequently sorted in order of decreasing value, normalised, and graphed in figure 3.13. Note the portion of the curve above 1.0, which is due to those trigrams assigned higher probabilities by the category-based model.

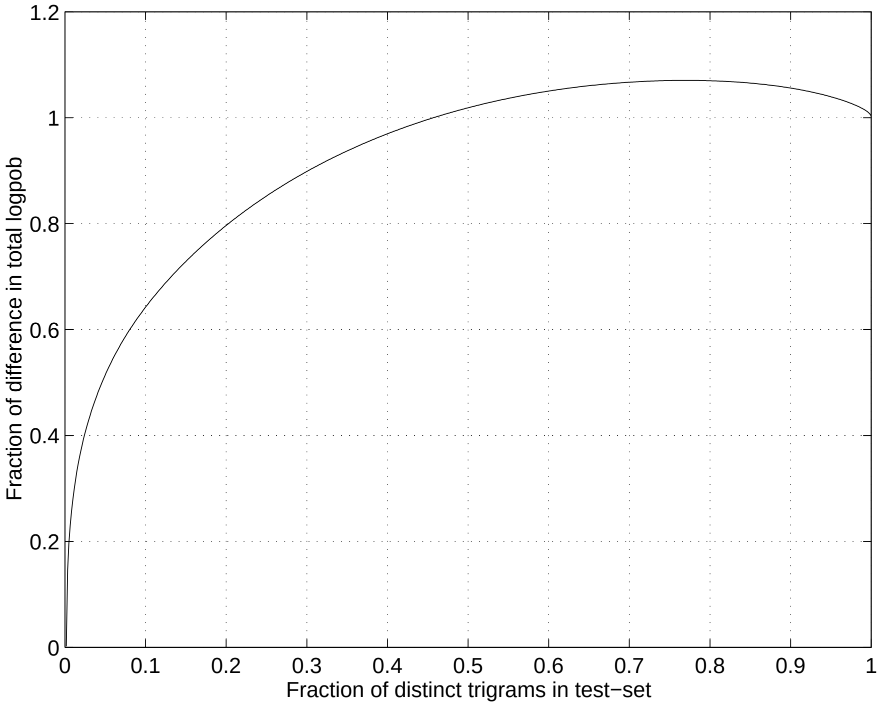

Figure 3.13: Contribution to the difference in log probabilities generated by word- and category-based models.

From figure 3.13 we may deduce that approximately half of the word trigrams contribute to the lead which the word-based model has over its category-based counterpart, the other half being predicted equally well or better by the latter. Furthermore, more than 50% of the improvement is contributed by only 5% of the trigrams. Thus, were category- and word-based models to be used in conjunction with one another, it should be possible to make the latter significantly more compact.

As a matter of interest, table 3.6 lists a few examples of trigrams for which the word- and category-models fare better respectively.

Word-model better	Category-model better
a border patrol	abortive rising of
a Roman Catholic	also compared him
accepted accounting principles	announcers or analysts
across national borders	as the point
Adam and Eve	be announced to
below zero Fahrenheit	both declined that
caught by surprise	closely held to
declaration of independence	declined to quote
five year old	farmers now raise
former investment banker	five hundred percent
Great Barrier Reef	grins and says
have grown accustomed	he agreed the
increase cash flow	Italian or French
lowest discount fares	last year shares
McDonnell Douglas Astronautics	minister or president
more than doubled	more than us
Nobel peace prize	new orders of
open heart surgery	open not closed
possible business combination	previously reported of
president Francois Mitterrand	president and Mr
racist and sexist	rest of an
registered as unemployed	rose to say
satellite into orbit	seven million futures
sentiment remained bearish	so healthy that
Soviet air defences	spokesman for an
television sports commentator	twenty thousand percent
Vancouver British Columbia	vision than chief

Table 3.6: Trigrams assigned higher probabilities by word- and category-based models respectively.

3.4.5. Robustness to domain-change

Experiments have to this point employed a test set whose character matches that of the training set closely, being derived from the same newspaper (the Wall Street Journal) and from the same period. An important issue in language modelling is how well the model will fare on data from a different test set domain (e.g. from a different newspaper, or with an entirely different character and style of text). In order to investigate the performance of the two language models on domains other than WSJ, the following four additional test sets were compiled from the LOB corpus [40].

- **Press reportage** (categories A, B and C from the LOB corpus, which are made up of various newspaper articles, editorials and reviews not from WSJ).

- **Religion** (category D of the LOB corpus, which contains text concerning religious topics).

- **Scientific writing** (category J of the LOB corpus).
- **Fiction** (categories K, L, N, P and R from the LOB corpus, consisting of adventure, mystery, Western, romantic and humorous fiction).

Table 3.7 summarises the performance of both the word- and category-based language models (WBM and CBM respectively) on these test sets. Performance on the WSJ test set is also shown for comparison. The category model used in this experiment was constructed with a pruning threshold of $\lambda_{ct} = 2e - 7$.

Test set	No. of words	% OOV	Perplexity (WBM)	Perplexity (CBM)
WSJ baseline	56,820	0.67	144.4	450.5
Press reportage	174,465	4.76	488.4	625.4
Religion	33,215	5.52	549.4	598.0
Scientific writing	154,755	6.63	624.3	671.7
Fiction	239,927	6.13	643.0	715.0

Table 3.7: Performance of the category- and word-based models on different test sets.

Although the word-based model outperforms the category-based model in all cases, it is interesting to see that, whereas the perplexity of the former increases by a factor of between 3.4 and 4.5, the perplexity of the latter does so only by a factor of between 1.3 and 1.6, indicating a reduced sensitivity to a change of test domain.

3.5. Summary and conclusion

A category-based language model employing *n*-grams of varying lengths has been described. A procedure allowing *n*-gram length to be optimised with respect to estimated performance is presented, and experiments using the LOB corpus show it to outperform conventional *n*-gram approaches while reducing the number of model parameters. Words may belong to multiple categories, and consequently the model bases its probability estimates on a set of possible classifications of the word history into category sequences. Each such classification has an associated probability, and is updated recursively for each successive word in a sentence.

In order to capture syntactic patterns with this language model, the categories have been chosen to correspond to part-of-speech classifications. To make these classifications available for large but untagged text corpora, the application of the language model as a statistical tagger has been described, and experimental evaluation shows improved in tagging accuracy when compared with a benchmark.

The category-based language models are significantly more compact than word-based models, and offer competitive performance in terms of perplexity, especially for sparse corpora. However, since category models are never able to capture relationships between particular words, but only between the categories to which these words belong, they are not able to fully exploit all the information available in a large training corpus. For this reason word-based models begin to perform significantly better as the amount of training material increases. However, closer analysis reveals that most of this lead is due to a relatively small proportion of *n*-grams in the word-model. In particular, category-based probability estimates remain competitive or superior in many instances, notably those for which the word-models needs to back-off. For this reason category-based models display a greater robustness to changes in the characteristics of the testing domain.

Chapter 4

Word-to-category backoff models

4.1. Introduction

Section 3.4 has shown that the category-based language model presented in section 3.2 delivers better probability estimates in backoff situations due to its intrinsic ability to generalise to unseen word sequences. It has also been shown that only a small fraction of word *n*-grams contribute to the largest part of this difference in performance. In this chapter we consider a technique that allows backoffs to take place from a word- to a category-based *n*-gram probability estimate, with the aim of retaining the advantages offered by each approach [58],[60].

4.2. Exact model

Consider the following language model, which backs off from a word- to a category-based probability estimate :

$$P_{wc}(w \mid \Phi_c) = \begin{cases} P_w(w \mid \Phi_w) & \text{if } \; w \in W_T(\Phi_w) \\ \beta(\Phi_c) \cdot P_c(w \mid \Phi_c) & \text{otherwise.} \end{cases} \tag{89}$$

where :

- w is the word for which we would like to estimate the probability of occurrence.
- Φ_w is the word-history upon which the probability estimate of the word-based *n*-gram language model depends, referred to the **word-level context** hereafter. For a bigram it is the preceding word, and for a trigram the preceding two words.
- Φ_c is the word history and associated set of category history equivalence-class postulates upon which the probability estimate of the category-based model depends, termed the **category-level context** hereafter. Due to the recursive nature in which the category history equivalence class postulates are maintained[29], the category-level context is in general only completely defined by the entire word history (i.e. to the beginning of the current sentence). Thus the number of category-level contexts is potentially huge, and the mapping from Φ_w to Φ_c is one-to-many.
- $P_w(w \mid \Phi_w)$ is the probability estimate for w obtained from the word-based language model.
- $P_c(w \mid \Phi_c)$ is the corresponding probability obtained from the category-based language model.

[29] Refer to section 3.2.3.

- $W_T(\Phi_w)$ is the set of words in the word-context Φ_w for which the word-model estimates will be used, backoffs occurring in other cases.
- $\beta\left(\cdot\right)$ is the backoff weight, $\beta\left(\cdot\right) > 0$.

The estimate (89) is designed to employ word *n*-grams to capture significant sequential dependencies between particular words, while using the category-based model for less frequent word combinations. From the requirement:

$$\sum_{\forall w} P_{wc}(w \mid \Phi_c) = 1.0 \quad \forall \;\; \Phi_c \tag{90}$$

it follows from (89) that:

$$\beta(\Phi_c) = \frac{1.0 - \sum\limits_{\forall w \in W_T} P_w(w \mid \Phi_w)}{1.0 - \sum\limits_{\forall w \in W_T} P_c(w \mid \Phi_c)} \tag{91}$$

4.3. Approximate model

Due to the large number of different possible Φ_c used by the category model, precalculation of $\beta\left(\cdot\right)$ according to equation (91) is not feasible. It would be more convenient to obtain backoff constants for every word-level context instead, but the dependence of the denominator of (91) upon Φ_c does not permit this, since it is not uniquely fixed by Φ_w. Run-time calculation of $\beta\left(\cdot\right)$ using equation (91) increases the computational complexity of a probability calculation within any particular context by a factor of approximately W_T in comparison with a model for which these parameters are precalculated, and thus represents a significant computational burden. To circumvent this, we note that the category-level context is most strongly influenced by the most recent words, and hence make the approximation:

$$P_c(w \mid \Phi_c) \simeq P_c(w \mid \hat{\Phi}_c) \tag{92}$$

where $\hat{\Phi}_c$ is the the category-level context corresponding to Φ_w when assuming no prior knowledge of the words preceding Φ_w. Note that there is a unique $\hat{\Phi}_c$ for each Φ_w, so that we may define:

$$\hat{P}_c(w \mid \Phi_w) \overset{def}{=} P_c\left(w \mid \hat{\Phi}_c\right) \qquad \text{for} \quad \Phi_w \Rightarrow \hat{\Phi}_c \tag{93}$$

and now we approximate the backoff weights by:

$$\beta(\Phi_c) \simeq \beta(\hat{\Phi}_c) \overset{def}{=} \beta(\Phi_w) \qquad \text{for} \quad \Phi_w \Rightarrow \hat{\Phi}_c$$

and hence from (91) we find:

$$\beta(\Phi_w) = \frac{1.0 - \sum\limits_{\forall w \in W_T} P_w(w \mid \Phi_w)}{1.0 - \sum\limits_{\forall w \in W_T} \hat{P}_c(w \mid \Phi_w)} \tag{94}$$

This choice of $\beta(\cdot)$ in general no longer satisfies equation (90) however, and so we adjust the backoff model (89) as follows:

$$P_{wc}(w \mid \Phi_c) = \begin{cases} P'_w(w \mid \Phi_c) & \text{if} \quad w \in W_T(\Phi_w) \\ \beta(\Phi_w) \cdot P_c(w \mid \Phi_c) & \text{otherwise.} \end{cases} \tag{95}$$

where $P'_w(w \mid \Phi_c)$ is some approximation of $P_w(w \mid \Phi_w)$. Now for equation (90) to be satisfied, it follows from (95) that:

$$\sum_{\forall w \in W_T} P'_w(w \mid \Phi_c) \; = \; \Big(1 - \beta(\Phi_w)\Big) + \beta(\Phi_w) \cdot \sum_{\forall w \in W_T} P_c(w \mid \Phi_c) \tag{96}$$

so we may choose to define:

$$P'_w(w \mid \Phi_c) \;\overset{def}{=}\; k(w \mid \Phi_w) \cdot \Big(1 - \beta(\Phi_w)\Big) + \beta(\Phi_w) \cdot P_c(w \mid \Phi_c) \tag{97}$$

where, in order to satisfy (96) we require that:

$$\sum_{\forall w \in W_T} k(w \mid \Phi_w) = 1.0 \tag{98}$$

The quantity $(1 - \beta(\Phi_w))$ may be interpreted as a probability mass which must be distributed among the elements of W_T by a suitable choice of $k(w \mid \Phi_w)$. The adopted approach is to distribute this mass using the ratio:

$$\frac{P_w(w \mid \Phi_w)}{\sum\limits_{\forall w \in W_T} P_w(w \mid \Phi_w)} \tag{99}$$

and therefore distribute probability mass to *n*-grams approximately in the same proportion as the word-based language model. Proceeding from equations (97) and (99), and employing the approximation (93) we are led to require :

$$\frac{k(w \mid \Phi_w) \cdot \Big(1 - \beta(\Phi_w)\Big) + \beta(\Phi_w) \cdot \hat{P}_c(w \mid \Phi_w)}{1 - \beta(\Phi_w) + \beta(\Phi_w) \cdot \sum\limits_{\forall w \in W_T} \hat{P}_c(w \mid \Phi_w)} = \frac{P_w(w \mid \Phi_w)}{\sum\limits_{\forall w \in W_T} P_w(w \mid \Phi_w)}$$

from which it follows that:

$$\begin{aligned} \alpha(w \,|\, \Phi_w) \; = \; & \left(1 - \beta(\Phi_w) + \beta(\Phi_w) \cdot \sum_{\forall w \in W_T} \hat{P}_c(w \,|\, \Phi_w)\right) \cdot \frac{P_w(w \,|\, \Phi_w)}{\sum\limits_{\forall w \in W_T} P_w(w \,|\, \Phi_w)} \\ & -\beta(\Phi_w) \cdot \hat{P}_c(w \,|\, \Phi_w) \end{aligned} \tag{100}$$

where

$$\alpha(w \mid \Phi_w) = k(w \mid \Phi_w) \cdot \Big(1 - \beta(\Phi_w)\Big) \tag{101}$$

This choice of $k(w \mid \Phi_w)$ satisfies equation (98). Furthermore, as:

$$\hat{P}_c\left(w \mid \Phi_w\right) \rightarrow P_c\left(w \mid \Phi_c\right)$$

we find from (94), (97) and (101) that:

$$\begin{aligned} P'_w\left(w \,|\, \Phi_c\right) \;&\rightarrow \left(1 - \frac{1 - \sum\limits_{\forall w \in W_T} P_w\left(w \,|\, \Phi_w\right)}{1 - \sum\limits_{\forall w \in W_T} P_c\left(w \,|\, \Phi_c\right)} + \frac{1 - \sum\limits_{\forall w \in W_T} P_w\left(w \,|\, \Phi_w\right)}{1 - \sum\limits_{\forall w \in W_T} P_c\left(w \,|\, \Phi_c\right)} \cdot \sum_{\forall w \in W_T} P_c\left(w \,|\, \Phi_c\right)\right) \cdot \\ &\qquad \left(\frac{P_w\left(w \,|\, \Phi_w\right)}{\sum\limits_{\forall w \in W_T} P_w\left(w \,|\, \Phi_w\right)}\right) \\ &= \left(1 - \left(\frac{1 - \sum\limits_{\forall w \in W_T} P_w\left(w \,|\, \Phi_w\right)}{1 - \sum\limits_{\forall w \in W_T} P_c\left(w \,|\, \Phi_c\right)}\right) \cdot \left(1 - \sum_{\forall w \in W_T} P_c\left(w \,|\, \Phi_c\right)\right)\right) \cdot \frac{P_w\left(w \,|\, \Phi_w\right)}{\sum\limits_{\forall w \in W_T} P_w\left(w \,|\, \Phi_w\right)} \\ &= P_w\left(w \,|\, \Phi_w\right) \end{aligned}$$

which means that the approximate backoff (95) converges to the exact backoff (89) as the estimates $\hat{P}_c\left(w \mid \Phi_w\right)$ approach the exact values $P_c\left(w \mid \Phi_c\right)$.

Finally, although equation (100) guarantees:

$$P'_w\left(w \mid \Phi_c\right) > 0 \tag{102}$$

when the approximation (92) is perfect, it does not do so in general. In order to guarantee (102) it is sufficient to require:

$$\alpha(w \mid \Phi_w) \geq 0 \tag{103}$$

so that, using equation (100) it follows that:

$$\beta(\Phi_w) \leq \frac{P_w(w \mid \Phi_w)}{\hat{P}_c\left(w \mid \Phi_w\right) \cdot \left(\sum\limits_{\forall w \in W_T} P_w\left(w \mid \Phi_w\right)\right) + P_w\left(w \mid \Phi_w\right) \cdot \left(1 - \sum\limits_{\forall w \in W_T} \hat{P}_c\left(w \mid \Phi_w\right)\right)} \tag{104}$$

While calculating $\beta(\Phi_w)$, the equality in equation (104) should be enforced whenever the inequality is violated. Referring back to equation (97), this is equivalent to demanding the probability mass distributed to each word to be positive. In practice, this adjustment is required very infrequently.

The use of the estimates $\hat{P}_c\left(\cdot\right)$ allows the backoff constants $\alpha\left(w \mid \Phi_w\right)$ and $\beta\left(\Phi_w\right)$ to be precalculated, making the model (95) significantly more computationally efficient than the exact model (89), while continuing to employ in backoff situations the probabilities delivered by the category model.

Related research, carried out independently and concurrently, has recently been reported in [49]. It describes a method that allows backoffs to occur from a word- to a category-based bigram language model. In this case, however, words must belong to a unique category, which greatly simplifies the calculation of the backoff weights.

4.4. Model complexity : determining W_T

To this point it has been assumed that, for each word-level context Φ_w, a set of words W_T has been established for which probabilities will be calculated according to the word-based language model. A simple choice for W_T would be the set of all words seen within the context Φ_w in the training set. Denote this choice by $\mathbf{W}_T$, and note that when $W_T = \mathbf{W}_T$, backing-off occurs only for truly unseen events.

However, the approach taken here has been to reduce the size of W_T by eliminating words whose presence do not afford the word-based model much predictive power in relation to the category-based model. Since this process eliminates *n*-grams from the word-based model component, it allows the complexity[30] of the word-to-category backoff language model to be reduced. Note that since the category-based component is fixed, it determines the minimum overall complexity.

The number of words in W_T has been reduced in two ways, both leading to similar results.

4.4.1. Testing *n*-gram counts

Consider a word w which has been seen in in context Φ_w a total of $N(w \mid \Phi_w)$ times, and denote the number of times the context Φ_w itself was seen by $N(\Phi_w)$. Based on the category-model probability estimate $\hat{P}_c\left(w \mid \Phi_w\right)$, the number of times we would expect to see w in Φ_w is:

$$\hat{N}_c\left(w \mid \Phi_w\right) \;=\; \hat{P}_c\left(w \mid \Phi_w\right) \cdot N\left(\Phi_w\right)$$

Assuming words to occur independently within each context, thus exhibiting a binomial distribution, the variance of the count $N\left(w \mid \Phi_w\right)$ is [18]:

$$\sigma^2\left(w \mid \Phi_w\right) \;=\; \hat{P}_c\left(w \mid \Phi_w\right) \cdot \Big(1 - \hat{P}_c\left(w \mid \Phi_w\right)\Big) \cdot N\left(\Phi_w\right)$$

Using the normal approximation of the binomial distribution, we determine whether the actual count $N\left(w \mid \Phi_w\right)$ exceeds the expected count $\hat{N}_c\left(w \mid \Phi_w\right)$ by a certain fraction δ with a certain confidence ξ by testing whether:

$$N\left(w \mid \Phi_w\right) - (1+\delta) \cdot \hat{N}_c\left(w \mid \Phi_w\right) \quad > \quad \xi \cdot \sigma\left(w \mid \Phi_w\right) \tag{105}$$

where w is retained in W_T when the test succeeds. In practice the value of ξ is fixed (e.g. to 2.33 for 1% or 1.645 for 5% confidence levels), and the value of δ is varied to control the size of W_T.

4.4.2. Testing the effect on the overall probability

Testing the *n*-gram counts as described in the previous section allows *n*-grams to be retained in W_T when their counts are seen to differ from the expected ones in a statistically significant manner. The effect of such pruning decisions upon the perplexity of the resulting language model is not clear, however. More precisely, a frequent *n*-gram occurring only slightly more often in the word- than in the category-model might be discarded even though it has a significant effect on the overall probability due to its high

[30] The complexity of the word-to-category backoff model is taken to be the sum of the total number of *n*-grams in the word- and category-based components.

frequency. To avoid this, a second pruning criterion has been implemented, which discards those *n*-grams with the smallest effect on the training set log probability. The perplexity versus complexity tradeoff of the resulting models is similar to that achieved with the count-pruning method.

In particular, an *n*-gram is retained when:

$$\Delta\overline{LP} > \delta \tag{106}$$

where $\Delta\overline{LP}$ is the change in mean per-word log probability when using the word-model instead of the category-model, and is calculated using:

$$\Delta\overline{LP} = \frac{N(w \mid \Phi_w) \cdot \left(\log\left[P_w\left(w \mid \Phi_w\right)\right] - \log\left[\hat{P}_c\left(w \mid \Phi_w\right)\right]\right)}{N_c}$$

where N_c is the total number of words in the training corpus [31].

4.5. Model building procedure

In summary of sections 4.3 and 4.4, the following steps need to be taken in order to construct a word-to-category backoff language model:

- Build a category-based language model as described in chapter 3.
- Build a word-based *n*-gram model.
- By application of equation (93), determine the probabilities $\hat{P}_c\left(w \mid \Phi_w\right)$ for each *n*-gram in the word model.
- For each context in the word-model, determine the set W_T using either equation (105) or equation (106). This is essentially a process of pruning *n*-grams from the word model.
- For the remaining *n*-grams in the word-based model, calculate the α and β values according to equations (100) and (94).
- Apply the language model according to equation (95).

4.6. Results

To gauge its performance, the backoff technique has been applied to the LOB, Switchboard and WSJ text corpora[32]. In each case language models of various complexities were generated by varying the size of W_T as described in section 4.4, and the resulting perplexities compared with those achieved using a word trigram trained on the same data. The size of the latter was controlled by the standard technique of discarding *n*-grams occurring fewer than a threshold number of times in the training text (i.e. varying the *n*-gram cutoffs). Identical thresholds were employed for both bigrams and trigrams in all cases.

[31] Normalisation by the quantity N_c makes $\Delta\overline{LP}$ and thus also δ fairly corpus-independent.
[32] Descriptions of these corpora and their division into test- and training sets may be found in appendix D.

4.6.1. LOB corpus

Category language models of differing complexities were built using pruning thresholds of $\lambda_{ct} = 1e-4$ and $\lambda_{ct} = 5e-6$ as described in chapter 3. Table 4.1 shows details for these, and figure 4.1 the performance of the resulting two word-to-category backoff (WTCBO) models[33].

	Category model 1 (1e-4)	**Category model 2** (5e-6)	**Word trigram**
Parameters	13,585	44,380	1,142,457
Perplexity	469.4	443.3	413.1

Table 4.1: Language models for the LOB corpus.

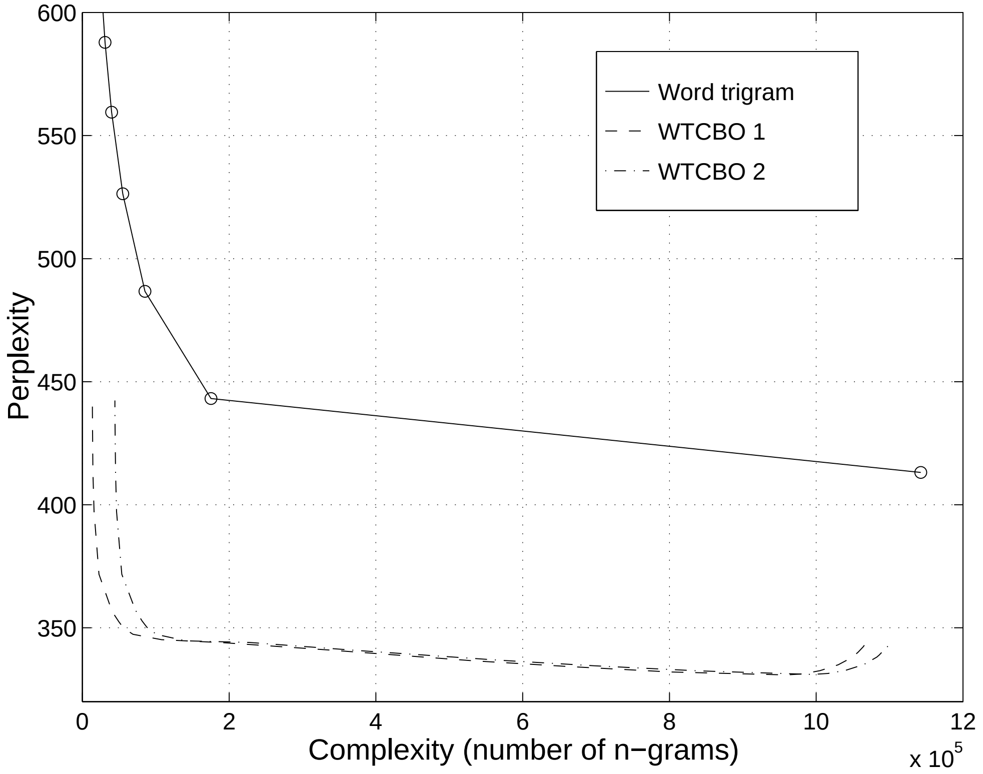

Figure 4.1: Performance of word-to-category backoff and trigram language models for the LOB corpus.

For both WTCBO models, significant perplexity reductions are achieved. For example, a model with approximately 1 million *n*-grams results in a perplexity of 330.8, which is a 20% improvement over the word-based trigram. Furthermore, a more favourable size versus performance tradeoff is achieved, particularly for the smaller ($\lambda_{ct} = 1e-4$) category model.

[33] WTCBO 1 and 2 are built using category model 1 and 2 respectively.

4.6.2. Switchboard corpus

Category-based language models were constructed again using pruning thresholds of $\lambda_{ct} = 1e-4$ and $\lambda_{ct} = 5e-6$, and table 4.2 shows the characteristics of these individual models while figure 4.2 shows the performance of the resulting two word-to-category backoff (WTCBO) models.

	Category model 1 (1e-4)	**Category model 2** (5e-6)	**Word trigram**
Parameters	13,627	54,547	1,183,880
Perplexity	154.4	144.1	96.57

Table 4.2: Language models for the Switchboard corpus.

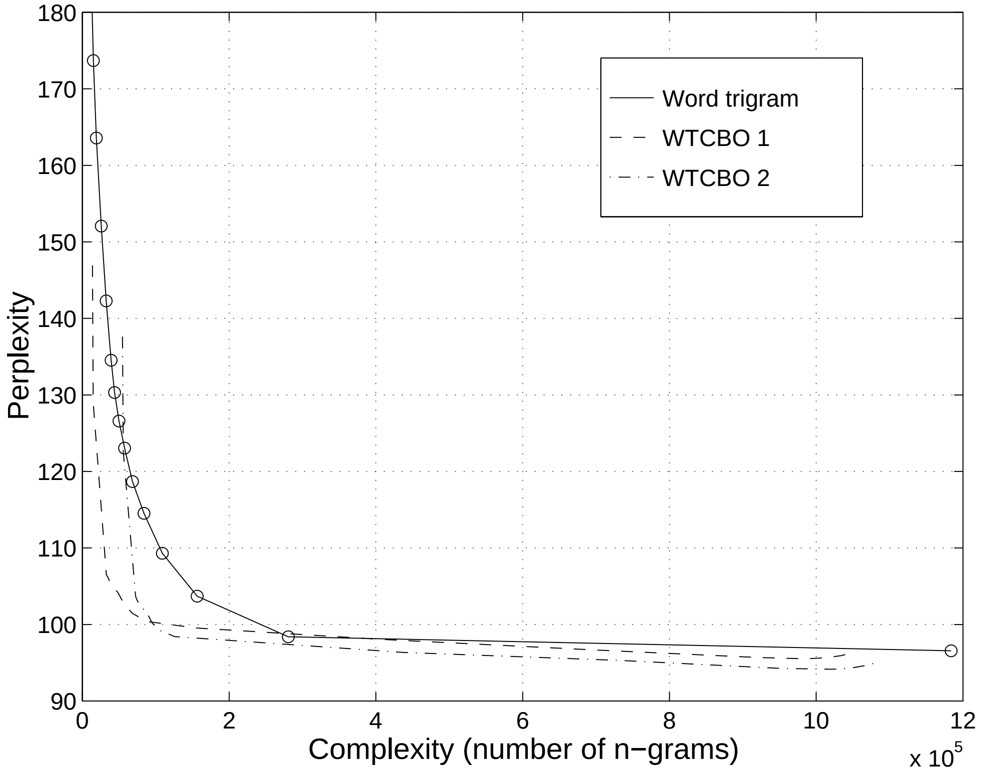

Figure 4.2: Performance of the word-to-category backoff and trigram models for the Switchboard corpus.

The larger category model leads to a word-to-category backoff model with lower minimum perplexity and improved performance for model complexities exceeding approximately 60,000 *n*-grams, while the smaller leads to performance that is slightly diminished in this region but better for smaller numbers of *n*-grams. The word-to-category backoff model offers a slight improvement in perplexity with respect to the trigram (approximately 2.7% in figure 4.2) and, depending on the choice of category model complexity, a significantly improved complexity versus performance tradeoff characteristic.

The limited number of conversational topics in the Switchboard corpus leads to a reduction in the training set sparseness and better coverage by the word trigram than for LOB (the trigram backoff rate drops by 41% from the latter to the former). Thus there is less need for the generalising ability of the category-model, and consequently a smaller perplexity improvement.

4.6.3. WSJ corpus

For this corpus, results were determined only for one category-based model, which was constructed using a pruning threshold of $\lambda_{ct} = 5e - 6$. Table 4.3 shows individual language model details, and figure 4.3 the performance of the word-to-category backoff (WTCBO) and trigram models.

	Category model	**Word trigram**
Parameters	174,291	13,599,678
Perplexity	476.5	144.4

Table 4.3: Language models for the WSJ corpus.

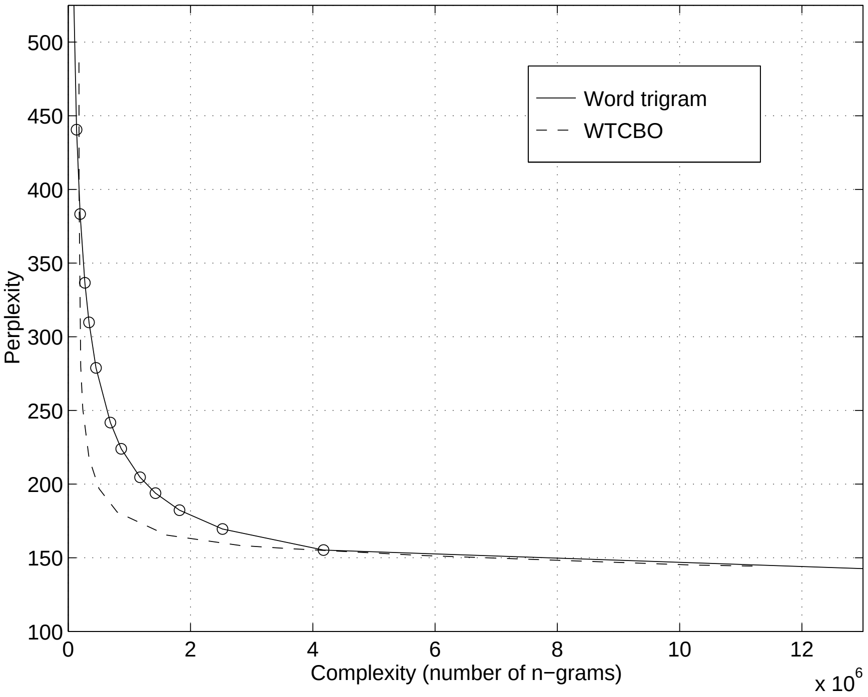

Figure 4.3: Performance of word-to-category backoff and trigram language models for the WSJ corpus.

From figure 4.3 we see that, in this case, the word-to-category backoff model does not offer substantial perplexity improvements over the word-based trigram when the number of parameters is large. However, it nevertheless continues to offer a better complexity versus performance tradeoff. As was found for the Switchboard corpus, perplexity improvements are small when the word-model is well-trained, which it is for WSJ due to the large quantity of training data.

4.7. Summary and conclusion

This chapter has presented a language model which backs-off from a word-based to a category-based *n*-gram estimate. The category-model is able to generalise to unseen word sequences and therefore appropriate in backoff situations. When compared with standard trigram models, this technique greatly improves perplexities for sparse corpora, and offers significantly enhanced complexity versus performance tradeoffs.

Chapter 5

Word-pair dependencies in category-based language models

5.1. Introduction

An underlying assumption of the category-based language model presented in chapter 3 is that the probability of a word depends only upon the category to which it belongs, and that its occurrence is therefore equally likely at any point in a corpus where this category occurs. However, factors such as the topic and the style of the text may cause certain words to occur in groups, thereby violating this assumption. While short-term fixed-length word dependencies may be incorporated by means of word *n*-grams as detailed in chapter 4, longer-term relations must be treated separately. This chapter presents a technique by means of which such long-range word associations may be captured by the category-based language model. Explicit account is taken of the transient strength of the relationship as a function of a particular definition of the separating distance [61], [62].

Word-pairs have been combined with word *n*-gram language models within a maximum-entropy framework [79] [80], by linear interpolation [53], and as long-distance bigrams [93]. The development here differs in that it takes explicit account of the distance between word occurrences, and takes specific advantage of the category-based language model.

5.2. Terminology

Consider the effect which the occurrence of a **trigger** word-category pair (w_{trig}, v_{trig}) has on the subsequent probability of occurrence of a **target** pair (w_{targ}, v_{targ}). Refer to the sequence consisting of all trigger occurrences as well as all words belonging to the target category as the **trigger-target stream**, and denote it by $S(w_{trig}, v_{targ})$. Let the total number of words in the stream $S(w_{trig}, v_{targ})$ be N_s, and the number of occurrences of the trigger and target words respectively be $N_s(w_{trig})$ and $N_s(w_{targ})$. It will henceforth be assumed that the stream has been taken from the training corpus, and under this condition we note that $N_s(w_{trig}) = N_c(w_{trig}, v_{trig})$ and $N_s(w_{targ}) = N_c(w_{targ}, v_{targ})$ where $N_c(w, v)$ is the number of times in the corpus w occurs as a member of category v. Furthermore:

$$N_s \;=\; \begin{cases} N_c(v_{targ}) & \text{if} \quad v_{trig} = v_{targ} \\ N_c(v_{targ}) + N_c(w_{trig}, v_{trig}) & \text{otherwise} \end{cases}$$

where $N_c(v)$ is the number of times category v occurs in the training corpus.

Assuming the trigger and the target to be statistically independent, their occurrence probabilities within $S(w_{trig}, v_{targ})$ are respectively given by:

$$p_s(w_{trig}) \;=\; \frac{N_s(w_{trig})}{N_s} \tag{107}$$

and

$$p_s(w_{targ}) \;=\; \frac{N_s(w_{targ})}{N_s} \tag{108}$$

but since:

$$p\left(w_{targ}|v_{targ}\right) \;=\; \frac{N_c(w_{targ}, v_{targ})}{N_c(v_{targ})}$$

we see that the stream and category-conditional word probabilities for the target are related by:

$$p\left(w_{targ}|v_{targ}\right) \;=\; K_s \cdot p_s(w_{targ}) \tag{109}$$

with:

$$K_s \;=\; \frac{N_s}{N_c(v_{targ})} \tag{110}$$

Define the **separating distance** d between a trigger-target pair as the number of times a word belonging to category v_{targ} is seen after witnessing the trigger and before the first sighting of the target itself. Hence $d \in \{0, 1, 2, 3, \ldots, \infty\}$ is the separating distance in the trigger-target stream. This definition of distance has been employed as a way of minimising syntactic effects on word co-occurrences, notably the phenomenon that there are certain categories which rarely follow one another for grammatical reasons. Syntactic effects should be reduced as much as possible since they are already modelled by the category n-gram component of the language model.

In the following a distinction will be drawn between the case where trigger and target are the same word (termed **self-triggers**) and the case where they differ (referred to as **trigger-target pairs**).

5.3. Probabilistic framework

Let the assumption that the probability of a word depends only upon its designated category, as stipulated by equation (52), be referred to as the **independence assumption**. Empirical investigation of the category-conditional probability $p\left(w_j|v_k\right)$ as a function of the distance d reveals an exponential decay towards a constant for words between which a correlation exists. Figure 5.1 illustrates this for the case where the trigger is the titular noun "*president*" and the target is the proper noun "*congress*". [34]

[34] The data are drawn from the WSJ corpus (refer to appendix D).

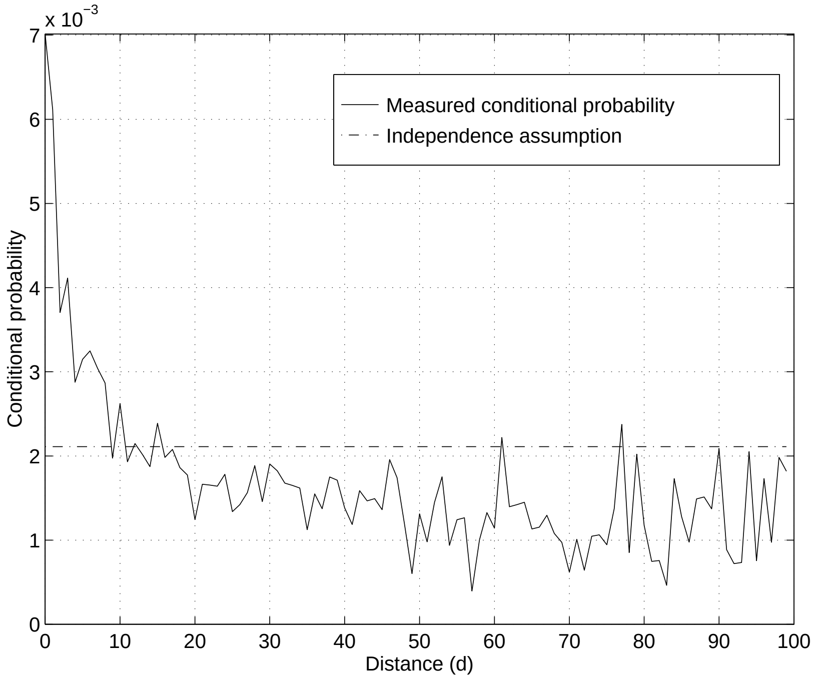

Figure 5.1: Measured conditional probability $P(w_j|v_k, d)$ of "*congress*" having seen "*president*".

The transient behaviour displayed in this graph is typical, and has motivated the following postulated form of the category-conditional probability:

$$p(w_{targ}|v_{targ}, d) \;=\; P_v + \gamma_v \cdot e^{-\rho \cdot d} \tag{111}$$

which is an exponential decay towards a constant probability P_v in which γ_v and ρ define the strength and rate of decay respectively. The stream-probability is found by scaling according to equation (109) :

$$p_s(w_{targ}, d) \;=\; P_b + \gamma \cdot e^{-\rho \cdot d} \tag{112}$$

with $P_v = K_s \cdot P_b$ and $\gamma_v = K_s \cdot \gamma$. Assuming the triggers to occur independently in the stream, their probability P_a is given by equation (107) :

$$P_a \;=\; p_s(w_{trig}) \tag{113}$$

It follows that the probability mass function $p_s(d)$ for the target occurrence after sighting the trigger is:

$$p_s(d) \;=\; \kappa \cdot \left(\prod_{i=0}^{d-1} \left(1 - P_a - P_b - \gamma \cdot e^{-\rho \cdot i} \right) \right) \cdot \left(P_b + \gamma \cdot e^{-\rho \cdot d} \right) \tag{114}$$

The normalising constant κ accounts for the probability mass associated with cases in which a trigger follows another trigger before sighting the target.

The empirical estimates of figure 5.1 have been obtained by binning counts over the graphed distance range. However, from a storage point of view, the potentially extremely large number of word-pair relations make this approach infeasible for large-scale application, and hence it is not possible to obtain the parameters of equation (112) from a direct fit to the data. The estimation of P_b, and then of γ and ρ, is treated in the following two sections.

5.3.1. Estimating P_b

The probability P_b may be estimated from the tail of the distribution, where the transient effect of the exponential term in (112) is assumed to be insignificant. Were the trigger and target to occur independently, their separating distance would have a geometric distribution, and we use its mean μ_g as a rough indication of the point at which the exponential term becomes negligible:

$$\mu_g \;=\; \frac{N_s - N_s(w_{trig}) - N_s(w_{targ})}{N_s(w_{trig}) + N_s(w_{targ})} \tag{115}$$

We could estimate P_b using counts of all trigger-pair occurrences with distances beyond this mean, i.e.:

$$P_b \;=\; \frac{N_s(w_{targ})|_{d>\mu_g}}{N_s|_{d>\mu_g}} \tag{116}$$

where:

$$N_s|_{d>\mu_g} \;=\; \begin{cases} N_c(v_{targ})|_{d>\mu_g} & \text{if} \quad v_{trig} = v_{targ} \\ N_c(v_{targ})|_{d>\mu_g} + N_c(w_{trig}, v_{trig})|_{d>\mu_g} & \text{otherwise} \end{cases}$$

and where $N_s(w_{targ})|_{d>\mu_g}$, $N_s(v_{targ})|_{d>\mu_g}$ and $N_s(w_{trig})|_{d>\mu_g}$ are the respective number of times the target word w_{targ}, the target category v_{targ}, and the trigger word w_{trig} have been seen at distances exceeding μ_g in the trigger-target stream[35]. However, when $N_s(w_{targ})|_{d>\mu_g}$ is small, this estimate may be unreliable, and therefore we introduce the following smoothed estimate:

$$P_b \;=\; \varepsilon \cdot \frac{N_s(w_{targ})|_{d>\mu_g}}{N_s|_{d>\mu_g}} + (1-\varepsilon) \cdot p_s(w_{targ}) \tag{117}$$

with the parameter ε chosen to be:

$$\varepsilon = \frac{N_s(w_{targ})|_{d>\mu_g}}{N_s(w_{targ})|_{d>\mu_g} + \eta}$$

Equation (117) is an interpolation of the relative frequency (116), and the relative frequency (108) obtained when pooling all the data (irrespective of d). The interpolation weight ε depends on the counts, and tends to unity[36] as the number of sightings beyond μ_g increases. The value of η is related to the number of sightings we consider to give us confidence in the estimate (116), since it determines how quickly ε approaches 1. Hence it should be chosen to be a small constant greater than zero. All experiments were carried out with $\eta = 5.0$, and its particular value was observed not to influence performance strongly.

5.3.2. Estimating γ and ρ

Expressions allowing the determination of γ and ρ from a knowledge of the mean and mean-square distances separating trigger and target have been derived. Since mean and mean-square calculation requires little storage, this represents a memory-efficient alternative to a direct fit of the conditional

[35] In practice ρ was constrained always to exceed 10^{-3}, and hence μ_g was clamped to a maximum of 3000 (3 time constants).

[36] As $\varepsilon \to 1$, equation (117) approaches equation (116).

probability function (111) to measured binned data. In order to obtain closed-form expressions for the mean and mean-square, the exact distribution was approximated by one of the form:

$$\hat{p}(d) \;=\; \kappa\cdot\Big[\epsilon_1\cdot(1-P_1)^d\cdot P_1 + \epsilon_0\cdot(1-P_0)^d\cdot P_0\Big] \qquad (118)$$

where

$$\kappa\,(\epsilon_1+\epsilon_0) \;=\; 1 \qquad (119)$$

The following equations relate the parameters of the exact and approximate distributions; details of their derivation are shown in appendix G.

$$\Psi \;=\; e^{\frac{-\gamma}{(1-P_a-P_b)\cdot(1-e^{-\rho})}} \qquad (120)$$

$$P_0 \;=\; P_a + P_b \qquad (121)$$

$$\epsilon_0 \;=\; \frac{P_b\cdot\Psi}{P_a+P_b} \qquad (122)$$

$$P_1 \;=\; \frac{P_b+\gamma-\epsilon_0\cdot P_0}{\epsilon_1} \qquad (123)$$

$$P_1 \;=\; 1-(1-P_0)\cdot e^{\left[\frac{\rho\cdot\left[\left(P_b+\gamma\right)\cdot\ln(\Psi)-\gamma\right]}{P_b+\gamma-P_b\cdot\Psi}\right]} \qquad (124)$$

The values of P_a and P_b are calculated from the stream counts, using equations (113) and (117). In order to solve for ϵ_0, ϵ_1 γ and ρ, using the above equations, the measured mean $\overline{d}$ and mean-square $\overline{d^2}$ distance between trigger and target is employed. However, when estimated from data, these quantities have been found to be sensitive to outliers. In particular, it may happen that the trigger and target occur in unrelated parts of the training corpus and are consequently separated by large quantities of text[37]. Robustness is significantly improved by measuring the mean and mean-square within only a predetermined distance range, $d \in \{0\cdots N_T-1\}$. Expressions for the mean-and mean-square expected for such **truncated** measurements[38] under the independence assumption have been derived in appendix H. Since equation (118) is the superposition of two geometric terms, we may employ the results of this appendix and express the truncated mean $\overline{d}(N_T)$ and mean-square $\overline{d^2}(N_T)$ as a linear combination of the corresponding terms for truncated geometric distributions:

$$\overline{d}(N_T) \;=\; \kappa\cdot\left[\epsilon_1\cdot\mu(P_1,N_T)+\epsilon_0\cdot\mu(P_0,N_T)\right] \qquad (125)$$

and

$$\overline{d^2}(N_T) \;=\; \kappa\cdot\left[\epsilon_1\cdot\nu(P_1,N_T)+\epsilon_0\cdot\nu(P_0,N_T)\right] \qquad (126)$$

Using equations (119), (120), (121), (122), (123), (124), (125) and (126), we may calculate $\overline{d}(N_T)$ and $\overline{d^2}(N_T)$ given values for P_a, P_b, γ and ρ. Since it was not possible to solve explicitly for γ and ρ in terms of the other four parameters, however, their values were determined numerically by means of nested bisection searches.

[37] This is a particular problem when information regarding the segmentation of the corpus, such as article boundary markers, is not available.

[38] The geometric mean calculated using equation (115) was employed as a truncation interval in practice.

5.3.3. Typical estimates

Figure 5.2 repeats the curves of figure 5.1, and adds the plot of equation (111) using the parameters P_b, γ and ρ determined from the results of sections 5.3.1 and 5.3.2. The estimated conditional probability reflects the true nature of the data much more closely than that used under the independence assumption.

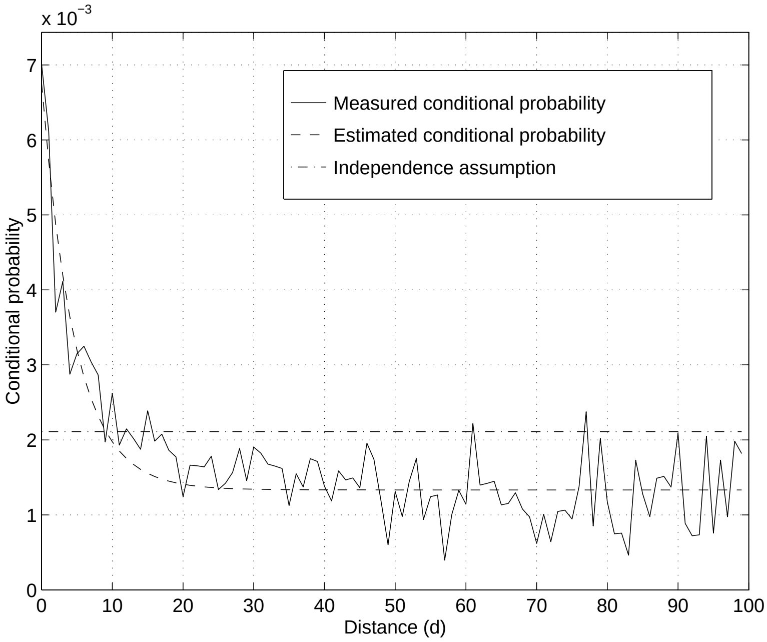

Figure 5.2: Measured and estimated conditional probability $P(w_j|v_k, d)$ of "*congress*" having seen "*president*".

5.4. Determining trigger-target pairs

While the number of possible self-triggers is bounded by the vocabulary size, the number of potential trigger-target pairs equals the square of this number, and it is not possible to consider these relations exhaustively except for very small vocabularies. In order to identify suitable candidates in a feasible manner, an approach employing two passes through the training corpus has been developed.

5.4.1. First-pass

This first stage of processing provides each potential target word in the lexicon with a **tentative-list** and a **fixed-list** for trigger candidates. The latter contains words for which a reliable correlation with the target has been established, while the former holds those for which no decision could yet be reached regarding the presence or absence of such relationship. The tentative-list includes storage for the cumulative totals required by distance-mean and -variance calculations, and is therefore more memory intensive than the fixed-list. At a certain point during processing, let the trigger-target pair have been seen N_m times at separations $\{d_0, d_1, \ldots, d_{N_m-1}\}$, each falling within the chosen truncation interval N_T, i.e.:

$$d_k < N_T \quad \forall \;\; k \in \{0, 1, \ldots, N_m - 1\}$$

so that the measured mean is given by:

$$\mu_{meas} \;=\; \frac{1}{N_m} \cdot \sum_{k=0}^{N_m-1} d_k$$

and the measured variance by:

$$\sigma^2_{meas} = \frac{1}{N_m - 1} \cdot \sum_{k=0}^{N_m - 1} (d_k - \mu_{meas})^2$$

Finally, assuming that the trigger and target occur independently, the expected mean is obtained from the truncated binomial distribution. Drawing on the results of appendix H we may therefore write:

$$\mu_{exp} \;=\; \frac{(1 - P_{tt}) \cdot \left[1 + (1 - P_{tt})^{N_T - 1} \left[(N_T - 1) \cdot (1 - P_{tt}) - N_T\right]\right]}{P_{tt} \cdot \left(1 - (1 - P_{tt})^{N_T}\right)}$$

where $P_{tt} = P_s(w_{trig}) + P_s(w_{targ})$ is the probability of occurrence of either trigger or target, calculated uniformly over the stream. Truncated mean- and variance-measurements are used to reduce the sensitivity of the measurements to outliers. The statistics for members of the tentative list are updated on each sighting the associated target during the sequential processing of the corpus. After each such update two hypothesis tests, termed the **kill**- and **fix**-tests respectively, are used to decide upon the strength of the correlation. Since both the mean and the variance are measured from the data, and since empirical observation of the samples d_k has indicated that they possess approximately normal distributions, the *t*-test [18] has been employed to make this decision in the following manner:

- **The kill-test.**

  When the measured mean μ_{meas} is found to exceed the expected mean μ_{exp} by a specified margin δ_{kill} and to a confidence of $(1 - \alpha_{kill})100\%$, the kill-test succeeds and the trigger candidate is deleted from the tentative list. In particular, let:

  $$\mu_{kill} \;=\; \mu_{exp} \cdot (1 + \delta_{kill})$$

  The critical value μ_t of the mean is then:

  $$\mu_t \;=\; \mu_{meas} - t(\alpha_{kill}, N_m - 1) \cdot \frac{\sigma_{meas}}{\sqrt{N_m}}$$

  where $t(\alpha_{kill}, N_m - 1)$ is the value obtained from the t-distribution for confidence $(1 - \alpha_{kill})100\%$ and $N_m - 1$ degrees of freedom. The kill-test succeeds when $\mu_t > \mu_{kill}$, and figure 5.3 illustrates these conditions.

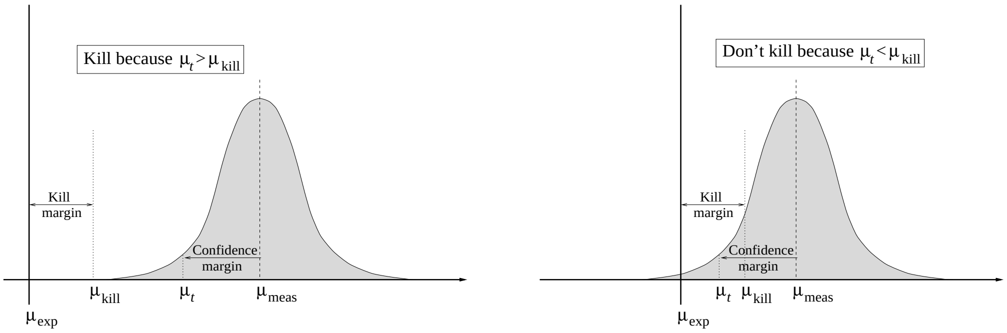

Figure 5.3: Illustration of the kill test.

- **The fix-test**

  When the expected mean μ_{exp} is found to exceed the measured mean μ_{meas} by a specified margin δ_{fix} and to a confidence of $(1-\alpha_{fix})100\%$, the fix-test succeeds and the trigger candidate is moved from the tentative- to the fixed-list. In particular, let:

  $$\mu_{fix} \;=\; \mu_{exp}\cdot(1-\delta_{fix})$$

  The critical value μ_t of the mean is then:

  $$\mu_t \;=\; \mu_{meas}+t(\alpha_{fix},N_m-1)\cdot\frac{\sigma_{meas}}{\sqrt{N_m}}$$

  where $t(\alpha_{fix},N_m-1)$ is the value obtained from the t-distribution for confidence $(1-\alpha_{fix})100\%$ and N_m-1 degrees of freedom. The fix-test succeeds when $\mu_t<\mu_{fix}$, and figure 5.4 illustrates these conditions.

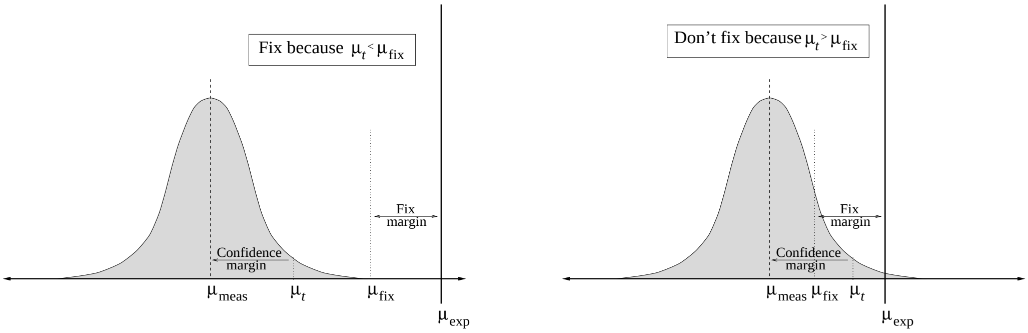

Figure 5.4: Illustration of the fix test.

The kill-test allows unpromising candidates to be pruned continually from the tentative list, thereby counteracting the explosion in the number of considered word-pairs that otherwise arises. Figure 5.5 illustrates this by showing the growth in the number of tentative triggers when the kill-test is disabled and when it is active.

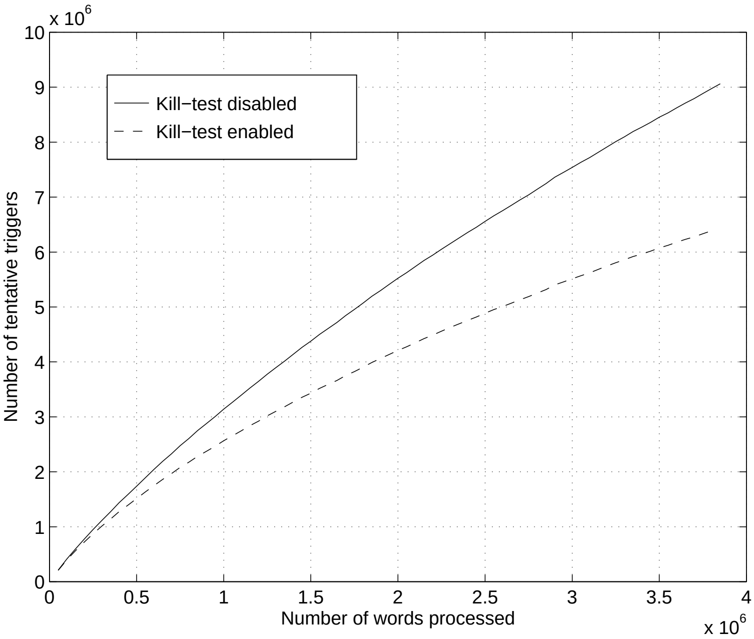

Figure 5.5: The effect of the kill-test on the total number of tentative triggers.

Once a correlation has been established (and the fix-test succeeds), the trigger is moved from the tentative to the fixed list, and no further statistics need be gathered. Separate tentative- and fixed-lists are maintained since the latter can be made much more compact by not including storage for means or variances. This is extremely important in view of the generally very large number of trigger-target candidates considered during the first pass.

Initially all tentative- and fixed-lists are empty. Furthermore, a record of the $\mathcal{H}$ most recent unique words is maintained during processing. As each word in the corpus is processed sequentially, each member of this history is hypothesised as a possible trigger word, and for each such candidate the following processing steps are performed:

- If this is not the first sighting of the target since the trigger, then END.
- If the trigger is already in the fixed-list, then END.
- If the trigger is not yet in the tentative-list, then:
  - Add it to the tentative-list.
  - Initialise the cumulative sum-of-distance and sum-of-squared-distance fields with the first measurement.
  - END.
- If the trigger is already in the tentative-list, then:
  - Update the sum-of-distance and sum-of-squared-distance with the new measurement.
  - Calculate the distance mean and variance.
  - Calculate the expected mean under the independence assumption.
  - Perform KILL and FIX *t*-tests :
    - ∗ Case A: The measured mean exceeds the independence-mean by a desired margin and to a desired level of confidence; conclude that there is no correlation.
      ⇒ **KILL**: remove the trigger from the tentative list.
    - ∗ Case B: The measured mean is lower than the independence-mean by a desired margin and to a desired level of confidence; conclude that there is a correlation.
      ⇒ **FIX**: remove the trigger from the tentative list and add it to the fixed-list.
    - ∗ Case C: neither of the above; conclude that there are as yet insufficient data to reach a decision; do nothing.
  - END.

The result of the first-pass is the set of all trigger-target relations present in the fixed-lists on completion, those remaining in the tentative-lists being discarded.

5.4.2. Second-pass

Since the fix-test in the first-pass uses means and variances often gathered only over a small portion of the training set, the detected correlations may be due to local anomalies which do not generalise to the corpus as a whole. Consequently the second-pass recalculates the means and variances for all candidates over the entire training corpus, and again applies the fix-test to each, discarding those which fail. Finally, the measured means and mean-squares of the remaining candidates are used to calculate the parameters P_b, γ and ρ of the postulated conditional probability function.

5.4.3. Regulating memory usage

Selection of the fix-test margin and confidence level allows the rate of transferrals from the tentative- to the fixed-list to be regulated, and thus provides control over the growth and the final number of fixed-triggers. The kill-test parameters, on the other hand, affect the rate of deletions from the tentative-list. Finally, the length of the history $\mathcal{H}$ determines the rate of addition of new tentative trigger candidates for the current target.

The size of the tentative-list is of prime practical importance during first-pass processing, since each entry requires significantly more storage than in the fixed-list. Despite the control over its size afforded by the choice of $\mathcal{H}$ and the kill-test parameters, it may still be difficult to limit the number of trigger-target pairs considered to practical levels. The following two refinements are employed as additional measures in this regard.

1. **Exclusion lists**

   Semantic correlations may be expected chiefly among content words, and since the grammatical functions of words are known, it is possible to exclude non-content words from consideration as triggers or targets during processing. Practically this is achieved by means of an **exclusion-list** containing all grammatical categories that should be disregarded in this way.

2. **Background culling**

   Observations during first-pass processing have shown a large number of tentative-list members to be predominantly idle. These are infrequent trigger-target candidates which, once added to the list, are neither fixed nor killed due to an insufficient number of measurements and long periods between updates. In order to reduce the number of these cases, a process termed **background culling** has been introduced. During processing, the distance to the last update is monitored for members of the tentative list, and the decision boundary for the kill-threshold is moved gradually towards that of the fix-threshold as this time increases. This relaxes the kill-test and ultimately forces a kill decision. The rate at which this occurs is normalised with respect to the frequency of the trigger, so that a single global parameter may be used to set the severeness of pruning.

   Background culling is an approximation necessitated by practical considerations, and generally introduces errors by eliminating valid but infrequent trigger-target relations. However it allows the size of the tentative list to be regulated to practical levels for large corpora and vocabularies, as illustrated in figure 5.6. Furthermore, the number of trigger-target pairs remaining after the second-pass appears in practice not to be affected strongly by the introduction of background culling. In particular, tests with the LOB corpus showed the number of tentative triggers after the first pass to be reduced by 67 % , while the number of fixed-triggers surviving the second-pass fell by only 3% . However, these figures depend on the fix- and kill-thresholds chosen during each pass.

5.4.4. Example pairs

Table 5.1 lists some examples of typical targets and their triggers found by the described technique when applied to the LOB corpus. The bracketed designations are the grammatical categories of the words in question[39]. It is appealing to find such intuitive relationships in meaning between word pairs gathered according to purely statistical criteria.

[39] JJ = adjective, NN = common noun, NNS = plural common noun, NNU = unit of measurement, NP = proper noun, NR = singular adverbial noun, VB = verb base form, VBN = past participle.

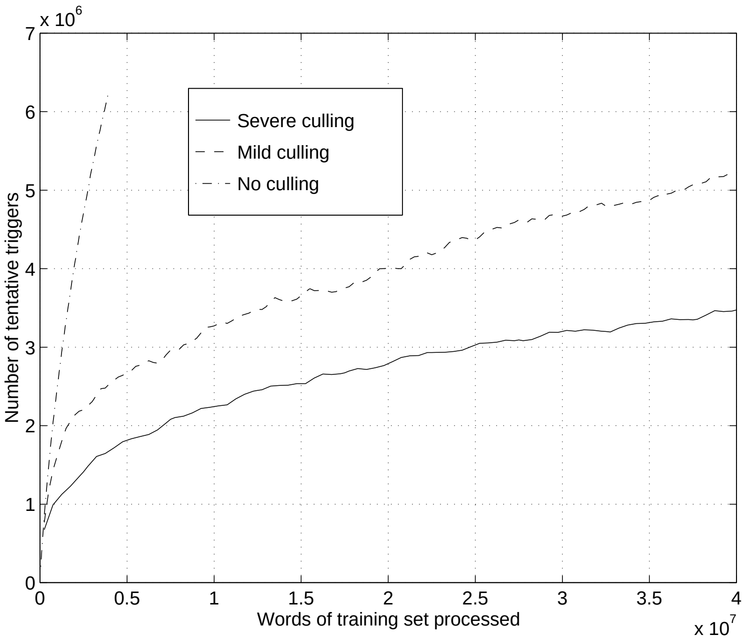

Figure 5.6: The effect of background culling on the total number of tentative triggers.

Target	**Triggers**
discharged (JJ)	prison (NN), period (NN), supervision (NN), need (NN), prisoner (NN), voluntary (JJ), assistance (NN)
advocate (NN)	truth (NN), box (NN), defence (NN), honest (JJ), face (VB), case (NN), witness (NN), evidence (NN)
Cambridge (NP)	university (NN), educational (JJ), affected (VBN), Oxbridge (NP), tomorrow (NR), universities (NNS)
worked (VBN)	demand (NN), changes (NNS), cost (NN), strength (NN)
dry (JJ)	currants (NNS), suet (NN), teasp. (NNU), wines (NNS), raisins (NNS)
judicial (JJ)	legal (JJ), binding (JJ), rules (NNS)
semiindustrialised (JJ)	world (NN), substantial (JJ), fall (NN), trade (NN), demand (NN), supply (NN)
cinema (NN)	directors (NNS), viewing (NN), film (NN), festival (NN), tastes (NNS)
current (NN)	inductance (NN), constant (NN), capacitor (NN), voltage (NN),
drowning (NN)	respiration (NN), failure (NN), inhaled (VBN), body (NN), spasm (NN), sea (NN), salt (JJ), minutes (NNS), lethal (JJ), water (NN), resuscitation (NN), recovery (NN), asphyxia (NN), survival (NN)
rotor (NN)	r.p.m. (NNU), values (NNS), blade (NN), pitching (NN), speed (NN), flapping (NN), wind (NN), tunnel (NN), helicopter (NN), body (NN), rotors (NNS)
syntax (NN)	language (NN), categories (NNS), formal (JJ), syntactic (JJ), grammatical (JJ), morphology (NN)
transfusion (NN)	bleeding (NN), blood (NN), cells (NNS), ml (NNU), reaction (NN), haematoma (NN), transfusions (NNS), patient (NN), group (NN), treated (VBN)
increases (NNS)	salary (NN), agreement (NN), salaries (NNS)
raisins (NNS)	list (NN), lemon (NN), milk (NN), salt (NN), teasp. (NNU), brandy (NN), mixed (JJ), currants (NNS), suet (NN), sugar (NN), nutmeg (NN), oz (NNU), sultanas (NNS), eggs (NNS), peel (NN), apples (NNS)
Orpheus (NP)	Heurodis (NP), Orfeo (NP), tale (NN), fairy (NN), Eurydice (NP)
Verwoerd (NP)	policy (NN), Africa (NP), South (NP)

Table 5.1: Examples of triggers and targets collected from the LOB corpus.

5.5. Perplexity results

The benefit of characterising trigger pairs as described in the previous sections was gauged by comparing the performance of a category-based language model employing the independence assumption with another using equation (111) but identical in all other respects. Experiments were carried out for the LOB and Wall-street Journal (WSJ) corpora[40]. Category-based language models were constructed for each corpus using respective pruning thresholds of $\lambda_{ct} = 5e-6$ and $\lambda_{ct} = 2e-7$ during construction of the variable-length category *n*-grams. Word-pair distances were not calculated across article boundaries.

The details of the language models constructed for each of these corpora are summarised in table 5.2. Information for a standard trigram language model using the Katz backoff and Good-Turing discounting [41] is given in order to establish a baseline. The symbols N_w, N_{wng} and N_{cng} refer to the number of words in the vocabulary, the number of *n*-grams in the trigram, and the number of *n*-grams in the category language model respectively. The number of self-triggers and trigger-target pairs for which parameters were estimated[41] are indicated by N_{st} and N_{tt}.

Corpus	N_w	N_{wng}	N_{cng}	N_{st}	N_{tt}
LOB	41,097	1,142,457	44,380	17,705	7,568
WSJ	65,000	13,047,678	934,894	49,952	83,696

Table 5.2: Language models and word-pair relations for the LOB and WSJ corpora.

Table 5.3 shows perplexities (PP) for the the trigram (TG) and category model (CM), and then for the category model with self-triggers (CM+ST), with trigger-target pairs (CM+TT), and lastly with both self-triggers and trigger-target pairs (CM+ST+TT).

Corpus	**TG**	**CM**	**CM+ST**		**CM+TT**		**CM+ST+TT**	
			PP	%	PP	%	PP	%
LOB	413.1	443.3	369.2	15.6	439.9	0.8	368.9	16.8
WSJ	144.4	450.5	380.1	16.7	437.4	2.9	368.4	18.2

Table 5.3: Perplexities when including word-pair relations in the category-model.

Finally, the following table shows the perplexities (PP) obtained when the word-to-category backoff language models (WTCBO) developed in chapter 4 are used in conjunction with a category model employing self-trigger and trigger-target pairs.

Corpus	**TG**	**WTCBO**		**WTCBO+ST**	**WTCBO+ST+TT**	
		Total *n*-grams	PP	PP	PP	Total%
LOB	413.1	138,678	368.9	291.1	291.0	29.5
WSJ	144.4	12,023,129	144.4	128.5	128.1	11.0

Table 5.4: Perplexities when including word-pair relations in the word-to-category backoff language model.

[40] Descriptions of the corpora many be found in appendix D.

[41] Function words were excluded by means of a suitable exclusion list, as described in section 5.4.3.

From table 5.4 we see that considerable reductions are obtained by combining the three approaches described in chapters 3, 4 and 5 respectively. Note that, for the LOB corpus, the perplexity is reduced by almost 30% with respect to to the word-based trigram, while using only 12% as many parameters.

5.6. Discussion

The largest perplexity improvement is obtained for the WSJ corpus, for which the largest number of self-trigger and trigger-target pairs were collected. This stems from the much greater corpus size and consequent lower sparseness. For LOB on the other hand, many words occur too infrequently to make estimation of the conditional probability parameters possible, thus leading to a reduced number of word relations.

For both corpora, the addition of self-triggers has a more significant impact on the perplexity than does the introduction of trigger-target pairs. This agrees with other reports in the literature [47], [53]. Self-triggers seem more reliable since the target, being its own trigger, is actually seen before being predicted to occur again. Trigger-target pairs, on the other hand, predict words that have either not yet been seen at all or have occurred in the distant past. Since such correlations are heavily dependent upon the topic of the passage, the effectiveness of a trigger-target association depends on how much the topics associated with a trigger coincide between the training- and test set. For the LOB corpus, which is very diverse in the material it contains, there is a significant mismatch in this regard, leading to the observed very small impact of self-triggers on performance. For the WSJ corpus the mismatch is smaller, leading to greater success.

Performance improvements obtained using the exponential decay self-trigger model (111) appears to compare favourably with those obtained using a cache component, which has a similar philosophy, but usually does not allow the cache probability to decay with distance in a comparable way. In [45], for example, the addition of a cache component decreases the perplexity of a part-of-speech language model by 14% for the LOB corpus. In our experiments, the addition of self-triggers causes this figure to fall by 16.7% (table 5.3). For the WSJ corpus, perplexity improvements of between 14% and 17% have been achieved [24], [70], though these are harder to compare directly with our results since they include also the effect of a bigram cache. Little information has been found in the literature regarding the performance of language models incorporating trigger-target pairs. Experiments in [53] found that their addition by means of equation (64) improves only the perplexity of a unigram language model. Although small, the improvements shown in table 5.3 are nevertheless promising in this respect. The work reported in [47] and [79] unfortunately does not show the individual impact on performance made by the cache, trigger-target pairs, or other added components. This is probably due to the numerically very intensive maximum entropy technique used to combine these additional knowledge sources with the baseline trigram language model. Finally, recent work affirms that also the performance of a standard cache component may be enhanced by allowing the probabilities to decay exponentially [16].

The addition of self-triggers increases the number of parameters in the category-based model by $2 \cdot N_{st}$ (storage of γ and ρ). This increase is mild, and offers a favourable size versus performance tradeoff. For instance, the category model with self-triggers for LOB uses 62,085 parameters and achieves a lower perplexity than the word-based trigram with 1.1 million parameters. Furthermore, the effectiveness of both types of word-pair modelling improves with corpus size, and since the parameter determination and final implementation of the model has low memory requirements, the technique is suitable for use with large training sets. This complements the category-based model, for which performance does not improve in the same way.

Finally, inspection of the values of ρ assigned to trigger-target pairs, as well as cases in which a trigger successfully predicts a target, shows that correlations well beyond the range of conventional *n*-gram models are captured. Hence the proposed technique is indeed able to model long-range dependencies. This may also be deduced from the further reductions obtained when adding word-pair relations to the category-based model used within the word-to-category backoff framework (table 5.4). In particular, these improvements indicate that each of the three components contribute different information to the model as a whole. The final 3-component models exhibit significantly lower perplexity than the baseline word trigram. This is true for both the LOB as well as the larger WSJ corpus, although the figure of 29.5% for the former is particularly striking, and indicates that the imposition of the structure advocated in section 1.3 is particularly effective when the training set is sparse.

5.7. Summary and conclusion

A new technique for modelling the empirically observed transient character of the occurrence probability between related words in a body of text has been introduced. Procedures both for the identification of such word pairs as well as for the estimation of the three parameters required for the parametric model have been developed. Experiments demonstrate that meaningful relations are indeed identified, and that the transient behaviour (which often spans many words) is successfully captured by the proposed model. Perplexity reductions of between 16.8 and 18.2% were achieved, with the greatest improvement for the largest and least-sparse corpus. Words correlated with themselves (self-triggers) had the most significant impact on performance. The modelling technique is able to reduce the performance limit displayed by category-based models for large corpora, while maintaining their good performance versus size trade-off. Finally, when integrated into the word-to-category backoff framework, further improvements are achieved, allowing the baseline trigram language model perplexities to be surpassed. This reduction was particularly pronounced for the smaller LOB corpus, totalling 29.5% .

Chapter 6

Word error rate performance

Perplexity is a popular measure of language model performance due to its lower computational demands when compared with a complete recognition experiment. However, reductions in perplexity do not guarantee reductions in recognition word error rates, which remain the ultimate measure of language model quality. The following sections introduce methods by means of which language models may be incorporated into the recognition search, and finally present recognition results obtained for a Wall Street Journal task using the language models developed in this thesis.

6.1. Language models in the recognition search

Ideally a connected speech recogniser should search the space of all possible concatenations of words to find the most likely combination with respect to the acoustic evidence. However, even for small vocabularies this exhaustive approach may be impractical, and thus more refined methods are necessary. It is important from a practical standpoint to understand the mechanism of the decoding process, since the nature of the language model may heavily influence the complexity of the search problem. In particular, various alternative means of language model application are available, some of which are illustrated in the remainder of this section. Consider as an example a speech recogniser which must recognise any 3-word sentence containing the words “he”, “says” and “hello”. The set of possible sentences may be visualised as the tree structure depicted in figure 6.1.

For connected speech the times at which word boundaries occur are not known. Since these affect the probabilities obtained from the acoustic models by determining which segments of the observation sequence are assigned to which model, each possible choice of word-boundaries must be considered to be a separate hypothesis by the recogniser. The total number of distinct hypotheses considered in the search is therefore much larger than the number of sentences in the tree shown in figure 6.1, and hence it should come as no surprise that, even for small vocabularies, the search space is in practice too large to be considered exhaustively. The size of the search space is limited by practical constraints such as processing time and available storage, and hence the following two techniques are instrumental in making recognition practical [66].

1. **Path-pruning** : As the search proceeds, certain hypotheses will become very unlikely, and may be discarded to save computation and storage. However, since it cannot be guaranteed that one of these paths is indeed the correct result (becoming much more probable later on), pruning may lead to **search errors**.

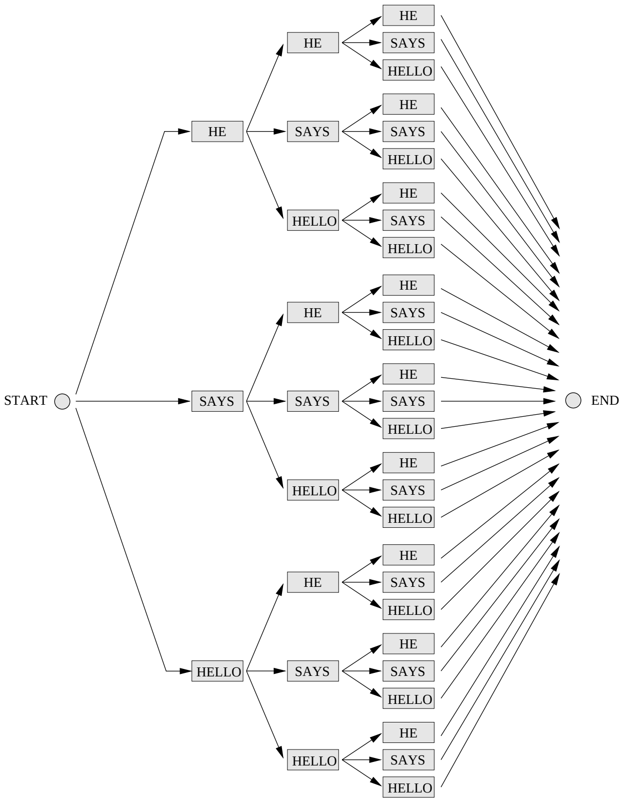

Figure 6.1: A tree-representation of sentences to be considered in the search.

2. **Path merging** : When two (or more) paths meet, it may be possible to merge them and consider only a single continuation. However, whether this is permissible depends on the extent to which the computation involved in extending each path is dependent on its history. In particular, paths may merge only when their histories are considered equivalent from a computational point of view. Since language models generally make use of more context than acoustic models[42], they often dictate the points at which merges may occur. In particular, merges may occur when the respective history equivalence classifications (discussed in section 2.2) match. Consider again the example of figure 6.1. The language model probabilities are calculated at word boundaries, so when a bigram is used, the search space illustrated in figure 6.1 is reduced to that shown in figure 6.2.

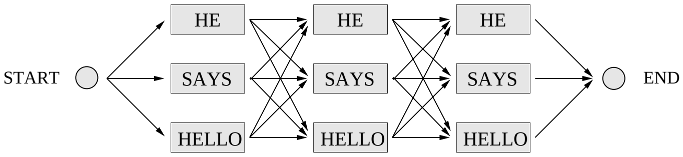

Figure 6.2: The search space when using a bigram language model.

When a trigram is employed, however, the language model probability calculation is based on the most recent word pair, and paths can merge only when exiting the same word as illustrated in figure 6.3. Path merging allows very substantial reductions in the number of separate paths requiring consideration at any one time.

[42] When context-independent or word-internal context-dependent phone models are employed, the calculation of acoustic probabilities is dependent only upon the current word. However, when cross-word context-dependent models are used, these calculations may be influenced by the identity of one or more phones preceding the word boundary. Nevertheless, the information used by the language model generally exceeds this.

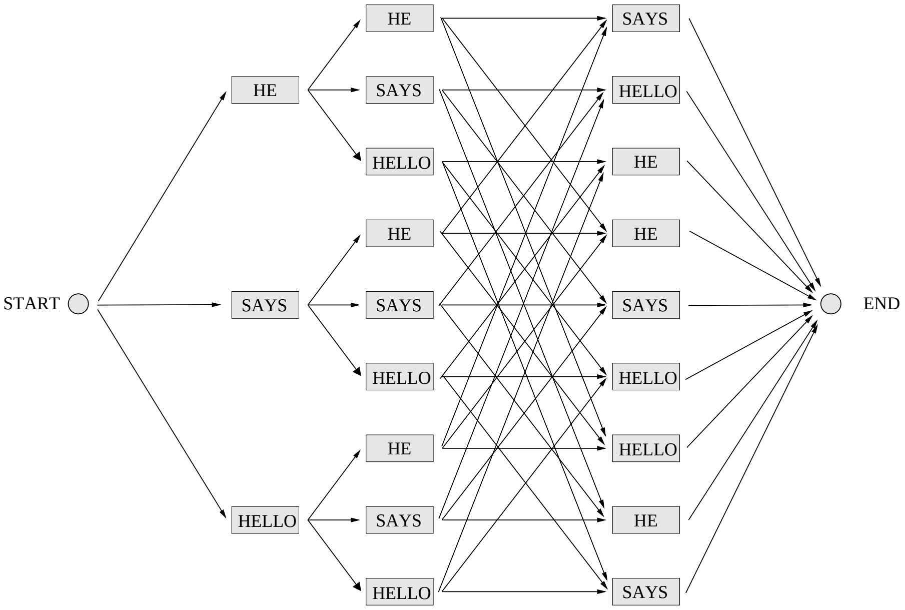

Figure 6.3: The search space when using a trigram language model.

For trigram and longer-span language models, the number of paths may become unmanageably large even with efficient path merging. Containing the search would require more severe pruning, which might unacceptably increase the number of search errors. In such cases we may adopt the following two-pass strategy:

1. Apply a simpler (e.g. bigram) language model during the recognition search, and output a subset of the search space containing a number of the most likely paths, termed the **intermediate hypotheses**.
2. Post-process, or **rescore**, the intermediate hypotheses by applying the more sophisticated language model, and pick the most likely as the final recognition result.

As long as the result that would be obtained after a recognition search using the more sophisticated language model is among the intermediate hypotheses, the final result will remain unaffected by the division of the recognition process into two stages. However, when the first-pass eliminates this hypothesis, a search error occurs which the second-pass is unable to correct. The **accuracy**[43] of the intermediate hypotheses is affected both by the exactness of the first pass as well as the size[44] of the intermediate hypotheses, and is an important issue when taking the two-pass approach.

Two forms of intermediate hypotheses, **N-best lists** and **lattices**, are described in the following sections.

6.1.1. N-best rescoring.

The N-best method [82] determines during the recognition search a list containing the N most likely hypotheses, together with the acoustic and language model probabilities for each word. The new language model is used either to replace or to modify the existing probabilities for each hypothesis, after which the overall likelihood is recalculated. The recognition result is the highest ranking hypothesis in this new list.

[43] The proportion containing the result of a full recognition search.
[44] The number of alternatives.

6.1.2. Lattice rescoring.

Instead of a list, the recognition search may output a network containing the most likely paths in the recognition search space. A **lattice** is a directed acyclical graph in which the nodes correspond to word boundaries in time, and the links between the nodes correspond to particular word hypotheses. Acoustic and language model likelihoods for each word are stored in the lattice, and each path from the start to the end node represents a distinct hypothesis. The most likely path through the lattice may be found using a suitable search algorithm, A* having been applied successfully [66], [67]. A lattice is a much more compact representation of a set of alternatives than an N-best list, and in certain situations the lattice search may be performed very efficiently. However, a prerequisite is that significant path recombination is possible, else the number of separate paths maintained during the lattice search may become unmanageable. For N-best lists on the other hand, the storage and computational effort required during rescoring is well defined *a-priori*. Note that N-best lists may be generated from lattices.

6.2. Recognition experiments

This section presents recognition results in terms or word error rates obtained using the language models developed in this thesis. The experiments were conducted on the WSJ corpus by using lattices generated at Cambridge University with the HTK large-vocabulary speech recognition system as part of the November 1994 ARPA CSR evaluation [91]. The HTK recogniser uses mixture Gaussian cross-word context dependent hidden Markov acoustic models and allows scoring in a single pass, while incorporating an *n*-gram language model to deliver state-of-the art performance. Lattices produced for the 65,000 word vocabulary 1994 H1 development test were rescored using the baseline bigram and trigram language models described in appendix D. The resulting trigram lattices were used to generate N-best lists, which were subsequently rescored by the new language models to obtain the final recognition results, as described in section 6.1.1. The following section presents the baseline recognition performance, while ensuing sections show how this performance changes on application of the various language models. Language model scaling factors used in the rescoring were optimised approximately for the test set in all cases, including the baseline.

6.2.1. Baseline results

There are 310 lattices in the development test set, comprising approximately 15 sentences from each of 20 speakers, and leading to a total of 7,388 words in the reference transcription. The out-of-vocabulary rate of the 65,366 word vocabulary, with respect to the reference transcription, is 0.28% . Since the original lattices were constructed using a language model trained on substantially more data than WSJ 87-89, they were rescored using the bigram and trigram language models described in appendix D to obtain baseline recognition accuracies, and these are shown in the following table. Perplexities are shown both for the language model (LM) test set (appendix D), as well as the H1 development test reference transcription.

	Perplexity		% **Word error**
	LM test set	H1 dev-test ref.	
Baseline bigram	215.4	275.6	13.66
Baseline trigram	144.4	204.3	11.84

Table 6.1: Baseline performance for the 1994 H1 development test set.

Changes in system performance after application of the language models will be measured relative to the baseline trigram error rate of 11.84% . To obtain an indication of the best possible recognition performance obtainable with the lattices and N-best lists, the error rate for each is shown in table 6.2.

	Error rate
Lattices	1.57 %
N-best (N=100)	5.36 %

Table 6.2: Lower bounds on word error rates.

Since all recognition experiments in the following sections will rescore the 100 best hypotheses of each of the lattices, the lowest achievable word error rate is 5.36% .

6.2.2. Rescoring results

The Entropic Lattice and Language Modelling Toolkit [67] was employed as an interface allowing convenient N-best rescoring with the new language models. In particular, the tools were used to determine the 100 best hypotheses from each lattice, to rescore these using the new language models, to re-rank the result, and finally to output the best hypothesis in the rescored list. The new language models were either used to replace the baseline language model probabilities entirely, or were combined with the baseline trigram by linear interpolation. Various combinations of the language model components presented in this thesis were used in the rescoring process, and results for each are shown in the following sections.

6.2.2.1. Category model

The variable-length category-based *n*-gram model developed in chapter 3 and constructed with pruning threshold $\lambda_{ct} = 2e - 7$ was used to rescore the N-best lists by means of linear interpolation with the baseline trigram probabilities, which are already present in the lattice. The following table summarises the results obtained for various interpolation conditions, perplexities again being shown for both the test set described in appendix D as well as the reference transcription. The interpolation parameter weights the category-model, and thus a value of zero corresponds to rescoring using the baseline trigram only.

Weight	**Perplexity**		**% Word error**	**% Improvement**
	LM test set	H1 dev-test ref.		
0.0	144.4	204.3	11.84	-
0.15	134.6	183.2	11.64	1.7
0.25	135.4	181.7	11.45	3.3
0.5	147.5	192.3	11.44	**3.4**
0.75	180.4	228.1	11.76	0.7
1.0	450.5	508.6	12.49	-5.5

Table 6.3: Rescoring using the category-based model interpolated with the baseline trigram.

Interpolation is able to reduce the perplexity of the baseline trigram by approximately 7 % on the test set and 11 % on the reference transcription, while resulting in a 3.4% improvement in the error rate. Note however that optimum performance is not achieved at the lowest perplexity on the test set. Similar observations have been made in [21] and [80]. Performance appears to be relatively insensitive to the interpolation weight while this is in the range 0.25 to 0.5.

6.2.2.2. Word-to-category backoff model

The word-to-category backoff models developed in chapter 4 were used to rescore the N-best lists by replacing the lattice scores entirely. The category model constructed with a pruning threshold of $2e-7$ was used, and experiments were carried out for word-to-category backoff models of different complexities, as shown in the following table.

Number of *n*-grams	**Perplexity**		**% Word error**	**% Improvement**
	LM test set	H1 dev-test ref.		
12.0M	144.4	197.8	11.52	2.7
11.6M	144.8	198.2	11.53	2.6
6.7M	151.3	204.5	11.69	1.3
1.0M	238.0	298.5	12.09	-2.1

Table 6.4: Rescoring using only the word-to-category language model.

These results show a word error rate improvement of 2.6% for the most complex model (which retains most of the word model's *n*-grams), but smaller improvements and eventual deterioration in performance when the number of parameters is reduced. Better performance both in terms of perplexity and word error rate is achieved by linear interpolation with the baseline, as was shown in table 6.3.

6.2.2.3. Category model with long-range correlations

Since the word-pair relations described in chapter 5 require the determination of their separating distance, a record must be kept of the document history. This is achieved by retaining the best recognition output for each lattice to the beginning of the current article, while processing the lattices in the order they were spoken. The following table shows perplexities and recognition results obtained when incorporating first self-triggers (ST) and then also trigger-target (TT) pairs into the category model used in section 6.2.2.1. Linear interpolation with a weight of 0.5 was employed in the rescoring process.

Configuration	**Perplexity**		**% Word error**	**% Improvement**
	LM test set	H1 dev-test ref.		
Category + ST	127.3	161.8	11.22	5.2
Category + ST + TT	126.7	160.8	11.18	5.6

Table 6.5: The category-based model with word-pair correlations interpolated with the baseline trigram.

The linear interpolation of the category-based model including word-pair relations has decreased the perplexities with respect to the baseline by 12.3 % and 21.3 % on the test set and reference transcription respectively, and has led to a word error rate improvement of 5.6% . This improvement has been verified to be significant at the 5% level[45] using a NIST scoring package [63]. As is the case for the perplexity, the largest part of the reduction in word error rate has been brought about by the addition of self-triggers.

6.2.2.4. *Word-to-category backoff model with long-range correlations*

Finally, the word-to-category backoff (WTCBO) model with 12.0 million parameters was used to rescore the N-best lists as in section 6.2.2.2, however now employing the category-based model with word-pair relations evaluated on its own in the previous section. Perplexities and recognition results when including both self-triggers (ST) and trigger-target (TT) pairs are shown in the following table.

Configuration	**Perplexity**		% **Word error**	% **Improvement**
	LM test set	H1 dev-test ref.		
WTCBO + ST + TT	128.1	167.5	11.59	2.1

Table 6.6: Rescoring using the word-to-category backoff model with word-pair correlations.

Although the perplexities are similar to those reported in table 6.5, improvements in the word error rate are smaller. Furthermore, it is disappointing to note that the addition of the word-pair relations has led to a deterioration of the word-to-category backoff model, results for which were shown in table 6.4.

6.3. Summary and conclusion

The use of the category-based language model in the N-best rescoring framework has led to improvements in recognition word error rate, when applied either by linear interpolation with the lattice-internal trigram probabilities or by means of the word-to-category backoff method. Subsequent addition of long-distance word correlation models lead to further improvements when interpolating the category model with the baseline trigram, but not in conjunction with the word-to-category backoff method. Since the baseline trigram language model probabilities are already present in the lattice, linear interpolation is both simpler to implement and less memory intensive, and should therefore be the method of choice for similar recognition problems. Interpolation with the category-based model employing both self-triggers and trigger-target pairs led to a relative word error rate improvement of 5.6% over the baseline, and this figure has been verified to be statistically significant.

[45] There is a less than 5% probability that the improvement is by chance.

Chapter 7

Summary and conclusions

This thesis has focussed on the use of linguistically-defined word categories as a means of improving the performance of statistical language models. In particular, an approach that aims to capture both general grammatical patterns as well as particular word dependencies by different model components was developed and evaluated.

7.1. Review of conducted work

The separate treatment of patterns due to syntax and patterns due to semantic relationships has led to three distinct sections in this document. The first section, in chapter 3, develops a model for syntactic dependencies based on word-category *n*-grams. The second section, in chapter 4, extends this model by allowing short-range word relations to be captured through the incorporation of selected word *n*-grams. Finally, a technique which permits also the inclusion of long-range word-pair relationships is presented in chapter 5.

7.1.1. The category-based syntactic model

The *n*-gram has proved a very successful model for short-term word dependencies. Noting that English grammatical constructs are often quite local in nature, *n*-grams of part-of-speech word-categories were adopted as a means of capturing general sequential grammatical patterns.

Since there are significantly fewer parts-of-speech than there are words in a typical vocabulary, these models contain a much smaller number of *n*-grams than a comparable word-based model. This reduces the sparseness of the data with respect to the number of parameters, which in turn allows us to capture longer-range effects by increasing n. Furthermore, an important advantage of category *n*-grams is their intrinsic ability to generalise to word *n*-tuples not seen during training. Part-of-speech categories embed syntactic information, and hence this generalisation proceeds according to the measure of grammatical correctness assigned to the unseen sequence by the model.

Chapter 3 proposes, develops, and evaluates a model employing category-based *n*-grams of variable length. Each word may belong to more than one category in order to account for different grammatical functions. The length of individual *n*-grams is optimised by allowing it to increase as long as further extension benefits the overall predictive quality. In order to avoid overfitting of the training set, the criterion used to estimate this predictive quality is based on leaving-one-out cross validation. The language models produced in this way are seen to contain *n*-grams whose length varies between one (unigram) and a maximum of four to fifteen, depending on the nature and amount of training material, as well as the parameters used during model construction.

Words may belong to several part-of-speech categories to account for multiple grammatical functions. Therefore, when calculating the probability of a particular word in a sentence based on the words seen before, the language model must take into account the many possible category sequences that correspond to this word history. An algorithm by means of which this may be achieved for the variable-length category-based model was developed in chapter 3. This algorithm consists of a recursive procedure that maintains a set of possible category sequences for the word history, with an associated probability for each. These probabilities have been employed also in assigning category classifications to new text, hence configuring the language model as a tagger. In this way 1.9 million words of transcribed telephone conversations and 37 million words of Wall Street Journal text were assigned part-of-speech tags, thus making it possible for category-based language models to be trained on both corpora.

Experimental evaluation has shown that, when used as a tagger, the variable-length category-based language model is able to deliver better results than a baseline system. Particularly significant improvements were achieved when tagging out-of-vocabulary words as a result of the proposed method for category-membership probability estimation based on cross validation. The algorithm developed to selectively extend the length of individual *n*-grams has led to better results than achieved by commonly used pruning methods based on count thresholds. Consequently, the variable-length models outperform conventional fixed-length approaches. Language model perplexities are seen to be competitive for sparse training sets, but not for larger ones, although large reductions in the number of model parameters are found in all cases. When interpolated with a word-based trigram, the category-based model has led to a 3.4% word error rate improvement for a Wall Street Journal recognition experiment within an N-best rescoring framework.

7.1.2. Inclusion of word *n*-grams

A language model based purely on category *n*-grams is not able to capture relationships between particular words, but only between the categories to which these words belong. In chapter 3, a detailed comparison between a conventional trigram and the variable-length category-based language model has shown that the former gives better estimates for word *n*-grams seen in the training set, but not in other cases. Word *n*-grams are an effective means of encoding short-term relationships between particular words, while category *n*-grams generalise to unseen word sequences and are therefore particularly appropriate in backoff situations. Chapter 4 develops a technique that allows a word-based *n*-gram language model to back-off to the category-based model presented in chapter 3, thereby combining the strengths of both approaches. An exact formulation would require an excessive number of backoff weights, due to the complex representation of the word history used by the category model in its probability calculations. Hence, an approximate model is developed which continues to employ the category model in backoff situations, and ensures correct normalisation by approximating the word-model probabilities. Furthermore, a procedure is proposed which selects only the most important word *n*-grams for inclusion into the word model, thus allowing the overall number of parameters to be traded for modelling accuracy. Experiments show these methods to deliver greatly reduced perplexities for sparse training sets, and significantly improved size versus performance tradeoffs when compared with standard trigram models, even for large corpora. Recognition experiments performed within an N-best rescoring framework show that the proposed technique fares approximately as well as a linear interpolation of the category-based model with the baseline trigram.

7.1.3. Inclusion of long-range word-pair relations

An underlying assumption of the category-based model is that the probability of a word depends only upon the category to which it belongs, and that its occurrence is therefore equally likely at any point in a corpus at which this category occurs. However, factors such as the topic and the style of the text cause certain words to occur in groups, thereby violating this assumption. While the inclusion of word *n*-grams, as described in the previous section, allows short-term relationships to be taken into account, it is not possible to capture dependencies spanning more than 3 or perhaps 4 words in this way. In view of this, chapter 5 presents a technique by means of which long-range relationships may be considered. Central to the technique is a definition of the distance between words, made in terms of word categories. Empirical observations using this measure of distance indicate that the conditional probability of a word, given its category, exhibits an exponential decay towards a constant, rather than maintaining the uniform value normally assumed. Consequently, a functional dependence of the occurrence probability upon this separation is postulated. Methods are then developed for determining both the related word pairs, as well as the function parameters, from a large corpus. Using these methods it has been possible to identify word-pairs that subjectively appear to be strongly related in semantic content, and which are often separated by many words in the text. Incorporation of these word-pair relations into the category-based language model leads to significant perplexity reduction on both sparse and large corpora. This is also true when employed within the word-to-category backoff scheme, demonstrating that additional information is captured by the word-pairs. When interpolating the category-based language model incorporating word-pair dependencies with a baseline word trigram, a significant 5.6% word error rate reduction was achieved for a Wall Street Journal recognition task employing the N-best rescoring framework.

7.2. Conclusion and topics for future investigation

In conclusion, the results obtained with the techniques proposed in this thesis have illustrated that grammatical word category classifications may be used to improve the performance of language models. This applies to large bodies of text, and especially to small or sparse corpora, for which conventional word-based *n*-gram language models are poorly trained. In particular, both category *n*-grams and word-pair relations convey useful information not captured by standard word-based *n*-gram language models. In this respect, the imposition of the classification scheme for word patterns advocated in section 1.3 has proved successful. Significant improvements were obtained even for a high-performance baseline recognition system, and hence further attention to the proposed approaches is warranted.

The developed techniques, and the manner in which they have finally been integrated into a speech-recognition system, leave room for refinement and further research. Some proposals in this respect are given below.

7.2.1. Tagging

The nature of the category definitions and the accuracy with which the words of the training corpus are tagged are key factors in the effectiveness of the category-based language modelling approach. Although the category-based language model used as a tagger in this work performs well in controlled tests, no further particular attention was given to the tagging operation itself, and it may be worthwhile investigating methods by means of which large quantities of new text may be tagged more accurately and in greater detail. A related topic is the use of larger training corpora, such as the British National Corpus [10], from which it should be possible to obtain a more reliable tagging language model.

7.2.2. Data-driven refinement of category definitions

At present the category definitions used by the syntactic model are fixed at the outset, based on a lexicon constructed from the training corpus, and on other information sources such as electronic dictionaries. While this takes advantage of *a-priori* grammatical knowledge in a clear and fundamental way, these definitions are not necessarily optimal for the modelling task at hand. In particular, for many members of a category the assignments are too general, since there are sufficient training data for better predictive discrimination to be attained with more refined definitions. Future research could therefore investigate automatic procedures for determining more refined categories. This would allow the extent to which the model generalises to be balanced against the extent to which it captures more detailed patterns. Hence it would become possible to optimise language model performance in accordance with the size and character of particular training corpora, as well as the ultimate application, in a well-defined and controlled way.

7.2.3. Punctuation

Although grammatically-motivated category definitions have been employed as a means of capturing syntactic patterns in text, punctuation has been ignored because it is generally not included in the output of a speech-recognition system. However, punctuation marks have grammatical function, and retaining them could improve the grammatical consistency of the training material. It may therefore be of benefit to investigate methods which allow the category-based model to include punctuation symbols, possibly by interaction with other features such as pitch, pauses, and stress in the acoustic signal.

7.2.4. Lattice rescoring

Recognition results have been presented using N-best rescoring. However, the accuracy of these lists is generally considerably lower than that of the lattices from which they were generated, and hence it would be better to allow the lattices to be rescored directly. This is not straightforward, due to the complex representation of the document history employed both by the category-based language model, and by the word-pair relations. Suitable approximations, and possibly specialised search strategies, therefore need to be investigated.

7.3. Final summary

A language modelling approach centred around grammatical word categories has been proposed, developed and evaluated. Variable-length *n*-grams of categories are employed to capture general grammatical patterns, while *n*-grams of words and long-range word-pair correlations are used to model semantic relationships between particular words. This language modelling approach has been shown experimentally to offer improved generalisation to previously unseen word sequences, while employing fewer parameters and offering better performance than standard word-based *n*-gram techniques. When incorporated into a high-performance baseline speech-recognition system, the proposed language models have led to significant improvements in word error rate.

References

[1] Antoniol, G; Brugnara, F; Cettolo, M; Federico, M; *Language model estimations and representations for real-time continuous speech recognition*, Proceedings of the International Conference on Spoken Language Processing, Yokohama, vol. 2, pp. 859-862, 1994.

[2] Bahl, L; Jelinek, P.F; Mercer, R.L; *A maximum likelihood approach to continuous speech recognition*, IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 5, no. 2, March 1983.

[3] Bahl, L; Brown, P; de Souza, P; Mercer, R. *A tree-based statistical language model for natural language speech recognition*, IEEE Transactions on Acoustics, Speech and Signal Processing, vol. 37, no. 7, July 1989.

[4] Berger, A.L; Brown, P.F; Della Pietra, S.A; Della Pietra, V.J; Gillett, J.R; Lafferty, J.D; Mercer, R.L; Printz, H; Ureš, L; *The Candide system fo machine translation*, Proceedings of the Human Language Technology Workshop, pp. 157-162, March 1994.

[5] Breiman, L; Friedman, J.H; Olschen, R; Stone, C.J; *Classification and regression trees*, Wadsworth and Brooks/Cole, 1984.

[6] Brew, C; Thompson, H.S; *Automatic evaluation of computer generated text: a progress report on the TEXTEVAL project*, Proceedings of the Human Language Technology Workshop, pp. 108-113, March 1994.

[7] Brown, P.F; Cocke, J; Della Pietra, S.A; Della Pietra, V.J; Jelinek, F; Lafferty, J.D; Mercer, R.L; Roossin, P.S; *A statistical approach to machine translation*, Computational Linguistics, vol. 16, no. 2, pp. 79-85, 1990.

[8] Brown, P.F; de Souza, P.V; Mercer, R.L; Della Pietra, V.J; Lai, J.C; *Class-based n-gram models of natural language*, Computational Linguistics, vol. 8, no. 4, 1992.

[9] Brown, P.F; Chen, S.F; Della Pietra, S.A; Della Pietra, V.J; Kehler, A.S; Mercer, R.L; *Automatic speech-recognition in machine-aided translation*, Computer Speech and Language, vol. 8, pp. 177-187, 1994.

[10] Burnard, L; *Users Reference Guide for the British National Corpus*, Oxford University Computing Services, May 1995.

[11] Byrne, W; *LM95 Switchboard*, Opening and Closing Day Reports for the 1995 Language Modelling Workshop (LM95), Centre for Language and Speech Processing, John's Hopkins University, Baltimore, July-August 1995.

[12] Chase, L; Rosenfeld, R; Ward, W; *Error-responsive modifications to speech-recognisers: negative n-grams*, Proceedings of the International Conference on Spoken Language Processing, Yokohama, pp. 827-830, 1994.

[13] Church, K.W; *A stochastic parts program and noun phrase parser for unrestricted text*, Proceedings of the Second Conference on Applied Natural Language Processing (ACL), pp. 136-143, 1988.

[14] Church, K.W; Hanks, P; *Word association norms, mutual information, and lexicography*, Computational Linguistics, vol. 16, no. 1, pp. 22 - 29, March 1990.

[15] Church, K.W; Gale, W.A; *A comparison of the enhanced Good-Turing and deleted estimation methods for estimating probabilities of English bigrams*, Computer Speech and Language, vol. 5, pp. 19-54, 1991.

[16] Clarkson, P.R; Robinson, A.J; *Language model adaptation using mixtures and an exponentially decaying cache*, Proceedings of the International Conference on Acoustics, Speech and Signal Processing, Munich, vol. 2, pp. 799-802, April 1997.

[17] Cutting, D; Kupiec, J; Pedersen, J; Sibun, P; *A practical part-of-speech tagger*, Proceedings of the Third Conference on Applied Natural Language Processing (ACL), pp. 133-140, 1992.

[18] Dougherty, E.R; *Probability and statistics for the engineering, computing and physical sciences*, Prentice-Hall, Englewood Cliffs, 1990.

[19] Duda, R., Hart, P. ; *Pattern classification and scene analysis*; Wiley, New York, 1973.

[20] Elworthy, D. *Tagger suite user's manual*, May 1993.

[21] Farhat, A; Isabelle, J.F; O'Shaughnessy, D; *Clustering words for statistical language models based on contextual word similarity*, Proceedings of the International Conference on Acoustics, Speech and Signal Processing, Atlanta, vol. 1, pp. 180-183, 1996.

[22] Federico, M; *Bayesian estimation methods for n-gram language model adaptation*, Proceedings of the International Conference on Spoken Language Processing, Philadelphia, vol. 1, pp. 240-243, 1996.

[23] Geutner, P; *Introducing linguistic constraints into statistical language modelling*, Proceedings of the International Conference on Spoken Language Processing, Philadelphia, vol. 1, pp. 402-405, 1996.

[24] Generet, M; Ney, H; Wessel, F; *Extensions of absolute discounting for language modelling*, Proceedings of Eurospeech, Madrid, pp. 1245-1248, 1995.

[25] Giachin, E; Baggia, P; Micca, G; *Language models for spontaneous speech-recognition: a bootstrap method for learning phrase bigrams*, Proceedings of the International Conference on Spoken Language Processing, Yokohama, pp. 843-846, 1994.

[26] Good, I.J; *The population frequencies of species and the estimation of population parameters*, Biometrika, vol. 40, pp. 237 - 264, 1953.

[27] Hull, J.J; *Incorporating lng syntax in visual text recognition with a statistical model*, IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 18, no. 12, pp. 1251-1256, December 1996.

[28] Iyer, R; Ostendorf, M; Rohlicek, R; *Language modelling with sentence-level mixtures*, Proceedings ARPA Human Language Technology Workshop, Princeton, pp. 82-96, 1994.

[29] Iyer, R; Ostendorf, M; *Modelling long distance dependence in language: topic mixtures vs. dynamic cache models*, Proceedings of the International Conference on Spoken Language Processing, Philadelphia, vol. 1, pp. 236-239, 1996.

[30] Jardino, M., Adda, G; *Automatic word classification using simulated annealing language modelling*, Proceedings of the International Conference on Acoustics, Speech and Signal Processing, vol. 2, pp. 1191-1194, 1993.

[31] Jardino, M., Adda, G; *Language modelling for CSR of large corpus using automatic classification of words*, Proceedings of Eurospeech, Berlin, pp. 1191-1194, 1993.

[32] Jardino, M; *A class bigram model for very large corpus*, Proceedings of the International Conference on Spoken Language Processing, Yokohama, vol.2, pp. 867-870, 1994.

[33] Jardino, M; *Multilingual stochastic n-gram class language models*, Proceedings of the International Conference on Acoustics, Speech and Signal Processing, Atlanta, vol 1, pp. 161-163, 1996.

[34] Jelinek, F; Mercer, R.L; Bahl, L.R; *A maximum likelihood approach to continuous speech recognition*, IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 5, pp. 179 - 190, March 1983.

[35] Jelinek, F. Mercer, R.L; *Probability distribution estimation from sparse data*, IBM Technical Disclosure Bulletin, 1984.

[36] Jelinek, F; *The development of an experimental discrete dictation recogniser*, Proceedings of the IEEE, vol. 73, no. 11, pp. 1616 - 1624, November 1985.

[37] Jelinek, F. *Up from trigrams: the struggle for improved language models*, Proceedings of Eurospeech, Genoa, vol. 3, pp. 1037-1040, 1991.

[38] Jelinek, F; Mercer, R.L; Roukos, S; *Principles of lexical language modelling for speech recognition*, in “Advances in speech signal processing”, Furui, S. and Sondhi, M.M. (eds.) , Marcel Dekker Inc., 1991.

[39] Jelinek, F; Merialdo, B; Roukos, S; Strauss, M; *A dynamic language model for speech-recognition*, Proceedings of the DARPA Speech And Language Workshop, pp. 293-295, February 1991.

[40] Johansson, S; Atwell, R; Garside, R; Leech, G; *The Tagged LOB corpus user’s manual*; Norwegian Computing Centre for the Humanities, Bergen, Norway 1986.

[41] Katz, S. *Estimation of probabilities from sparse data for the language model component of a speech recogniser*; IEEE Transactions on Acoustics, Speech and Signal Processing, vol. 35, no. 3, pp. 400-401, March 1987.

[42] Kneser, R; Ney, H; *Improved clustering techniques for class-based statistical language modelling*, Proceedings of Eurospeech, Berlin, vol. 2, pp. 973-976, 1993.

[43] Kneser, R; Personal communication, 1994.

[44] Kneser, R; Ney, H; *Improved backing-off for m-gram language modelling*, Proceedings of the International Conference on Acoustics, Speech and Signal Processing, Detroit, pp. 181-184, 1995.

[45] Kuhn, R; de Mori, R. ; *A cache-based natural language model for speech recognition*, IEEE Transactions On Pattern Analysis And Machine Intelligence, vol. 12, no. 6, pp. 570-583, June 1990. Corrected IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 14, pp. 691-692, 1992.

[46] Kukich, K; *Techniques for automatically correcting words in text*, ACM Computing Surveys, vol. 24, no. 4, December 1992.

[47] Lau, R. Rosenfeld, R; Roukos, S; *Trigger-based language models a maximum entropy approach*, Proceedings of the International Conference on Acoustics, Speech and Signal Processing, vol. 2, pp. 45-48, 1993.

[48] Martin, S; Liermann, J; Ney, H; *Algorithms for bigram and trigram clustering*, Proceedings of Eurospeech, Madrid, pp. 1253-1256, 1995.

[49] Miller, J.W; Alleva, F; *Evaluation of a language model using clustered model backoff*, Proceedings of the International Conference on Spoken Language Processing, Philadelphia, pp. 390-393, 1996.

[50] Mitton, R; *A description of a computer-usable dictionary file based on the Oxford Advanced Learner's Dictionary of Current English*, Department of Computer Science, Birkbeck College, University of London, June 1992.

[51] Nádas, A; *Estimation of probabilities in the language model of the IBM speech recognition system*, IEEE Transactions on Acoustics, Speech and Signal Processing, vol. 32, no. 4, pp. 859 - 861, August 1984.

[52] Nádas, A; *On Turing's formula for word probabilities*, IEEE Transactions on Acoustics, Speech and Signal Processing, vol. 33, no. 6, pp. 1414-1416, December 1985.

[53] Ney, H; Essen, U; Kneser, R; *On structuring probabilistic dependencies in stochastic language modelling*, Computer Speech and Language, vol. 8, pp. 1-38, 1994.

[54] Ney, H; Personal communication, 1994.

[55] Ney, H; Essen, U; Kneser, R; *On the estimation of "small" probabilities by leaving-one-out*, IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 17, no. 12, December 1995.

[56] Niesler, T.R; Woodland, P.C. *Variable-length category-based n-grams for language modelling*, Technical report CUED/F-INFENG/TR.215, Dept. Engineering, University of Cambridge, U.K., April 1995.

[57] Niesler, T.R; Woodland, P.C; *A variable-length category-based n-gram language model*, Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing, Atlanta, vol. 1, pp. 164-7, April 1996.

[58] Niesler, T.R; Woodland, P.C; *Word-to-category backoff language models*, Technical report CUED/F-INFENG/TR.258, Department of Engineering, University of Cambridge, U.K., May 1996.

[59] Niesler, T.R; Woodland, P.C; *Comparative evaluation of word- and category-based language models*, Technical report CUED/F-INFENG/TR.265, Department of Engineering, University of Cambridge, U.K., July 1996.

[60] Niesler, T.R; Woodland, P.C; *Combination of word-based and category-based language models*, Proceedings of the Fourth International Conference on Spoken Language Processing, Philadelphia, vol. 1, pp. 220-3, October 1996.

[61] Niesler, T.R; Woodland, P.C. *Word-pair relations for category-based language models*, Technical report CUED/F-INFENG/TR.281, Department of Engineering, University of Cambridge, U.K., February 1997.

[62] Niesler, T.R; Woodland, P.C; *Modelling word-pair relations in a category-based language model*, Proceedings of the International Conference on Acoustics, Speech and Signal Processing, Munich, vol. 2, pp. 795-798, April 1997.

[63] NIST (National Institute of Standards and Technology) *The NIST Speech Recognition Scoring Package (SCORE)*, version 3.6.2, available by anonymous ftp from `jaguar.ncsl.nist.gov/pub/score3.6.2.tar.Z`.

[64] O'Boyle, P; Owens, M; Smith, F.J; *A weighted average n-gram model of natural language*, Computer Speech And Language, vol. 8, pp. 337-349, 1994.

[65] O'Boyle, P; Ming, J; McMahon, J; Smith, F.J; *Improving n-gram models by incorporating enhanced distributions*, Proceedings of the IEEE International Conference on Acoustics, Speech and Signal Processing, Atlanta, vol. 1, pp. 168-171, April 1996.

[66] Odell, J.J; *The use of context in large vocabulary speech recognition*, Ph.D. Thesis, Department of Engineering, University of Cambridge, 1995.

[67] Odell, J.J; Niesler, T.R; *Lattice and language modelling toolkit v2.0*, Reference manual, Entropic Cambridge Research Laboratories Inc., 1996.

[68] Paul, D.B; Baker, J.M; *The design for the Wall Street Journal-based CSR corpus*, Proceedings of the International Conference on Spoken Language Processing, pp. 899-902, 1992.

[69] Placeway, P; Schwartz, R; Fung, P; Nguyen, L; *The estimation of powerful language models from small and large corpora*, Proceedings of the International Conference on Acoustics, Speech and Signal Processing, vol. 1, pg. 33-36, 1993.

[70] Pye, D; Woodland, P.C; *Large vocabulary speech recognition using cache-based language model adaptation*, Technical report CUED/F-INFENG/TR.285, Department of Engineering, University of Cambridge, U.K., January 1997.

[71] Rabiner, L.R; *A tutorial on hidden Markov models and selected applications in speech recognition*, Proceedings of the IEEE, vol. 77, pp. 257-285, February 1989.

[72] Rabiner, L.R, Juang, B.H; *Fundamentals of Speech recognition*, Prentice-Hall, 1993.

[73] Rao, P.S; Monkowski, M.D; Roukos, S; *Language model adaptation via minimum discrimination information*, Proceedings of the International Conference on Acoustics, Speech and Signal Processing, Detroit, pp. 161-164, 1995.

[74] Reza, F.M; *An introduction to information theory*, McGraw-Hill, 1961.

[75] Ries, K; Buo, F.D; Waibel, A; *Class phrase models for language modelling*, Proceedings of the International Conference on Spoken Language Processing, Philadelphia, pp. 398-401, 1996.

[76] Riseman, E.M; Ehrich, R.W; *Contextual word recognition using binary diagrams*, IEEE Transactions on Computers, vol. 20, no. 4, pp. 397-403, 1971.

[77] Robinson, A; Hochberg, M; Renals, S; *IPA: Improved phone modelling with recurrent neural networks*, Proceedings of the International Conference on Acoustics, Speech and Signal Processing, Adelaide, vol. 1, pp. 37-40, 1994.

[78] Rosenfeld, R; Huang, X; *Improvements in stochastic language modelling*, Proceedings of the DARPA Speech and Natural Language Workshop, New York, pp. 107-111, 1994.

[79] Rosenfeld, R; *Adaptive statistical language modelling: a maximum entropy approach"*, Ph.D. Dissertation CMU-CS-94-138, School of Computer Science, Carnegie Mellon University, April 1994.

[80] Rosenfeld, R; *A hybrid approach to adaptive statistical language modelling*, Proceedings ARPA Human Language Technology Workshop, Princeton, pp. 76-81, 1994.

[81] Rosenfeld, R; *The CMU statistical language modelling toolkit, and its use in the 1994 ARPA CSR evaluation*, ARPA Spoken Language Technology Workshop, Austin Texas, January 1995.

[82] Schwartz, R; Chow, Y-L; *The N-best algorithm: an efficient and exact procedure for finding the N most likely sentence hypotheses*, Proceedings of the International Conference on Acoustics, Speech and Signal Processing, pp. 81-84, Albuquerque, April 1990.

[83] Senior, A.W; *Off-line cursive handwriting recognition using recurrent neural networks*, Ph.D. Thesis, Department of Engineering, University of Cambridge, 1994.

[84] Shanmugan, K.S; Breipohl, A.M; *Random Signals: Detection, Estimation and Data Analysis*, Wiley, 1988.

[85] Shannon, C.E. *Communication theory : exposition of fundamentals*, IRE Transactions on Information Theory, no. 1, Feb. 1950.

[86] Smadja, F; McKeown, K; *Translating collocations for use in bilingual lexicons*, Proceedings of the Human Language Technology Workshop, pp. 152-156, March 1994.

[87] Starner, T; Makhoul, J; Schwartz, R; Chou, G; *On-line cursive handwriting recognition using speech recognition methods*, Proceedings of the International Conference on Acoustics, Speech and Signal Processing, Adelaide, vol. 5, pp. 125-128, 1994.

[88] Suhm, B; Waibel, A; *Towards better language models for spontaneous speech*, Proceedings of the International Conference on Spoken Language Processing, Yokohama, pp. 831-834, 1994.

[89] Waegner, N. P; *Stochastic models for language acquisition*, Ph.D. Thesis, University of Cambridge, Department of Engineering, 1993.

[90] Witschel, P; *Constructing linguistic oriented language models for large vocabulary speech recognition*, Proceedings of Eurospeech, Berlin, vol. 2, pp. 1199-1202, 1993.

[91] Woodland, P.C; Leggetter, C.J; Odell, J.J; Valtchev, V; Young, S.J; *The 1994 HTK large vocabulary speech-recognition system*, Proceedings of the International Conference on Acoustics, Speech and Signal Processing, Atlanta, pp. 73-76, 1995.

[92] Woodland, P.C; Gales, M.J.F; Pye,D; Valtchev, V; *The HTK Large Vocabulary Recognition System for the 1995 ARPA H3 Task*, Proceedings of the ARPA Speech Recognition Workshop, Harriman House, New York, 1996.

[93] Wright, J.H; Jones, G.J.F; Lloyd-Thomas, H; *A consolidated language model for speech-recognition*, Proceedings of Eurospeech, Berlin, vol. 2, pp. 977-980, 1993.

[94] Young, S.J; Jansen, J; Odell, J; Ollason, D; Woodland, P.C; *The HTK book (for HTK version 2.0)*, Cambridge University, March 1996.

[95] Zaho, R , Kenny, P; Labute, P; O’Shaughnessy, D; *Issues in large-scale statistical language modelling*, Proceedings of Eurospeech, Berlin, vol. 2, pp. 965-968, 1993.

Appendix A

Major text corpora

Statistical language models derive their parameters from a large body of example text, referred to as the **training corpus** or alternatively the **training set**. The Table below briefly describes some of the corpora most frequently encountered in the literature.

Corpus name	**Size (in words)**	**Details**
AP	AP (1987) : 15 million AP (1988) : 36 million	General news reportage.
ATIS	150 thousand	Air travel enquiries.
WSJ	87-89: 38 million 90-92: 35 million	Press reportage from the Wall Street Journal.
NAB1	227 million	Superset of WSJ and AP, including press reportage from the San José Mercury (general business news) and further material.
LOB	1.1 million	15 different text categories, including press reportage, fiction and scientific writing. Tagged with part-of-speech classifications
Verbmobil	278 thousand	Appointment negotiations, in German.
Switchboard	1.9 million	Spontaneous telephone conversations concerning 70 different predetermined topics.

Appendix B

Leaving-one-out cross-validation

Use of the training set probability as a performance criterion when testing the quality of a statistical model may lead to overfitting and consequent poor generalisation. Techniques based on cross-validation normally remedy this by employing a heldout set, but this reduces the amount of material available for training purposes. Leaving-one-out cross validation is an approximate way of modelling a heldout set while nevertheless making maximum use of the training data.

Consider a training set Ω^{tot}, containing N_ω members, that is divided into two subsets, Ω^{RT} (termed the **retained** part) and Ω^{HO} (termed the **heldout** part). Cross-validation approaches select models for Ω^{RT} so as to optimise the model performance on Ω^{HO}. In the case of probabilistic models, this may be achieved by maximising the probability of Ω^{HO}.

The leaving-one-out method [19] is a special case of this procedure in which Ω^{HO} is chosen to contain exactly one member of Ω^{tot}, while Ω^{RT} consists of the remaining $N_\omega - 1$. Let Ω^{tot} comprise the events $\{x(0)\,, x(1)\,, \ldots, x(N_\omega - 1)\}$ where each $x(i)$ is drawn from a finite alphabet $\mathbf{A}_x = (x_0, x_1, \ldots, x_{N_A-1})$ and where N_A is the alphabet size. Denoting the single member in Ω^{HO} by $x(i)$, the log probability of the heldout part may be written as :

$$LL(\Omega^{\text{HO}} \,|\, \Omega^{\text{RT}}) \;=\; \log\Big[P\Big(x(i) \,|\, \Omega^{\text{RT}}\Big)\Big]$$

where the probability estimate $P(\cdot)$ is made exclusively on the grounds of the data in the retained part Ω^{RT}. Leaving-one-out cross-validation involves the consideration of all N_ω possible ways in which Ω^{tot} may be partitioned into Ω^{HO} and Ω^{RT}. Denote the N_ω partitions formed by assigning $x(i)$ to Ω^{HO} by Ω_i^{HO} and Ω_i^{RT} respectively, where $i = \{0, 1, \ldots, N_\omega - 1\}$. The cumulative log probability over all these partitions is:

$$LL_{\text{cum}}(\Omega^{\text{tot}}) \;=\; \sum_{i=0}^{N_\omega-1} \log\Big[P\Big(x(i) \,|\, \Omega_i^{\text{RT}}\Big)\Big] \tag{127}$$

Now denote the number of occurrences of $x(i)$ in Ω^{tot} by $N^{\text{tot}}(x_i)$, and rewrite (127) as:

$$LL_{\text{cum}}(\Omega^{\text{tot}}) \;=\; \sum_{i=0}^{N_v-1} N^{\text{tot}}(x_i) \cdot \log\Big[P\Big(x(i) \,|\, \Omega_i^{\text{RT}}\Big)\Big] \tag{128}$$

Additional assumptions regarding the form of $P(x(i) \,|\, \Omega_i^{\text{RT}})$ may allow (128) to be simplified further.

In making use of all possible subdivisions into retained and heldout parts, the leaving-one-out approach makes optimal use of the available training data, an important consideration in situations where the data are sparse, as is indeed often the case for language modelling problems. Its drawback is the increased computation implied by the exhaustive partitioning operation, although efficient results may sometimes be obtained by simplifications possible for specific forms of $P(x(i) \,|\, \Omega_i^{\mathrm{RT}})$.

Appendix C

Dealing with unknown words

For open-vocabulary tasks, the test set generally contains words not present in the training corpus. In order to process such **out-of-vocabulary** (or simply **OOV**) words, we will add a dedicated entry labelled “UW” to the language model lexicon. As this entry refers not to a particular but to any unknown word, it is in fact a word category with an undetermined number of members. The UW entry will allow the language model both to predict the occurrence of OOV words, as well as to employ them as part of the context upon which further probability estimates can be based. This appendix describes the incorporation of the UW entry into a category-based language model.

Since by definition OOV words are not encountered during training[46], we will employ the leaving-one-out cross validation framework [19]. In particular, we will estimate the probability of an unknown word $P(\texttt{UW} \mid v_j)$ for each category v_j in the lexicon. Let N_c be the total number of words in the training corpus. Consider now the N_c possible ways in which we may split the training corpus into the two partitions $\mathbf{W}_i^{\text{RT}}$ and w_i^{HO} , where the first contains $N_c - 1$ members and the second exactly one, and where $i = \{0, 1, \ldots, N_c - 1\}$. Denote the set of categories to which the word w_i belongs in the training corpus by $V(w_i)$, and define:

$$\delta\left(w_i^{\text{HO}}, v_j\right) = \begin{cases} 1 \ \text{ if } \ (w_i^{\text{HO}} \notin \mathbf{W}_i^{\text{RT}}) \cap \Big(v_j \in V(w_i^{\text{HO}})\Big) \\ \\ 0 \ \text{ otherwise} \end{cases}$$

The probability $P\left(\texttt{UW} \,|\, v_j\right)$ of encountering an unknown event in category v_j within the sub-corpus $\mathbf{W}_i^{\text{RT}}$ of size $N_c - 1$ may then be estimated by the relative frequency :

$$P\left(\texttt{UW} \,|\, v_j\right) \;=\; \mathcal{E}\big\{\delta(w_i^{\text{HO}}, v_j)\big\} \;=\; \frac{\sum_{i=0}^{N_c-1} \delta(w_i^{\text{HO}}, v_j)}{N(v_j)} \qquad (129)$$

where $N(v_j)$ is the total number of events that have been seen in category v_j . Note that the numerator is simply the number of events which occur in this category and which also occur only once in the entire corpus. The number of unknown events $N_{uw}(v_j)$ that may be expected to be seen in category v_j in a

[46] Here we assume that the vocabulary contains all distinct words in the training corpus.

sub-corpus of size $N_c - 1$ may be estimated by again using the relative frequencies:

$$P\left(\mathtt{UW}|\,v_j\right) = \frac{N_{uw}(v_j)}{N(v_j) + N_{uw}(v_j)}$$

$$\Rightarrow N_{uw}\left(v_j\right) = \frac{P\left(\mathtt{UW}|\,v_j\right) \cdot N\left(v_j\right)}{1 - P\left(\mathtt{UW}|\,v_j\right)} \tag{130}$$

This equation may be used to estimate the count that should be assigned to the `UW` entry in every category v_j. Precautions must be taken, however, when the training data for certain categories are sparse, since it may then happen that the numerator of (129) approaches or even equals the denominator, leading to an extremely large estimate for $N_{uw}(v_j)$ according to (130). In order to avoid this, the probability estimate (129) has been altered heuristically as follows :

$$P\left(\mathtt{UW}|\,v_j\right) \overset{def}{=} \frac{\sum\limits_{i=0}^{N_c-1} \delta(w_i^{\mathrm{HO}}, v_j)}{N\left(v_j\right) + \eta} \qquad \text{where} \quad \eta > 0 \tag{131}$$

The constant η ensures that the denominator is always larger than the numerator, thus never permitting $P\left(\mathtt{UW}|\,v_j\right) = 1$. When $N\left(v_j\right)$ is small, indicating the category v_j to be sparsely trained, η will have a significant limiting effect on $P\left(\mathtt{UW}|\,v_j\right)$. However, as $N(v_j)$ increases, η becomes less significant and the estimate (131) approaches (129). Intuitively, the quantity η may be interpreted as an indication of the number of observations that should be made in a category v_j for the relative frequency estimates to be used with confidence. The effect η on the language model performance was seen to be weak, and a value between 5 and 10 was found to yield satisfactory results for the LOB corpus.

Appendix D

Experimental corpora and baseline language models

D.1. Introduction

The three different text corpora that will be used for practical evaluation of the proposed language models are described in the following sections. In addition, word-based bigram and trigram models employing the Katz back-off in conjunction with Good-Turing discounting [41] are constructed for each corpus, and will serve as benchmarks for experimental results.

D.2. The LOB corpus

The Lancaster-Oslo/Bergen (LOB) [40] consists of approximately one million words of British English text drawn from a wide spectrum of sources, including: press reportage, religious writing, popular lore, scientific matter, fiction, and humorous writings. Each word in the corpus is tagged with its part-of-speech (POS) classification, which describes its grammatical function within the sentence it appears in. A list of the classifications as used in this work is given in appendix E.

D.2.1. Preprocessing

In order to ensure compatibility with the other two corpora, the following preprocessing steps were applied to the LOB corpus:

- **Headings**: Headings were discarded from the corpus, since they are in general not grammatically correct sentences.
- **Numbers**: Numerics found in the corpus were converted to text, for instance: “510” was converted to “five hundred and ten”.
- **Punctuation**: All punctuation except sentence start and end markers was discarded and not used by any language model.
- **Contractions**: Contractions, which are tagged separately in the LOB corpus, are collapsed and tagged as single words, e.g.: “I” (PP1A) “’m” (BEM) becomes “I’m” (CPPBEM). This has been done because contractions have their own pronunciations, and therefore need to be treated as distinct words by the language model, and also because they appear as single words in the other two corpora. Appendix E describes how each individual contraction was tagged.

- **De-hyphenation**: Hyphenated words were treated separately, since this is the convention for the Switchboard and WSJ corpora. This also reduces the lexicon size and OOV rate, since a hyphenated pair might not appear in the vocabulary while the constituent words do. Some of the most common examples of this were numbers, like “two-hundred”.

  The removal of a hyphen leads to the introduction of an untagged word. This was remedied in the following way:

  1. When the untagged word is found to belong to only one category in the lexicon[47], it is assigned this tag.
  2. The remaining words were assigned the most likely category using a statistical tagger trained on the part of the corpus which was already intact.

D.2.2. Corpus statistics

Of the preprocessed corpus, 95% was allotted to the training corpus and 5% to the test set. This was done evenly across all text categories by assigning the first 95% of each of the 54 source files to the training set and the remainder to the test set, the division occurring at the closest sentence break. The following table summarises the important corpus statistics:

	Whole corpus	**Training set**	**Test set**
Total sentences	48,132	45,696	2,436
Total words	1,059,772	1,003,839	55,933
Vocabulary size	42,258	41,097	-
Lexicon size	51,936	50,462	-
OOV-rate	-	-	2.51%

Statistics for the LOB corpus.

D.2.3. Baseline word-based *n*-gram models

The following word-based bigram and trigram language models were constructed on the training corpus to serve as benchmarks in evaluations. The models employ the Katz back-off [41] in conjunction with Good-Turing discounting [26].

	Bigram	**Trigram**
Unique bigrams	388,614	388,614
Unique trigrams	-	753,843
Total n-grams	388,614	1,142,457
Perplexity	434.3	413.1

Baseline word-based language models for the LOB corpus.

[47] The **lexicon** refers to the collection of unique word-category pairs. Since words may belong to more than one category, the lexicon always has at least as many entries as the vocabulary.

D.3. The Switchboard (SWBD) corpus

This corpus consists of transcribed spontaneous telephone conversations concerning a predefined set of topics, as used at the 1995 Language Modelling Workshop (LM95) [11]. Part-of-speech information is not present in this corpus.

D.3.1. Preprocessing

Since a tagger based on the LOB corpus will be used to supply part-of-speech information, punctuation and spelling conventions must agree between the two. To achieve this, the following steps were taken:

- **Noises**: Noise markers such as "[LIPSMACK]", "[LAUGHTER]" and "[SIGH]" were removed.
- **Spelling**: Conversion of American to British spelling was effected to maintain compatibility with the LOB corpus, which consists of British English. For example, all occurrences of "color" were replaced with "colour". In total, 3,152 such changes were made automatically using software developed as part of this research.

D.3.2. Corpus statistics

The Switchboard dev-test text was used as a test set, and a closed vocabulary was employed. The following table summarises the important corpus statistics:

	Training set	**Test set**
Total words	1,860,178	10,179
Total sentences	225,437	1,192
Vocabulary size	22,643	-
Lexicon size	28,423	-
OOV-rate	-	Zero

Statistics for the Switchboard corpus.

D.3.3. Baseline word-based *n*-gram models

The following word-based bigram and trigram language models were constructed on the training corpus to serve as benchmarks in evaluations. The models employ the Katz back-off [41] in conjunction with Good-Turing discounting [26].

	Bigram	**Trigram**
Unique bigrams	305,793	305,793
Unique trigrams	-	878,087
Total n-grams	305,793	1,183,880
Perplexity	111.3	96.6

Baseline word-based language models for the Switchboard corpus.

D.4. The Wall Street Journal (WSJ) corpus

The WSJ corpus contains newspaper text collected from the Wall Street Journal over the period 1987-89 inclusive [68]. Since it is a considerably sized body of text, findings made using it are expected to hold also for larger corpora such as NAB1, on which state-of-the-art speech recognition systems have been based [91],[92].

D.4.1. Preprocessing

The verbalised pronunciation processed version of the WSJ corpus was used to build all language models. The filters `vp2svp1` and `sgml2text`[48] were employed to obtain plain text output, after which the following steps of preprocessing were carried out prior to model construction :

- **Trailing periods**: Remove all periods from the ends of words (e.g. “Mr.” ⇒ “Mr”), as this is the convention in both LOB and Switchboard corpora.
- **Typographical errors**: Correct common misspellings present in the WSJ corpus (e.g “million” is frequently misspelled as “milllion”). A total of 3,238 corrections of this type were made.
- **Spelling**: American to British spelling conversion was effected to maintain compatibility with the LOB corpus. A total of 134,109 such corrections were made automatically using software developed as part of this research.

D.4.2. Corpus statistics

Approximately the first 19,000 words of the standard WSJ setaside dev-test text for each of the years 1987-89 was employed as a test set and subjected to the same preprocessing steps as the training corpus. The vocabulary was that used in the 1994 DARPA evaluation [91]. The following table summarises the important corpus statistics:

	Training set	**Test set**
Total words	37,346,118	56,820
Total sentences	1,625,606	2,517
Different words	162,002	-
Vocabulary size	65,464	-
Lexicon size	110,487	-
OOV-rate	-	0.67%

Statistics for the WSJ corpus.

[48] These filters are part of the CMU language modelling toolkit [81].

D.4.3. Baseline word-based *n*-gram models

The following word-based bigram and trigram language models were constructed on the training corpus to serve as benchmarks in evaluations. The models employ the Katz back-off [41] in conjunction with Good-Turing discounting [26].

	Bigram	**Trigram**
Unique bigrams	4,051,165	4,051,165
Unique trigrams	-	9,548,230
Total n-grams	4,051,165	13,599,395
Perplexity	215.4	144.4

Baseline word-based language models for the WSJ corpus.

Appendix E

LOB corpus word-categories

E.1. Part-of-speech tags found in the LOB corpus

The following table lists the grammatical word classifications found in the LOB corpus [40]. Punctuation symbols have been omitted, and the tag "SE" referring to the end-of-sentence marker was added.

Category name	Description
&FW	foreign word
ABL	pre-quantifier (quite, rather, such)
ABN	pre-quantifier (all, half)
ABX	pre-quantifier / double conjunction (both)
AP	post-determiner (few, fewer, former, last, latter, least, less, little, many, more, most, much, next, only, other, own, same, several, very)
AP$	other's
APS	others
APS$	others'
AT	singular article (a, an, every)
ATI	singular or plural article (the, no)
BE	be
BED	were
BEDZ	was
BEG	being
BEM	am, 'm
BEN	been
BER	are, 're
BEZ	is, 's
CC	coordinating conjunction (and, and/or, but, nor, only, or, yet)
CD	cardinal (two, three, etc; hundred, thousand,etc; dozen, zero)
CD$	cardinal + genitive
CD-CD	hyphenated pair of cardinals
CD1	one, 1
CD1$	one's
CD1S	ones
CDS	cardinal + plural (tens, millions, dozens, etc)
CS	subordinating conjunction (after, although, etc)

Category name	Description
DO	do
DOD	did
DOZ	does, ’s
DT	singular determiner (another, each, that, this)
DT$	singular determiner + genitive (another’s)
DTI	singular or plural determiner (any, enough, some)
DTS	plural determiner (these, those)
DTX	determiner/double conjunction (either, neither)
EX	existential there
HV	have
HVD	had, ’d
HVG	having
HVN	had (past participle)
HVZ	has, ’s
IN	preposition (about, above, etc)
JJ	adjective
JJB	attributive-only adjective (chief, entire, main, etc)
JJR	comparative adjective
JJT	superlative adjective
JNP	adjective with word-initial capital (English, German, etc)
MD	modal auxiliary (’ll, can, could, etc)
NC	cited word
NN	singular common noun
NN$	singular common noun + genitive
NNP	singular common noun with word-initial capital (Englishman, German, etc)
NNP$	singular common noun with word-initial capital + genitive
NNPS	plural common noun with word-initial capital
NNPS$	plural common noun with word-initial capital + genitive
NNS	plural common noun
NNS$	plural common noun + genitive
NNU	abbreviated unit of measurement unmarked for number
NNUS	abbreviated plural unit of measurement
NP	singular proper noun
NP$	singular proper noun + genitive
NPL	singular locative noun with word-initial capital (Abbey, Bridge, etc)
NPL$	singular locative noun with word-initial capital + genitive
NPLS	plural locative noun with word-initial capital
NPLS$	plural locative noun with word-initial capital + genitive
NPS	plural proper noun
NPS$	plural proper noun + genitive
NPT	singular titular noun with word-initial capital (Archbishop, Captain, etc)
NPT$	singular titular noun with word-initial capital + genitive
NPTS	plural titular noun with word-initial capital
NPTS$	plural titular noun with word-initial capital + genitive
NR	singular adverbial noun (January, February, etc; Sunday, Monday, etc; east, west, etc; today, tomorrow, tonight, downtown, home)
NR$	singular adverbial noun + genitive
NRS	plural adverbial noun
NRS$	plural adverbial noun + genitive
OD	ordinal (1st, first, etc)
OD$	ordinal + genitive

Category name	Description
PN	nominal pronoun (anybody, anyone, anything; everybody, everyone, everything; nobody, none, nothing; somebody, someone, something; so)
PN$	nominal pronoun + genitive
PP$	possessive determiner (my, your, etc)
PP$$	possessive pronoun (mine, yours, etc)
PP1A	personal pronoun, 1st person singular nominative (I)
PP1AS	personal pronoun, 1st person plural nominative (we)
PP1O	personal pronoun, 1st person singular accusative (me)
PP1OS	personal pronoun, 1st person plural accusative (us, ’s)
PP2	personal pronoun, 2nd person (you, thou, thee, ye)
PP3	personal pronoun, 3rd person singular nominative + accusative (it)
PP3A	personal pronoun, 3rd person singular nominative (he,she)
PP3AS	personal pronoun, 3rd person plural nominative (they)
PP3O	personal pronoun, 3rd person singular accusative (him,her)
PP3OS	personal pronoun, 3rd person plural accusative (them, ’em)
PPL	singular reflexive pronoun
PPLS	plural reflexive pronoun, reciprocal pronoun
QL	qualifier (as, awfully, less, more, so, too, very, etc)
QLP	post-quantifier (enough, indeed)
RB	adverb
RB$	adverb + genitive (else’s)
RBR	comparative adverb
RBT	superlative adverb
RI	adverb (homograph of preposition: below, near, etc)
Rn	nominal adverb (here, now, thee, then, etc)
RP	adverbial particle (back, down, off, etc)
SE	end-of-sencence marker
TO	infinitival to
UH	interjection
VB	base form of verb
VBD	past tense of verb
VBG	present participle, gerund
VBN	past participle
VBZ	3rd person singular of verb
WDT	WH-determiner (what, whatever, whatsoever, interrogative which, whichever, whichsoever)
WDTR	WH-determiner relative (which)
WP	WH-pronoun, interrogative, nominative + accusative (who, whoever)
WP$	WH-pronoun, interrogative, gen (whose)
WP$R	WH-pronoun, relative, gen (whose)
WPA	WH-pronoun, nominative (whosoever)
WPO	WH-pronoun, interrogative, accusative (whom, whomsoever)
WPOR	WH-pronoun, relative, accusative (whom)
WPR	WH-pronoun, relative, nominative + accusative (that, relative who)
WRB	WH-adverb (how, when, etc)
XNOT	not, n’t
ZZ	letter of the alphabet (e, pi, x, etc)

E.2. Contactions and their part-of-speech tag assignments

Contraction(s)	POS tag	Contraction(s)	POS tag
he’ll, it’ll, I’ll, she’ll, that’ll	CPPMD	gonna	CTOGO
they’ll, we’ll	CPPSMD	gotta	CTOGT
what’ll	CWDTMD	I’d	CPPMD, CPPHVD
who’ll	CWPMD	she’d	CPPMD, CPPHVD
you’ll	CPPMD	he’d	CPPMD, CPPHVD
there’ll	CEXMD	it’d	CPPMD, CPPHVD
I’m	CPPBEM	who’d	CWPMD, CWPHVD
I’ve, you’ve	CPPHV	that’d	CPPMD, CPPHVD
we’ve	CPPSHV	we’d	CPPSMD, CPPSHVD
who’ve	CWPHV	you’d	CPPMD, CPPHVD
they’ve	CPPSHV	they’d	CPPSHVD, CPPSMD
they’re, we’re	CPPSBE	he’s	CPPBEZ, CPPHVZ
who’re	CWPBE	it’s	CPPBEZ, CPPHVZ
you’re	CPPBE	she’s	CPPBEZ, CPPHVZ
what’re	CWDTBE	that’s	CPPBEZ, CPPHVZ
let’s	CLET	what’s	CWDTBEZ, CWDTHVZ
here’s, there’s	CRNBEZ	who’s	CWPBEZ, CWPHVZ
wanna	CWNTO		
when’s, how’s, where’s	CWRBBEZ		
ain’t, isn’t	CNBEZ		
aren’t	CNBER		
wasn’t	CNBDZ		
weren’t	CNBED		
hadn’t	CNHVD		
hasn’t	CNHVZ		
cannot, can’t, couldn’t, daren’t	CNMD		
didn’t	CNDOD		
doesn’t	CNDOZ		
don’t	CNDO		
haven’t	CNHV		
mayn’t, mightn’t, mustn’t, needn’t, oughtn’t, shan’t, shaln’t, shouldn’t, won’t, wouldn’t	CNMD		

Notes:

- Tags for contractions have been chosen by considdering the part-of-speech assignments of the two constituent words.
- Contractions ending in “’d” may have two tags, depending on whether this is a contraction with “would” (MD) or with “had” (HVD).
- Contractions ending in “’s” may have two tags, depending on whether this is a contraction with “is” (BEZ) or with “has” (HVZ).

Appendix F

OALD tag mappings

An electronic version of the Oxford Advanced Learner's Dictionary (OALD) containing sufficiently detailed word tagging information was used to augment the lexicon extracted from the LOB corpus with the purpose of reducing the OOV rate [50]. The following table lists the OALD tags used in this process, as well as the LOB tags to which they were mapped. In a few cases it was necessary to map an OALD tag to more than one LOB tag, this being indicated by separating the latter with colons.

OALD tag	LOB tag
Gb	VBG
Gc	VBD
Gd	VBN
Ha	VBZ
Hb	VBG
Hc	VBD
Hd	VBN
H0	VB
H1	VB
H2	VB
H3	VB
H4	VB
H5	VB
Ia	VBZ
Ib	VBG
Ic	VBD
Id	VBN
I0	VB
I1	VB
I2	VB
I3	VB
I4	VB
I5	VB
Ja	VBZ
Jb	VBG
Jc	VBD
Jd	VBN
J0	VB
J1	VB
J2	VB
J3	VB
J4	VB
J5	VB

OALD tag	LOB tag
Ki	NN
Kj	NNS
K6	NN
K7	NN
K8	NN
K9	NN : NNS
Lk	NN
L@	NN
Mi	NN
Mj	NNS
M6	NN
M7	NN
M8	NN
M9	NN : NNS
M@	NN
Nl	NP
Nm	NP
Nn	NP
No	NP
OA	JJ
OB	JJ
OC	JJ
OD	JJ
OE	JJ
Op	JJ
Oq	JJB
Or	JJR
Os	JJT
Ot	JJ
Pu	RB
P+	RP
T-	IN
W-	UH

Appendix G

Trigger approximations

By virtue of the category-conditional probability function chosen to model occurrence correlations between trigger and target words, the distribution function is :

$$p(d) = \kappa \cdot \left(\prod_{i=0}^{d-1} \Big(1 - P_a - P_b - \gamma \cdot e^{-\rho \cdot i} \Big) \right) \cdot \Big(P_b + \gamma \cdot e^{-\rho \cdot d} \Big) \tag{132}$$

where

- d is the distance separating trigger and target.
- P_a is the probability of occurrence of the trigger.
- P_b, γ and ρ are the parameters describing the transient occurrence probability of the target with respect to a trigger sighting.
- κ is a normalising constant.

This appendix describes the two-stage algebraic approximation of (132) by a distribution of the form

$$\hat{p}(d) \;=\; \kappa \cdot \Big[\epsilon_1 \cdot (1 - P_1)^d \cdot P_1 + \epsilon_0 \cdot (1 - P_0)^d \cdot P_0 \Big]$$

G.1. First approximation

The objective here is to eliminate the product operator from equation (132), since its presence makes algebraic manipulation difficult. Consider the first term on the right-hand side of (132) :

$$\prod_{i=0}^{d-1} \Big(1 - P_a - P_b - \gamma \cdot e^{-\rho \cdot i} \Big) = (1 - P_a - P_b)^d \cdot \left(\prod_{i=0}^{d-1} \Big(1 - \zeta \cdot e^{-\rho \cdot i} \Big) \right)$$

where

$$\zeta = \frac{\gamma}{1 - P_a - P_b} \tag{133}$$

Take the logarithm and apply the first-order Taylor approximation $\ln(1+x) \approx x$ to find:

$$\begin{aligned}\ln\left(\prod_{i=0}^{d-1}\left(1-P_a-P_b-\gamma\cdot e^{-\rho\cdot i}\right)\right) &\approx d\cdot\ln(1-P_a-P_b)-\sum_{i=0}^{d-1}\zeta\cdot e^{-\rho\cdot i}\\ &= d\cdot\ln(1-P_a-P_b)-\frac{\zeta\left(1-e^{-\rho\cdot d}\right)}{1-e^{-\rho}}\end{aligned}$$

Now, take the inverse logarithm and resubstitute (133) into the above we obtain:

$$\prod_{i=0}^{d-1}\left(1-P_a-P_b-\gamma\cdot e^{-\rho\cdot i}\right) \approx (1-P_a-P_b)^d\cdot\left[e^{-\frac{\gamma\cdot\left(1-e^{-\rho\cdot d}\right)}{(1-P_a-P_b)\cdot(1-e^{-\rho})}}\right] \tag{134}$$

For clarity now define:

$$\Psi = e^{-\frac{\gamma}{(1-P_a-P_b)\cdot(1-e^{-\rho})}} \tag{135}$$

and it follows from (132), (134) and (135) that:

$$p(d)\approx\tilde{p}(d)=\kappa\cdot\left(P_b+\gamma\cdot e^{-\rho\cdot d}\right)\cdot(1-P_a-P_b)^d\cdot\Psi^{1-e^{-\rho\cdot d}} \tag{136}$$

This approximation is good when $\frac{\gamma}{1-P_a-P_b}\ll 1$, which is true when $P_a\ll 1$, $P_b\ll 1$ and $\gamma\ll 1$, as may be expected for content words.

G.2. Second approximation

Since we would ultimately like to find closed-form expressions for the approximate mean and mean-square of the probability distribution (132), and this is not yet possible using (136), we will further approximate the latter by:

$$\hat{p}(d)=\kappa\cdot[\epsilon_1\cdot(1-P_1)^d\cdot P_1+\epsilon_0\cdot(1-P_0)^d\cdot P_0] \tag{137}$$

where

$$\kappa\,(\epsilon_1+\epsilon_0)=1 \tag{138}$$

The functional form of (137) has the following motivations:

- As the superposition of two geometric terms, it retains the overall geometric character exhibited empirically by the distribution $p(d)$.
- The faster geometric component should model the initially more rapid decay of the observed distribution (which is in turn due to the higher conditional probability at small d).
- The slower geometric component should model the tail of the observed distribution.
- Closed form expressions for the mean and mean-square exist.

Note firstly that

$$\sum_{d=0}^{\infty} \hat{p}(d) \; = \; 1$$

and that for $0 \leq P_0 \leq 1$ and $0 \leq P_1 \leq 1$:

$$0 \leq \hat{p}(d) \leq 1 \qquad \forall \;\; d \in (0, 1, 2, \ldots, \infty)$$

so that equation (137) represents a valid probability mass function. In order to solve for the parameters of (137) in terms of the parameters of (136), we impose the following three constraints :

1. Equality in the limit as $d \rightarrow \infty$: From (136) we find that:

   $$\lim_{d\rightarrow\infty} \tilde{p}(d) \; = \; \kappa \cdot P_b \cdot (1 - P_a - P_b)^d \cdot \Psi$$

   and from (137), assuming $P_0 < P_1$:

   $$\lim_{d\rightarrow\infty} \hat{p}(d) = \kappa \cdot \epsilon_0 \cdot (1 - P_0)^d \cdot P_0$$

   and so by requiring

   $$\lim_{d\rightarrow\infty} \tilde{p}(d) \; = \; \lim_{d\rightarrow\infty} \hat{p}(d)$$

   we may choose

   $$P_0 = P_a + P_b \tag{139}$$

   and

   $$\epsilon_0 = \frac{P_b \cdot \Psi}{P_a + P_b} \tag{140}$$

2. Equality at $d = 0$: From (136) we find that:

   $$\tilde{p}(0) \; = \; \kappa \cdot (P_b + \gamma)$$

   and from (137) :

   $$\hat{p}(0) \; = \; \kappa[\epsilon_1 \cdot P_1 + \epsilon_0 \cdot P_0]$$

   so that, for $\tilde{p}(0) = \hat{p}(0)$ we find

   $$P_1 = \frac{P_b + \gamma - \epsilon_0 \cdot P_0}{\epsilon_1} \tag{141}$$

3. Equality of the first derivative at $d = 0$: From (136) we find that:

$$\frac{\partial}{\partial d}\tilde{p}(d) \;=\; \kappa\Big[-\rho\cdot\gamma\cdot e^{-\rho d} + \ln\left(1-P_a-P_b\right)\cdot\Big(P_b+\gamma\cdot e^{-\rho d}\Big) + \ln\left(\Psi\right)\cdot\rho\cdot e^{-\rho d}\cdot\Big(P_b+\gamma\cdot e^{-\rho d}\Big)\Big]\cdot\left(1-P_a-P_b\right)^d\cdot\Psi^{1-e^{-\rho d}}$$

from which, taking $d = 0$, we obtain:

$$\left.\frac{\partial}{\partial d}\tilde{p}(d)\right|_{d=0} \;=\; \kappa\cdot\Big[-\rho\cdot\gamma + (P_b+\gamma)\cdot\ln(1-P_a-P_b) + \rho\cdot(P_b+\gamma)\cdot\ln(\Psi)\Big] \tag{142}$$

Similarly, from (137):

$$\frac{\partial}{\partial d}\hat{p}(d) \;=\; \kappa\cdot\Big[\epsilon_1\cdot\ln\left(1-P_1\right)\cdot(1-P_1)^d\cdot P_1 \;+\; \epsilon_0\cdot\ln\left(1-P_0\right)\cdot(1-P_0)^d\cdot P_0\Big]$$

from which, taking $d = 0$, we obtain:

$$\left.\frac{\partial}{\partial d}\hat{p}(d)\right|_{d=0} \;=\; \kappa\cdot\Big[\epsilon_1\cdot\ln(1-P_1)\cdot P_1 + \epsilon_0\cdot\ln(1-P_0)\cdot P_0\Big] \tag{143}$$

Using (139) and (140) we may write:

$$\begin{aligned}\epsilon_0\cdot\ln(1-P_0)\cdot P_0 \;&=\; \frac{P_b\cdot\Psi}{P_a+P_b}\cdot\ln(1-P_0)\cdot(P_a+P_b)\\ &=\; P_b\cdot\Psi\cdot\ln(1-P_0)\end{aligned} \tag{144}$$

Now, by requiring:

$$\left.\frac{\partial}{\partial d}\tilde{p}(d)\right|_{d=0} \;=\; \left.\frac{\partial}{\partial d}\hat{p}(d)\right|_{d=0}$$

we find from (139), (141), (142), (143) and (144) that:

$$\epsilon_1\cdot\ln(1-P_1)\cdot P_1 \;=\; \ln(1-P_0)\cdot\Big[P_b+\gamma-P_b\cdot\Psi\Big] + \rho\cdot\Big[\,(P_b+\gamma)\cdot\ln(\Psi)-\gamma\Big]$$

$$\Rightarrow (P_b+\gamma-P_b\cdot\Psi)\cdot\ln(1-P_1) \;=\; \ln(1-P_0)\cdot\Big[P_b+\gamma-P_b\cdot\Psi\Big] + \rho\cdot\Big[\,(P_b+\gamma)\cdot\ln(\Psi)-\gamma\Big]$$

$$\begin{aligned}\Rightarrow \ln(1-P_1) \;&=\; \frac{\ln(1-P_0)\cdot\Big[P_b+\gamma-P_b\cdot\Psi\Big]+\rho\cdot\Big[\,(P_b+\gamma)\cdot\ln(\Psi)-\gamma\Big]}{P_b+\gamma-P_b\cdot\Psi}\\ &=\; \ln(1-P_0)+\frac{\rho\cdot\Big[\,(P_b+\gamma)\cdot\ln(\Psi)\Big]}{P_b+\gamma-P_b\cdot\Psi}\end{aligned}$$

so that, finally, we obtain:

$$P_1 \;=\; 1-(1-P_0)\cdot e^{\left[\frac{\rho\cdot\left[\left(P_b+\gamma\right)\cdot\ln(\Psi)-\gamma\right]}{P_b+\gamma-P_b\cdot\Psi}\right]} \tag{145}$$

Appendix H

The truncated geometric distribution

Consider an experiment consisting of Bernoulli trials with probability of success P_x. The number of repetitions before witnessing the first positive result is described by the geometric distribution:

$$P_{gtc}(d) = (1 - P_x)^d \cdot P_x \tag{146}$$

Now select the subset of trials for which $d < N$, where N is an integer greater than zero. The probability distribution over this interval $P(d)$ may be determined by applying the normalisation requirement

$$\sum_{d=0}^{N-1} P(d) \;=\; 1$$

to equation (146) and obtaining:

$$P(d) \;=\; \frac{P_{gtc}(d)}{\sum\limits_{d=0}^{N-1} P_{gtc}(d)} \;=\; \frac{(1-P_x)^d \cdot P_x}{1-(1-P_x)^N} \qquad \text{with} \qquad d \in \{0, 1, \ldots, N-1\} \tag{147}$$

In this appendix we find expressions for the mean and mean-square of this **truncated geometric distribution**, and begin by calculating the moment generating function:

$$\begin{aligned} M_d(t) \;&=\; \sum_{d=0}^{N-1} e^{t \cdot d} \cdot P(d) \\ &=\; \frac{P_x}{1-(1-P_x)^N} \cdot \sum_{d=0}^{N-1} (1 - P_x)^d \cdot e^{t \cdot d} \\ &=\; \frac{P_x \cdot \left(1 - \left[e^t \cdot (1-P_x)\right]^N\right)}{\left(1-(1-P_x)^N\right) \cdot \left(1 - e^t \cdot (1-P_x)\right)} \\ &=\; \Upsilon \cdot \frac{1 - \left(e^t \cdot (1-P_x)\right)^N}{1 - e^t \cdot (1-P_x)} \end{aligned} \tag{148}$$

where:

$$\Upsilon \;=\; \frac{P_x}{\Big(1-(1-P_x)^N\Big)}$$

Taking the derivative of (148) with respect to t we find:

$$\begin{aligned}
\frac{\partial}{\partial t}M_d(t) \;&=\; \Upsilon \cdot \frac{e^t \cdot (1-P_x)\cdot\Big(1 - e^{Nt}\cdot(1-P_x)^N\Big) - N\cdot e^{Nt}\cdot(1-P_x)^N\cdot\Big(1-e^t\cdot(1-P_x)\Big)}{\Big(1-e^t\cdot(1-P_x)\Big)^2} \\
&=\; \Upsilon \cdot \frac{e^t\cdot(1-P_x) - N\cdot e^{Nt}\cdot(1-P_x)^N + (N-1)\cdot e^{(N+1)t}\cdot(1-P_x)^{N+1}}{\Big(1-e^t\cdot(1-P_x)\Big)^2}
\end{aligned}$$

and to obtain the mean $\mu(P_x, N)$ we set $t = 0$:

$$\begin{aligned}
\mu(P_x, N) \;&=\; \frac{\partial}{\partial t}M_d(t)\bigg|_{t=0} \\
&=\; \Upsilon \cdot \frac{(1-P_x)\cdot\Big[1 + (1-P_x)^{N-1}\Big[(N-1)\cdot(1-P_x) - N\Big]\Big]}{P_x^2}
\end{aligned} \tag{149}$$

Taking the second derivative of (148) with respect to t we find:

$$\begin{aligned}
\frac{\partial^2}{\partial t^2}M_d(t) \;&=\; \Upsilon\cdot\frac{\Big(1-e^t\,(1-P_x)\Big)^2\cdot\Big[e^t\,(1-P_x) - N^2 e^{Nt}\,(1-P_x)^N + (N-1)\,(N+1)\,e^{(N+1)t}\,(1-P_x)^{N+1}\Big]}{\Big(1-e^t\,(1-P_x)\Big)^4} \\
&+\; 2\cdot\Upsilon\cdot\frac{\Big[e^t\,(1-P_x) - N\cdot e^{Nt}\,(1-P_x)^N + (N-1)\cdot e^{(N+1)t}\cdot(1-P_x)^{N+1}\Big]\cdot\Big(1-e^t\,(1-P_0)\Big)\cdot e^t\,(1-P_x)}{\Big(1-e^t\,(1-P_x)\Big)^4}
\end{aligned}$$

and obtain the mean-square $\nu(P_x, N)$ of the distribution by again setting $t = 0$:

$$\begin{aligned}
\nu(P_x, N) \;&=\; \frac{\partial^2}{\partial^2 t}M_d(t)\bigg|_{t=0} \\
&=\; \Upsilon\cdot\frac{P_x^2\cdot\Big[(1-P_x)\cdot\Big(1 - N^2\cdot(1-P_x)^{N-1} + (N-1)\cdot(N+1)\cdot(1-P_x)^N\Big)\Big]}{P_x^4} \\
&+\; 2\cdot\Upsilon\cdot\frac{\Big[(1-P_x)\cdot\Big(1-N\cdot(1-P_x)^{N-1} + (N-1)\cdot(1-P_x)^N\Big)\Big]\cdot P_x\cdot(1-P_x)}{P_x^4}
\end{aligned} \tag{150}$$