Statistical Language Modelling

Yoshihiko Gotoh and Steve Renals

University of Sheffield, Department of Computer Science

1 Introduction

Grammar-based natural language processing has reached a level where it can ‘understand’ language to a limited degree in restricted domains. For example, it is possible to parse textual material very accurately and assign semantic relations to parts of sentences. An alternative approach originates from the work of Shannon over half a century ago [41], [42]. This approach assigns probabilities to linguistic events, where mathematical models are used to represent statistical knowledge. Once models are built, we decide which event is more likely than the others according to their probabilities. Although statistical methods currently use a very impoverished representation of speech and language (typically finite state), it is possible to train the underlying models from large amounts of data. Importantly, such statistical approaches often produce useful results. Statistical approaches seem especially well-suited to spoken language which is often spontaneous or conversational and not readily amenable to standard grammar-based approaches.

This chapter concerns statistical language modelling. In a speech recognition system the role of the language model is to assign probabilities to word sequences. Recently, similar models to speech recognition language models have been employed to perform higher level tasks, such as structuring and extracting information from spoken language. In this chapter, we first outline the basic framework of *n*-gram language models (section 2), which form the core of current statistical approaches. A crucial technical consideration here is how to estimate *n*-gram statistics from sparse training data. We go on to describe two approaches—based on *n*-gram models—to encapsulate varying contents and styles: section 3 is concerned with mixture language models and section 4 builds on the observation that the occurrence rate of a word is not uniform, but varies between documents. Finally we describe a statistical finite state model for the extraction of information, such as proper names and dates from spoken language.

2 *n*-gram Language Modelling

2.1 The Basics of *n*-gram Modelling

The standard formulation of a statistical speech recognition system may be written as:

$$p(w \mid x) \propto \underbrace{p(x \mid w)}_{\substack{\text{acoustic}\\ \text{model}}} \cdot \underbrace{p(w)}_{\substack{\text{language}\\ \text{model}}} \quad . \tag{1}$$

The generation of the acoustic data $x = \{x_1, x_2, \ldots, x_t\}$ from a word sequence $w = \{w_1, w_2, \ldots, w_m\}$ is described by the acoustic model, $p(x \mid w)$. This often takes the form of a hidden Markov model (HMM). The language model, $p(w)$, is a prior probability distribution over word sequences, and is typically an n-gram model (discussed below). Since $p(w)$ does not depend on acoustics, it is usual to estimate the language model from textual data. Although this can introduce some distortion to the model, the amount of reliable speech transcription is generally not sufficient for statistical estimation.

Building *n*-grams. First we consider the following sentence:

later the prime minister tony blair telephoned mr. yeltsin

taken from the *THISL data collection*[1]. An n-gram is simply a sequence of successive n words, *e.g.*,

unigram	yeltsin
bigram	mr. yeltsin
trigram	telephoned mr. yeltsin
four-gram	blair telephoned mr. yeltsin

An n-gram model is statistical because it builds on the 'counts' (*i.e.*, the number of occurrences) of such events. Indeed, counting may be thought of as the simplest form of statistical learning. Shown below are the most frequently occurring words in the collection:

the	394 481 occurrences
to	240 001
a	225 506
in	177 997
and	133 962
is	109 217
be	84 020
that	69 265

Given that there were 7 488 445 words in the collection, we can make some simple statistical guesses: for example the word 'the' appears once in every 20 words on average.

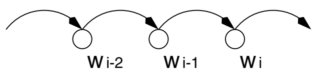


Fig. 1. A finite state machine builds an n-gram language model.

[1] The *THISL data collection* [36] consists of a large amount of programme scripts, audio data, and some human generated reference transcriptions from a variety of TV and radio news and current affairs programmes broadcast by the BBC since 1997.

More formally, we can specify a finite state machine that generates a sequence of words using a probabilistic model[2] (figure 1). For computational reasons, we apply the Markov assumption; that is to say the model has a limited horizon, and is time invariant. The former implies that the current word does not depend on the entire history, but at most on the last few words. The latter asserts that an n-gram model is roughly stationary. Assuming a dependency on the two previous words results in a trigram model:

$$p_{\text{trigram}}(w) = p(w_1)\ p(w_2 \mid w_1)\ p(w_3 \mid w_1, w_2)\ \ldots\ p(w_m \mid w_{m-2}, w_{m-1}) \ . \tag{2}$$

Although crude in appearance, it has proven difficult to develop more sophisticated language models that consistently outperform trigrams in large vocabulary speech recognition tasks [21].

In practice, the text is *normalized* before counting n-gram occurrences. Text normalization includes the removal of most punctuation and case information, the verbalization of numbers (*e.g.*, $12.8bn becomes 'twelve point eight billion dollars'), the setting of relevant markers such as sentence breaks, and the correction of spelling errors. The vocabulary size may be restricted to a certain number (say, 65 536 words). In this case, out-of-vocabulary (OOV) words may be mapped to a single unknown word symbol (<unk>) when counting n-grams.

The value of n is an important question for n-gram modelling. A small value of n leads to more reliable parameter estimation. Larger values of n lead to a more detailed context. The total number of potential n-grams scales exponentially with n, so most higher order n-grams do not occur in the training data[3]. Consequently, bigram or trigram models are most widely used for n-gram modelling of large vocabularies.

Sparseness of Training Data. Maximum likelihood (ML) estimation maximizes the probability of the model generating the training data. For example, the ML estimate for a bigram (v, w)—word v, followed by word w—is given by the conditional form:

$$p(w \mid v) = \frac{c(v, w)}{c(v)} \ , \tag{3}$$

where $c(v, w)$ and $c(v)$ imply unigram and bigram frequencies observed in the training data[4].

ML estimation of n-gram language models can be seriously affected by training data sparsity. For example, the *THISL data collection* contains about 7.5 million words;

[2] This is equivalent to saying that we play a Shannon game ('what is the next word?') using a probabilistic model [42].

[3] Suppose the vocabulary size is 65 536 words, the number of parameters for bigram, trigram, and four-gram models are up to 4.3×10^9, 2.8×10^{14}, and 1.8×10^{19}, respectively. On the other hand, a typical corpus size is on the order of 10^5 to 10^9.

[4] The ML estimate is also given by the joint probability form, $p(v, w) = \frac{c(v, w)}{N}$, where N denotes the total number of training instances. However, it is more natural to use the conditional form because most calculations for language modelling are carried out by exploiting conditional relations. Thus for the rest of this chapter, we show conditional forms only.

if we estimate a trigram language models using a 65 536 word vocabulary, the number of possible trigrams is about 37 million times greater than the size of the training data. Hence simple ML estimation will result in many 'zero probabilities'; any word sequence, whose n-gram component was not present in the training data, will not be processed properly using the ML formulation (2).

In order to address this problem, a variety of smoothing techniques have been developed. They are designed to 'smooth' the probability estimates for n-gram models so that any n-gram component is given a non-zero probability. Smoothing techniques may be divided into two main classes: discounting schemes for re-distributing frequencies to unseen events; and approaches to combine different level models (*e.g.*, interpolation, back-off).

2.2 Discounting Techniques

ML estimation of n-gram language models results in over-estimates of the probabilities of those n-grams that are observed in the training data. The probabilities of unobserved n-grams are under-estimated (set to zero). This is sometimes called the *zero probability problem* . Discounting schemes, such as Good-Turing and absolute discounting, address this problem by reducing, or *discounting*, the ML probability estimates, and re-distributing the 'freed' probability mass to previously unseen events (figure 2).

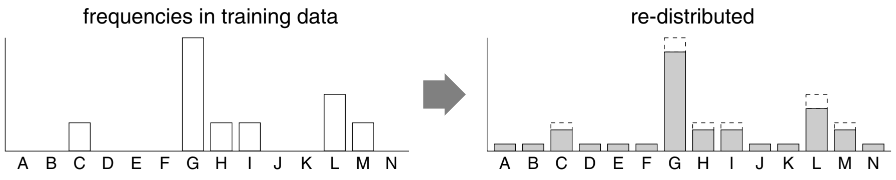


Fig. 2. 'Discounted mass' is re-distributed to events not observed in the training data.

Table 1 shows discounted bigram frequency estimates, calculated from the *THISL data collection.* The details of each approach are given below.

Empirical Estimation. An empirical approach (sometimes referred to as held-out estimation) may be based on the question 'how often do bigrams that appear r times in the training data tend to occur in new data?', using a held-out data set to empirically validate an estimated model. Let u_r denote the number of n-grams that occur exactly r times in the training data, and $c_t(\cdot)$ and $c_h(\cdot)$ represent frequencies in the training and the held-out data, respectively. A discounted frequency for a bigram (v,w) is calculated as follows [8], [28]:

$$\hat{r}_{emp} = \frac{1}{u_r} \cdot \sum_{\substack{(v,w): \\ c_t(v,w)=r}} c_h(v,w) \quad . \tag{4}$$

Table 1. This table summarizes discounted bigram frequency estimates for the *THISL data collection*. u_r denotes the number of bigrams that occurred r times in the training data. $\hat{r}_{emp}$ is an empirical estimate derived from the test data for bigrams that occurred r times in the training data. $\hat{r}_{ml}(= r)$ is a simple ML estimate. $\hat{r}_{cv}$, $\hat{r}_{gt}$, and $\hat{r}_{abs}$ are the cross validation, the Good-Turing, and the absolute discounting estimates. They were obtained from the training data only, without looking at the test data, but they are closer to $\hat{r}_{emp}$ than the ML estimate.

r	u_r	$\hat{r}_{emp}$	$\hat{r}_{ml}$	$\hat{r}_{cv}$	$\hat{r}_{gt}$	$\hat{r}_{abs}$
0	–	0.0019	0	0.0015	0.0021	0.0023
1	839 300	0.47	1	0.45	0.53	0.34
2	220 490	1.22	2	1.25	1.23	1.34
3	90 077	2.21	3	2.23	2.26	2.34
4	50 826	3.24	4	3.21	3.23	3.34
5	32 882	4.10	5	4.20	4.18	4.34
6	22 928	5.14	6	5.15	5.21	5.34
7	17 053	6.10	7	6.13	6.23	6.34
8	13 290	7.23	8	7.11	7.24	7.34
9	10 693	8.13	9	8.18	8.19	8.34

If the amounts of training and held-out data are different, they should be normalized. The conditional relative bigram frequency is:

$$\hat{f}_{emp}(w \mid v) = \frac{\hat{r}_{emp}}{c(v)} .$$

Table 1 contains a column showing the empirical estimation of discounted bigram frequencies for the *THISL data collection*, using program scripts (7 488 445 words, 1 394 406 bigrams) as the training data and reference transcriptions (487 027 words, 182 441 bigrams) as the held-out data.

Cross Validation. Cross validation is a related approach to the held-out estimation. It may be carried out by the following steps [28]:

1. Separate the entire training data into $K > 1$ sections.
2. For each $k = 1 \ldots K$, hold out section k and
   (a) from the remaining $K - 1$ sections (referred to as 'development data'), collect statistics $u_r(k)$: the number of n-grams that occur exactly r times;
   (b) from the held-out section k, collect the number of total occurrences $t_r(k)$ of bigrams that appear exactly r times in the development data; normalize $t_r(k)$ according to the development data and the held-out data sizes.
3. Calculate the average:

$$\hat{r}_{cv} = \frac{\sum_{k=1 \ldots K} t_r(k)}{\sum_{k=1 \ldots K} u_r(k)} . \tag{5}$$

Table 1 also shows estimates by cross validation, where program scripts were split into two parts (one with 3 755 764 words, 906 100 bigrams, and the other with 3 732 681 words, 903 811 bigrams).

Good-Turing Discounting. The Good-Turing estimate of discounted frequencies does not rely on a held-out data set. Originally attributed to Turing, it may be derived as a special case of cross validation in which a single training instance is held out at each time of iteration [8], [14], [22], [30]. The Good-Turing discounted frequency $\hat{r}_{gt}$ (for $r > 0$) has the form:

$$\hat{r}_{gt} = (r+1)\frac{u_{r+1}}{u_r} \quad , \tag{6}$$

where u_r is the number of n-grams that occur exactly r times in the training data.

Suppose we denote the number of the training instances by $N = \sum_r r \cdot u_r$, then the total discounted mass from cases for $r > 0$ is given by

$$N - \sum_r \hat{r} \cdot u_r = N - \sum_r (r+1) \cdot u_{r+1} = u_1 \quad .$$

which is then redistributed to all unobserved events. For a vocabulary of size V, the number of n-grams not observed in the training data is $u_0 = |V|^n - \sum_r u_r$. In this case the Good-Turing estimate for $r = 0$ is $\hat{r}_{gt} = \frac{u_1}{u_0}$.

For the *THISL data collection* with 19 959 vocabulary words, the number of zero frequency bigrams is $u_0 = 19\,959^2 - 1\,393\,940 = 396\,967\,741$, among which the discounted mass was thinly distributed. The zero frequency estimate is in fact not zero but $\hat{r}_{gt} = 0.0021$ as indicated in table 1.

If we define the discounting factor as $d_r = \frac{\hat{r}_{gt}}{r}$, then we can write the conditional relative frequency for discounted bigrams as:

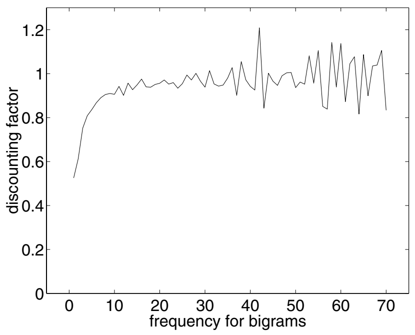
$$\hat{f}_{gt}(w \mid v) = d_{c(v,w)} \cdot f_{ml}(w \mid v) \quad , \tag{7}$$

for $c(v,w) = r > 0$, using the ML estimate $f_{ml}(w \mid v) = \frac{c(v,w)}{c(v)}$.

Absolute Discounting. Absolute discounting [14], [30] is an alternative scheme that does not require a held out set, in which a constant e is subtracted from each non-zero count, and redistributed over unseen events:

$$\hat{r}_{abs} = \begin{cases} r - e & \text{if } r > 0 \\ \frac{e}{u_0} \times \sum_r u_r & \text{if } r = 0 \end{cases} \quad , \tag{8}$$

where r and u_r are defined as before. The discounting constant e may be estimated from the held-out data. Alternatively, Ney *et al.* [30] suggested $e \sim \frac{u_1}{u_1 + 2u_2}$; using



r	d_r	d_r^*
0	-	-
1	0.53	0.46
2	0.61	0.56
3	0.75	0.72
4	0.81	0.78
5	0.84	0.81
6	0.87	0.85
7	0.89	0.87
8	0.91	1.00
9	0.91	1.00

Fig. 3. The Good-Turing estimate may not be very accurate for large r. The graph on the left indicates it is no longer reliable for the *THISL data collection*. Instead Katz [23] suggested an alternative approach that discounts for $1 \le r \le k$ only, using $d_r^* = \frac{\frac{\hat{r}_{gt}}{r} - \xi_k}{1 - \xi_k}$ where $\xi_k = \frac{(k+1)u_{k+1}}{u_1}$, and $d_r^* = 1$ for $r > k$. For example, suppose we wish to discount d_r^* between $1 \le r \le 7$, then $\xi_7 = \frac{8 \cdot u_8}{u_1} = 0.127$. The table on the right shows discounted frequencies by the Good-Turing (6) and by Katz's formulation.

this approximation, we have calculated $e = 0.656$, and thus the zero frequency estimate $\hat{r}_{abs} = 0.0023$, for the *THISL data collection* (table 1).

The conditional relative frequency for discounted bigrams is given by

$$\hat{f}_{abs}(w \mid v) = \frac{\hat{r}_{abs}}{c(v)} = f_{ml}(w \mid v) - \frac{e}{c(v)} \quad , \tag{9}$$

for $c(v,w) = r > 0$, using the ML estimate $f_{ml}(w \mid v)$.

2.3 Smoothing with Lower Level *n*-grams

In the previous section we discussed discounting techniques that freed some probability mass to take account of unseen events. We now consider two approaches that enable a redistribution of the discounted mass by combining different levels of *n*-gram models (unigram, bigram, trigram, ...) in such a way that the most precise model available is used. We consider *interpolation* and *back-off* smoothing: each approach recursively redistributes discounted mass from each level of an *n*-gram model to lower level *n*-gram models.

Interpolation. The interpolation method is based on a linear combination of *n*-gram models. For example, the probability for a trigram (u,v,w)—word 'u', followed by word 'v', then 'w'—may be smoothly estimated as:

$$p_{int}(w \mid u,v) = \lambda_3 p(w \mid u,v) + \lambda_2 p(w \mid v) + \lambda_1 p(w) + \lambda_0 \cdot \mathcal{A} \quad , \tag{10}$$

with some constant $\mathcal{A}$ and constraints $\sum_j \lambda_j = 1$ for $0 \le \lambda_j \le 1$ [20]. Generally, λ can be a function of the history (*i.e.*, '$\ldots,u,v$') that satisfies the total probability constraints, $\sum p_{int} = 1$.

Combining different levels of *n*-gram models is beneficial because the discounted probability estimate of a lower level *n*-gram (observed in the training data) is more reliable than probability estimates of unseen higher level *n*-grams. The following recursive formulation is based on that by Federico *et al.* [14].

Consider the estimation of a conditional trigram probability $p(w \mid u,v)$. This may be calculated using a discounted relative frequency $\hat{f}(w \mid u,v)$ and a bigram probability $p(w \mid v)$:

$$\begin{aligned} p(w \mid u,v) &= \hat{f}(w \mid u,v) + \{1 - \alpha(u,v)\} \cdot p(w \mid v) \\ \text{where} \quad \alpha(u,v) &= \sum_{w \in E(u,v,w)} \hat{f}(w \mid u,v) \enspace . \end{aligned} \tag{11}$$

$\alpha(u,v)$ is a non-zero estimate of the marginal probability that a trigram with context (u,v) exists in the model, where $E(u,v,w)$ implies the trigram entry in the model.

The bigram and unigram probabilities are estimated in a similar fashion:

$$\begin{aligned} p(w \mid v) &= \hat{f}(w \mid v) + \{1 - \alpha(v)\} \cdot p(w) \\ \text{where} \quad \alpha(v) &= \sum_{w \in E(v,w)} \hat{f}(w \mid v) \enspace ; \end{aligned} \tag{12}$$

$$\begin{aligned} p(w) &= \hat{f}(w) + \{1 - \alpha\} \cdot A \\ \text{where} \quad \alpha &= \sum_{w} \hat{f}(w) \enspace . \end{aligned} \tag{13}$$

One possible choice of the constant may be $A = \dfrac{1}{|W|}$ with sufficiently large $|W|$ (say, $|V| \ll |W|$). Finally, to verify the total probability constraints for the conditional trigram probability, suppose that $\sum_{w \in E(v,w)} p(w \mid v) = 1$,

$$\sum_{w \in E(u,v,w)} p(w \mid u,v) = \sum_{w \in E(u,v,w)} \hat{f}(w \mid u,v) + \{1 - \alpha(u,v)\} \cdot \sum_{w \in E(v,w)} p(w \mid v) = 1 \enspace .$$

Similar calculations may be done for bigram and unigram probabilities as well.

Back-Off. Rather than combining different level *n*-gram models, the back-off method chooses the most appropriate *n*-gram level to use when estimating conditional probabilities. The back-off method partitions the mass between *n*-grams which are backed off from higher level models. An analogous recursive formulation to that used for interpolation-based smoothing may be employed.

The conditional trigram probability $p(w \mid u,v)$ is estimated using a discounted relative frequency $\hat{f}(w \mid u,v)$ if the trigram is observed in the training data, otherwise the model *backs off* to a bigram probability estimate $p(w \mid v)$:

$$p(w \mid u,v) = \begin{cases} \hat{f}(w \mid u,v) & \text{if } E(u,v,w) \text{ exists} \\ \beta(u,v) \cdot p(w \mid v) & \text{otherwise} \end{cases}$$

$$\text{where} \quad \beta(u,v) = \frac{1 - \alpha(u,v)}{1 - \sum_{w \in E(u,v,w)} \hat{f}(w \mid v)} \quad . \tag{14}$$

$\beta(u,v)$ is the back-off factor, $\alpha(u,v)$ is the non-zero marginal probability estimate, and $E(u,v,w)$ implies the trigram entry in the model.

The bigram and unigram estimates again take a similar form:

$$p(w \mid v) = \begin{cases} \hat{f}(w \mid v) & \text{if } E(v,w) \text{ exists} \\ \beta(v) \cdot p(w) & \text{otherwise} \end{cases}$$

$$\text{where} \quad \beta(v) = \frac{1 - \alpha(v)}{1 - \sum_{w \in E(v,w)} \hat{f}(w)} \quad ; \tag{15}$$

$$p(w) = \begin{cases} \hat{f}(w) & \text{if } E(w) \text{ exists} \\ \beta \cdot A & \text{otherwise} \end{cases}$$

$$\text{where} \quad \beta = \frac{1 - \alpha}{1 - \sum_w A} \quad . \tag{16}$$

We note that $\sum_w A = |V| \cdot A$, where $A = \frac{1}{|W|}$ is again a possible choice. To verify the total probability constraints, suppose $E(u,v,w)$ does not exist,

$$\sum_{E(u,v,w)=\phi} p(w \mid v) = 1 - \sum_{w \in E(u,v,w)} p(w \mid v) = 1 - \sum_{w \in E(u,v,w)} \hat{f}(w \mid v) \quad .$$

Thus,

$$\sum_w p(w \mid u,v) = \sum_{w \in E(u,v,w)} \hat{f}(w \mid u,v) + \beta(u,v) \cdot \sum_{E(u,v,w)=\phi} p(w \mid v) = 1 \quad .$$

Evaluating Language Models. In order to evaluate language models, an average log probability is often used. Suppose the test data contain N words:

$$LP = \frac{1}{N} \sum_{i=1 \ldots N} \log_2 p(w_i \mid w_{i-n+1}, \ldots, w_{i-1}) \quad ,$$

and the *perplexity* is the average branching factor defined by $PP = 2^{-LP}$. The best language model will average the fewest guesses over the text data. For the example sentence at the beginning of this chapter, table 2 shows individual trigram probabilities using the Good-Turing discounted and backed off model. From the table, we can immediately calculate the average log probability, $LP = -7.55$, and the perplexity, $PP = 187.6$.

Table 2. This table shows probabilities for individual trigram components from the test sentence ‘later the prime minister tony blair telephoned mr. yeltsin’, using Good-Turing discounting and back-off modelling, derived from the *THISL data collection*. All numbers are in log domain (base 2). For example, this table indicates that a trigram entry for ‘the prime minister’ exists in the model, thus the probability estimate is simply the discounted relative frequency for the trigram. However, the trigram entry for ‘tony blair telephoned’ is not found in the model, so we need to back-off to a bigram ‘blair telephoned’, which again does not exist and we further back-off to a unigram ‘telephoned’. The trigram probability estimate for ‘tony blair telephoned’ is thus a product (a sum in the log domain) of back-off factors and the discounted relative frequency for the unigram ‘telephoned’.

(u,v,w)	$p(w \mid u,v)$	$\hat{f}(w \mid u,v)$	$\beta(u,v)$	$\hat{f}(w \mid v)$	$\beta(v)$	$\hat{f}(w)$
later	-10.86					-10.86
later the	-4.08			-4.08		
later the prime	-8.27	-8.27				
the prime minister	-0.31	-0.31				
prime minister tony	-6.10	-6.10				
minister tony blair	-1.29	-1.29				
tony blair telephoned	-20.88	$\rightarrow$	-0.22	$\rightarrow$	-2.15	-18.51
blair telephoned mr.	-10.03	$\rightarrow$	0	$\rightarrow$	-0.80	-9.23
telephoned mr. yeltsin	-6.15	$\rightarrow$	0	-6.15		

3 Adaptive Language Modelling: Topic Coherence

The n-gram model is syntactic and locally constrained, based on a Markov chain of a word sequence whose parameters are derived from word frequency counts given a training corpus. Because the n-gram is a statistical model, a fundamental assumption is that the task domain is similar to that for the training corpus. Consequently, a relatively large amount of training data is required to accommodate the great number of variations that may occur in spoken language. The n-gram approach works well when these underlying assumptions of static task domain and sufficient training data hold, while it does not when the application domain varies from the training conditions.

To address this problem, several adaptive language modelling schemes have been proposed. Since the n-gram model has a constrained context (typically the previous two or three words) most adaptive language modelling schemes attempt to exploit longer distance dependencies. In this section we develop a mixture language modelling approach, in which the component models display a degree of topic coherence. We decompose the task into two problems: document classification (employing techniques first developed for text retrieval) and mixture modelling.

3.1 Document Classification

Topic dependent language models may be obtained by the automatic derivation of topic information from text, followed by the combination of global and topic dependent text statistics. A ‘bag-of-words’ model, which is based on a histogram of weighted unigram frequencies, is used to estimate the topic of a document[5]. This approach—frequently adopted in text retrieval—assumes that the similarity between documents can be measured from word (or term) co-occurrence statistics. The similarity operation may be regarded as being carried out in a high dimensional space whose dimension is given by the vocabulary size. One advantage of using such a measure is that local constraints that might have an adverse global effect, such as word order, may be discarded. We present a brief overview of some relevant text retrieval concepts; a more thorough discussion is available elsewhere (*e.g.*, [44], [45], or chapter 4 of this book).

Term Weighting. A focal point of document classification is the calculation of weights for words according to their importance in documents. For example, unigram frequencies of vocabulary items may be used. As the total word counts often vary in orders of magnitude between documents, estimates of unigram probabilities can be used instead in order to avoid possible effects of document size. It would also be beneficial to weight the more important words in order to avoid distortions occurring due to common non-content words.

Term weighting schemes combine global and local factors to produce weighting factors for the within-document unigram probabilities[6]. Suppose that g_i implies a global weight for a word w_i in a collection of documents and that l_{ij} is a local value within a document d_j. The global weight is designed to enhance words which are not widely distributed across many documents, whereas the local weight is usually related to term frequency within the document. There exist a number of approaches to evaluate g_i and l_{ij}, such as $tf \cdot idf$ [48], and Okapi term weighting [37]. In general, a term weight a_{ij} has the form:

$$a_{ij} = g_i \cdot l_{ij} \ , \tag{17}$$

and a document vector for d_j is defined as a collection of term weights $d_j = \{a_{ij}\}$ (which is usually very sparse). Documents can be classified according to their vector representations. A consistent distance measure is the angle between two document vectors.

Dimension Reduction. So far the distance between two documents has been defined in a $|V|$-dimensional space. One text retrieval approach, known as *latent semantic indexing* , estimates document similarity in a reduced dimension space by calculating the principal components (eigenvectors) of the $|V|$-dimensional document vectors [11]. It is based on the *singular value decomposition* of a very large, sparse, word by document matrix [3]. Let $A = \{d_j\}$ denote an $m \times n$ matrix whose rank is r; each column describes a document vector d_j, with the entries being some measure associated with vocabulary items in that document. A can be decomposed as

$$A = U\Sigma V^T \ , \tag{18}$$

[5] We use the term ‘document’ loosely; in spoken language applications it may refer to an entire story, paragraph, or even a fixed size window of, say, 500 words.

[6] More generally stopping and stemming algorithms may be used before term weighting schemes [15].

where V^T is the transpose of V. Σ is an $r \times r$ diagonal matrix whose non-zero elements correspond to the singular values, or the non-negative square roots of r eigenvalues for AA^T. U and V are $m \times r$ and $n \times r$ matrices whose rows may be referred to as word and document singular vectors. They define the orthonormal eigenvectors associated with the r eigenvalues of AA^T and A^TA, respectively.

The singular vectors corresponding to the s ($s \leq r$) largest singular values are then used to define an s-dimensional document space. Using these vectors, $m \times s$ and $n \times s$ matrices U_s and V_s can be redefined along with $s \times s$ singular value diagonal matrix Σ_s. It is known that $\hat{A}_s = U_s \Sigma_s V_s^T$ is the closest matrix (in a least square sense) of rank s to the original matrix A [3]. As a consequence, given an m-dimensional vector d that describes a document, it is warranted that an s-dimensional projection $\hat{d}_s$ computed by

$$\hat{d}_s = d^T U_s \Sigma_s^{-1} \quad , \tag{19}$$

lies in the closest s-dimensional document subspace with respect to the original m-dimensional space. The projection $\hat{d}_s$ represents principal components that capture the largest variation of words and documents without sacrificing much information. s is typically of the order of 100, many times smaller than the original document dimension.

3.2 Mixture Language Modelling

Once documents have been classified into topics, topic-dependent n-gram models may be derived. We now outline a scheme to combine these component language models into a *mixture language model.*

Formulation. A mixture model, denoted by $\mathcal{M}$, is built as the weighted sum of J components, $<\mathcal{M}_1, \ldots, \mathcal{M}_J>$, derived from a partitioned corpus; this is a similar approach to the dynamic cache models discussed in section 4. Let $f(w_t \mid w_1^{t-1}; \mathcal{M})$ and $f(w_t \mid w_1^{t-1}; \mathcal{M}_j)$ imply n-gram type parameters for a mixture and its j^{th} component, respectively. A mixture is defined as

$$p(w_t \mid w_1^{t-1}; \mathcal{M}) = \sum_{j=1}^{J} c_j p(w_t \mid w_1^{t-1}; \mathcal{M}_j) \quad . \tag{20}$$

Given this form and the constraints $\sum_{j=1}^{J} c_j = 1$, we wish to find mixing factors, c_j, that maximize the likelihood for a document (sequence of words). The expectation maximization (EM) algorithm [13] is an iterative procedure that is suitable for this purpose; we start from an appropriate initial guess $c_j^{[0]}$. If there are T words in the document, then the p^{th} estimate is given by

$$c_j^{[p]} = \frac{1}{T} \sum_{\tau=1}^{T} \frac{c_j^{[p-1]} p(w_\tau \mid w_1^{\tau-1}; \mathcal{M}_j)}{\sum_{k=1}^{J} c_k^{[p-1]} p(w_\tau \mid w_1^{\tau-1}; \mathcal{M}_k)} \quad . \tag{21}$$

The procedure is similar to other mixture density parameter estimation problems [35].

Formula (21) produces updated estimates of the mixing factors only after the entire document is processed. This is not very useful because a major objective is to flexibly adjust to the varying style of documents. Suppose that $t-1$ words $\{w_1, \dots, w_{t-1}\}$ have been processed so far; a new word w_t is given. Then the t^{th} incremental estimate is obtained recursively by

$$c_j^{[t]} = \frac{t-1}{t} c_j^{[t-1]} + \frac{1}{t} \gamma_j^{[t]} \quad \text{where} \quad \gamma_j^{[t]} = \frac{c_j^{[t-1]} p(w_t \mid w_1^{t-1}; M_j)}{\sum_{k=1}^{J} c_k^{[t-1]} p(w_t \mid w_1^{t-1}; M_k)} \,. \tag{22}$$

Using this form, the mixing factors are adjusted automatically to maximize the likelihood of the document, thus indirectly incorporating topic information.

3.3 Evaluation

There is a large body of work in the general area of document classification (see chapter 4 and [28]). The Text Retrieval Conference (TREC) has been a forum for the evaluation of text retrieval systems for a variety of tasks including routing, filtering and spoken document retrieval. Recent evaluations have consistently indicated that many of the best performing systems use Okapi term weighting [37] (or a closely related approach) and do not employ dimension reduction. Our own experiments in mixture language modelling have indicated that Okapi term-weighting performs best and that SVD dimension reduction causes a degradation in performance [16].

Table 3 indicates that the mixture model improves the perplexity over the conventional model, even though the trigram hit rate declines. A lower hit rate is unavoidable when a corpus is partitioned to smaller subsets: despite this handicap, the mixture model has shown improved perplexities compared with the conventional approach.

Table 3. Perplexities and trigram hit rates for a baseline model (a single trigram derived from the complete training data), and a mixture of 10 component models (derived using Okapi term weighting). The experiment was performed using the *British National Corpus* [5].

model	perplexity	trigram hit (%)
single model	180.0	62.4
mixture of 10 class models	164.2	42.8

4 Adaptive Language Modelling: Word Level Correlation

In both spoken and written language, word occurrences are not random but vary greatly from document to document. Indeed, modern text retrieval relies on the degree of departure from randomness as a discriminative indicator [44], [45]. In this section we discuss ways to mathematically realize the intuition that an occurrence of a certain word may increase the chance for the same word being observed later. Dynamic language models aim to incorporate longer distance correlations between words, either through an explicit probabilistic model or by blending statistics for recent words with a global model. An alternative approach uses a statistical model of word occurrence based on the use of the Poisson distribution [18], [38].

4.1 Dynamic Language Models

Although a constant word rate is an unlikely premise, it is nevertheless assumed in *n*-gram language modelling. The notion of adaptive modelling by a mixture of topic-dependent language models (section 3) is one proposed solution to this problem. Alternatively, Rosenfeld incorporated trigger pairs (longer distance word-level dependencies) into a model structure using maximum entropy [22], [40].

The *dynamic cache model* is a simpler approach, based on an observation that recently appearing words are more likely to re-appear than would be predicted by a static *n*-gram model [10], [25]. This model blends the global *n*-gram model with a local model:

$$\hat{p}(w_t|w_a^{t-1}) = p_{\text{global}}(w_t|w_a^{t-1}) + p_{\text{local}}(w_t|w_a^{t-1}) \quad . \tag{23}$$

$p_{\text{local}}(w_t|w_a^{t-1})$ is usually estimated using a *cache* of the last K words. In the simplest form, dynamic cache models blend a locally estimated unigram model with the globally estimated *n*-gram (typically trigram). Such models have been reported to lower perplexity by around 10% [10], [25], but to have minimal effects on the word error rate in large vocabulary speech recognition (*e.g.*, [47]).

4.2 Variable Word Rates

A variant of the dynamic cache model incorporates recency into the cache by using an exponential decaying weight on the contribution of words in the cache. However, rather than relying on such *ad hoc* devices to model variable word occurrences, it is possible to use an explicit probabilistic model of word rate, such as a Poisson mixture. Church and Gale [9] have demonstrated that a continuous mixture of Poisson distributions can produce accurate estimates of variable word rate. In related work Lowe [26] applied a beta-binomial mixture model to topic tracking and detection.

If we consider that occurrences of each word are the result of an underlying Poisson process, then the word rate is no longer uniform. Consider a set of documents[7] and a word w. We assume that each document produces w independently according to a Poisson process with a single parameter $\lambda > 0$:

$$\theta^{[poiss]}(x) = P(X = x;\, \lambda) = \frac{e^{-\lambda}\lambda^x}{x!} \quad , \tag{24}$$

where X is a discrete, non-negative random variable representing the number of occurrences of w, with expected value $E[X] = \lambda$ and variance $V[X] = \lambda$.

[7] Recall our loose definition of a document; basically it is a unit of spoken (or written) data of a certain length that contains some topic(s), or content(s).

A less constrained model of variable word rate is offered by a *mixture of Poissons*. Suppose the parameter λ of the pdf (24) is distributed according to some function $\phi(\lambda)$, then we define a continuous mixture of Poisson distributions by

$$\theta(x) = \int_0^\infty \theta^{[poiss]}(x)\phi(\lambda)d\lambda \ . \tag{25}$$

In particular, if $\phi(\lambda)$ is a gamma distribution, *i.e.*, $\phi(\lambda) = G(\lambda;\ \alpha,\beta) = \dfrac{\lambda^{\alpha-1}e^{-\frac{\lambda}{\beta}}}{\beta^\alpha\Gamma(\alpha)}$, for $\alpha > 0$ and $\beta > 0$, then the integral (25) is reduced to a discrete distribution for $x = 0, 1, \ldots$ such that

$$\theta^{[nb]}(x) = NB(X = x;\ \alpha,\beta) = \binom{\alpha+x-1}{x}\frac{\beta^x}{(1+\beta)^{\alpha+x}} \ . \tag{26}$$

This $\theta^{[nb]}(x)$ is a *negative binomial distribution*[8] and its expected value and variance are given by $E[X] = \alpha\beta$ and $V[X] = \alpha\beta(\beta+1)$, respectively.

The histograms in figure 4 show the number of word (unigram) and bigram occurrences in news broadcast. 'for' and 'you' appeared approximately the same number of times across all the transcripts. Using the constant word rate assumption, they would have been assigned a probability of around 0.0086. However their occurrence rates varied between documents; about 11% and 33% of all documents did not contain 'for' and 'you' (respectively), while 1% and 3% contained these words more than 30 times. This seems to indicate that occurrences of 'for' is less dependent on the content (or the style) of a document. A negative binomial distribution was used to model the variable word rate in each case.

The negative binomial seems to model word occurrence rate relatively well for most vocabulary items, regardless of frequency. Figure 4 illustrates this for one of the most frequent words 'of' (probability of 0.023 according to the constant word rate assumption) and the less frequently occurring 'church' (less than 0.00029). In particular, 'church' appeared only in 93 out of 2583 documents, but 28 of them contained more than 10 instances, suggesting strong correlation with the document content. We also collected statistics of bigrams. They are very sparse; for example, 'for you' and 'of church' appeared in 127 and 6 documents. The negative binomial model fits bigrams as well, indicating that variable bigram rate can also be modelled using a continuous mixture of Poissons.

4.3 Variable Word Rate Language Models

Taking word occurrence rate into account changes a probabilistic language model from a situation akin to playing a lottery, to something closer to betting on a horse race:

[8] Let $\phi(\lambda) = G(\lambda;\ \alpha,\beta)$. Integration (25) is straightforward using the definition of the gamma function, $\Gamma(\alpha) = \int_0^\infty t^{\alpha-1}e^{-t}dt$, and the recursion, $\Gamma(\alpha+1) = \alpha\Gamma(\alpha)$. The resultant pdf (26) has a slightly unconventional form in comparison to that in most of standard textbooks (*e.g.*, [12]), but is identical by setting a new parameter $\gamma = \dfrac{1}{1+\beta}$ with $0 < \gamma < 1$.

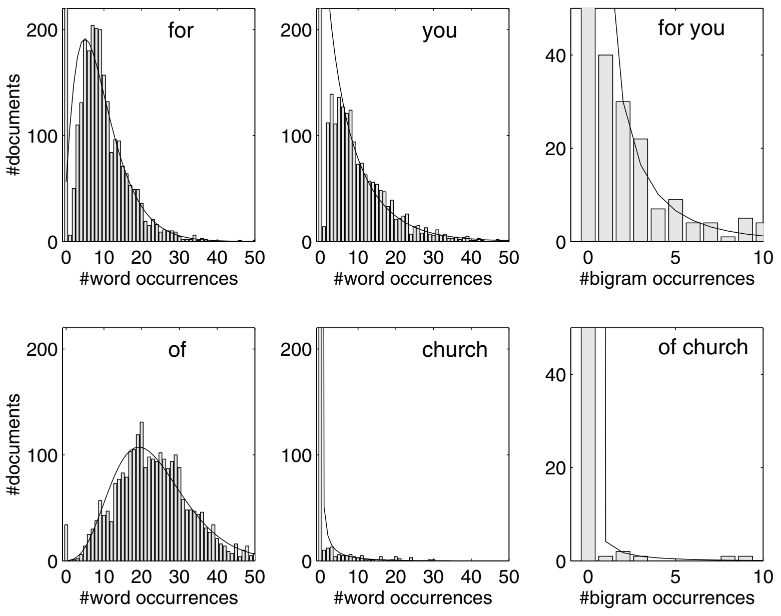


Fig. 4. Word and bigram occurrences vary between documents. Histograms show the number of occurrences taken from transcripts of the *Hub–4* Broadcast News acoustic training data (1996–97 — not free, but available from `http://www.ldc.upenn.edu/`). These transcripts were separated into documents according to section markers and those with less than 100 words were removed, resulting 2583 documents containing slightly less than 1.3 million words in total. The number of occurrences were then normalized to 1000-word length documents. A negative binomial distribution (solid line) was used to approximate each histogram.

the odds for a certain word improve if it has come up in the past. We eliminate the constant word rate assumption and present a variable word rate n-gram language model. Discounting and smoothing schemes are also considered.

Relative Frequencies with Prior Word Occurrences. An expected value for a random variable X, given a condition $X \geq a$ for a certain value a, is calculated by replacing the pdf $\theta(X = x)$ with the conditional pdf $\theta(X = x;\ X \geq a)$:

$$E[X;\ X \geq a] = \int_{-\infty}^{\infty} x\theta(X = x;\ X \geq a)dx = \frac{\int_a^{\infty} x\theta(x)dx}{\int_a^{\infty} \theta(x)dx} \tag{27}$$

(*e.g.*, see page 105 of [33]). Now in discrete space, let $c(w)$ imply the frequency of word w in the training data. We denote by $f(w;\ c(w) \geq r_w)$ a conditional relative frequency after observing r_w occurrences of word w, which is given by

$$f(w;\ c(w) \geq r_w) = \frac{1}{N} E[w;\ c(w) \geq r_w] = \frac{1}{N} \frac{\sum_{j=r_w}^{N} j \cdot \theta_w(j)}{\sum_{j=r_w}^{N} \theta_w(j)} \quad . \tag{28}$$

N is a document length (*e.g.*, for histograms in figure 4, N is normalized to 1000) and function (28) is defined for $r_w = 0, 1, \ldots, N$. $\theta_w(j)$ is the occurrence rate for word w in an N-length document (*e.g.*, Poisson, negative binomial), satisfying $\sum_{j=0}^{N} \theta_w(j) = 1$.

The conditional relative frequency formula satisfies our intuition as well; the value of (28) increases monotonically as the number of observation r_w accumulates (easy to verify), and it reaches '1' when $r_w = N$. To the other end,

$$f(w;\ c(w) \geq 0) = \frac{1}{N} \sum_{j=0}^{N} j \cdot \theta_w(j) \quad , \tag{29}$$

and this corresponds to the case with no prior information of word occurrence. For the conventional approach with the constant word rate assumption, this $f(w;\ c(w) \geq 0)$ is used regardless of any word occurrences.

Figure 5 illustrates how conditional expected values change after observing word occurrences. Figure 6 demonstrates conditional relative frequencies derived by function (28). The graph on the right indicates that the first few instances do not increase the relative frequency very much for frequent words (*e.g.*, 'of'), but have a substantial effect for the less common word (*e.g.*, 'church'). As the number of observations increases, the former is caught up by the latter.

Variable bigram rate relative frequencies can be calculated in a similar fashion. In the following we use short hand notations $f(r_w)$ and $f(r_{w|v})$ for indicating $f(w;\ c(w) \geq r_w)$ and $f(w \mid v;\ c(v,w) \geq r_{w|v})$, respectively. The same notations are used for $p(r_w)$ and $p(r_{w|v})$, *i.e.*, probabilities by variable rate models.

Discounting and Smoothing Techniques. Recall that, for any practical application, smoothing of the probability estimates is essential to avoid zero probabilities for events that were not observed in the training data. As before, let $E\,(v,w)$ denote a bigram entry in the model. A bigram probability $p(r_{w|v})$ can be smoothed with a unigram probability $p(r_w)$ using the interpolation [14]:

$$\begin{aligned} p(r_{w|v}) &= \hat{f}(r_{w|v}) + \{1 - \alpha(v)\} \cdot p(r_w) \\ \text{where} \quad \alpha(v) &= \sum_{w \in E\,(v,w)} \hat{f}(r_{w|v} \quad , \end{aligned} \tag{30}$$

where $\hat{f}(r_{w|v})$ implies a 'discounted' relative frequency of variable bigram rate (described later) and $\alpha(v)$ is a non-zero probability estimate as introduced in section 2.

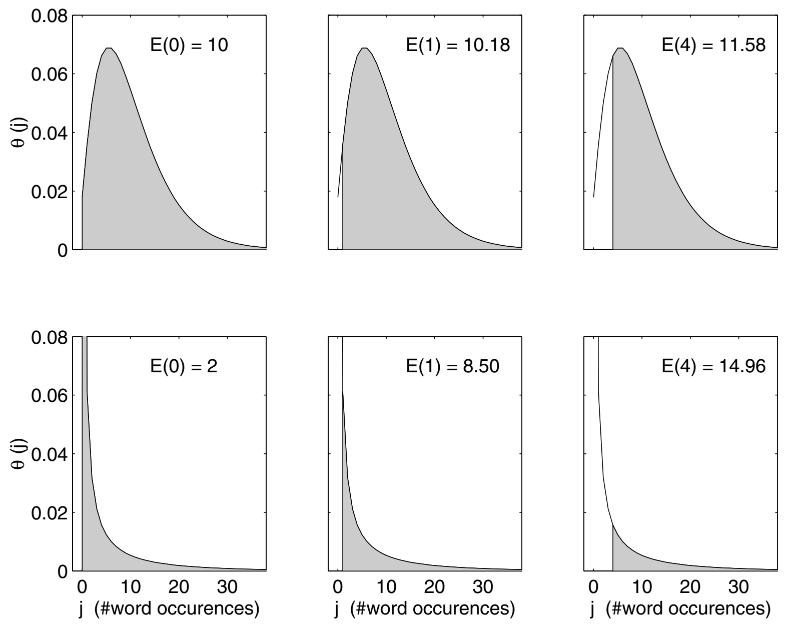


Fig. 5. These graphs illustrates the modelled occurrence rates typically observed for frequent function words (top three, *e.g.*, ‘of’) and for words that bear strong content information (bottom three, *e.g.*, ‘church’). Conditional expected values $E(r_w) \equiv E[w;\ c(w) \geq r_w]$ are calculated from the shaded areas and shown together. Two graphs in the most left indicate the cases with no prior knowledge of word occurrence (constant word rate assumption). Once a word is observed, the expected value changes very little for a function word, but increases dramatically for a content words (graphs in the middle). Graphs in the right show the cases when each word has occurred four times; now we expect to see this content word more often than some function words.

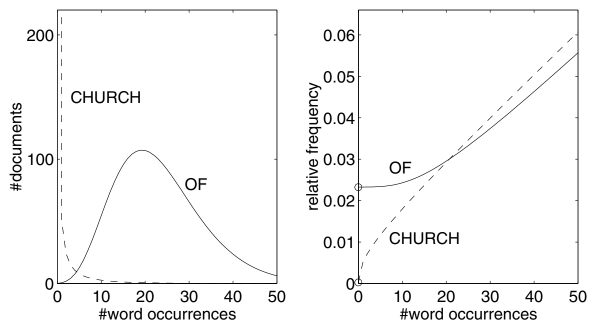


Fig. 6. The left graph shows word occurrence rates for a function word, ‘of’, and a content word, ‘church’, observed in documents of 1000-word length (normalized), modelled by negative binomial distributions (identical to those in figure 4). The right graph indicates conditional relative frequencies after a certain number of word occurrences. Circles (‘o’) correspond to relative frequencies under the constant word rate assumption (0.023 for ‘of’ and 0.00029 for ‘church’).

The alternative is to apply a back-off smoothing [23]. Let $\beta(v)$ denote a back-off factor, then

$$p(r_{w|v}) = \begin{cases} \hat{f}(r_{w|v}) & \text{if } E(v,w) \text{ exists} \\ \beta(v) \cdot p(r_w) & \text{otherwise} \end{cases}$$

$$\text{where} \quad \beta(v) = \frac{1 - \alpha(v)}{1 - \sum_{w \in E(v,w)} \hat{f}(r_w)} \ . \tag{31}$$

A variable rate unigram probability $p(r_w)$ can be obtained in a similar fashion by smoothing with some constant value.

Finally, discounting functions for variable word rate models, analogous to those described in section 2, may be

$$\hat{f}_{gt}(r_{w|v}) = d_r \cdot f(r_{w|v}) \tag{32}$$

for the Good-Turing discounting and

$$\hat{f}_{abs}(r_{w|v}) = f(r_{w|v}) - \frac{e}{N} \tag{33}$$

for the absolute discounting. We note that zero prior information case (*i.e.*, $f(0)$'s) may be used to calculate discounting factors, d_r and e.

Table 4 shows the improvement achieved by variable word rate modelling. The document size was set to 1000 words for the unigram case, however, in order to obtain substantial improvement using bigram models, we need to handle the larger size of document (say, over 10 000 words). It is predictable because bigrams are orders of magnitude more sparse than unigrams.

Table 4. This table shows unigram perplexities using the constant (baseline) and the variable word rate models, using conditions described in figure 4. Perplexities were calculated for the reference transcription of the 1997 *Hub–4* evaluation data, containing three hours of speech (approximately 32 000 words). For the variable word rate model, parameters were adjusted 'on-line' — for each occurrence of a word in the evaluation data, a histogram of the past 1000 words was collected and relative frequencies were calculated using the Poisson estimates. Appropriate normalization, discounting and smoothing techniques were applied.

model	perplexity
constant word rate model	936.5
variable word rate model	845.8

5 Information Extraction

Simple statistical models underlie many successful applications of speech and language processing. The language model component of state-of-the-art large vocabulary speech recognition systems uses the n-gram approaches described in this chapter. The most accurate document retrieval systems are typically based on unigram statistics. Although these models are limited representationally, they are trainable and can be scaled to large corpora containing 10^9 words or more.

More recently, similar statistical finite state models have been developed for spoken language processing applications beyond direct transcription to enable, for example, a production of structured transcriptions [4], [17], [24], [31], [43]. This section discusses the development of trainable statistical models for *information extraction* from spoken language. In particular we concentrate on statistical finite state models for identifying proper names and other *named entities* (NE) in television and radio broadcast news.

Named Entities. Proper names account for around 9% of broadcast news output, and their successful identification would be useful for structuring the output of a speech recognizer (through punctuation, capitalization and tokenization), and as an aid to other spoken language processing tasks, such as summarization and database creation. The task of NE identification involves identifying and classifying those words or word sequences that may be classified as proper names, or as certain other classes such as monetary expressions, dates and times. This is not a straightforward problem. While 'Wednesday 1 September' is clearly a date, and 'Alan Turing' is a personal name, other strings, such as 'the day after tomorrow', 'Sheffield Linux Users' Group' and 'Nobel Prize' are more ambiguous. Here we consider seven classes of named entity (<location>, <person>, <organization>, <date>, <time>, <money> and <percentage>) which were defined for a recent *Hub–4* broadcast news evaluation [7]. According to this definition the following NE tags would be correct:

<date>Wednesday 1 September</date>
<person>Alan Turing</person>
the day after tomorrow
<organization>Sheffield Linux Users' Group</organization>
Nobel Prize

In this case 'The day after tomorrow' is not tagged as a date, since only 'absolute' time or date expressions are recognized; 'Nobel' is not tagged as a personal name, since it is part of a larger construct that refers to the prize. Similarly, 'Sheffield' is not tagged as a location since it is part of a larger construct tagged as an organization.

Both rule-based [19], [46] and statistical approaches have been used for NE identification, with some grammar-based systems employing probabilistic or trainable components [1], [29]. Bikel *et al.* introduced a trainable,. statistical system for NE identification [4] based on an ergodic HMM, in which the hidden states corresponded to NE classes, and the observed symbols corresponded to words.

A straightforward approach to identifying named entities in speech is to transcribe the speech automatically using a recognizer, then to apply a text-based NE identification method to the transcription. It is more difficult to identify NEs from automatically transcribed speech compared with text, since speech recognition output is missing features that may be exploited by 'hard-wired' grammar rules or by attachment to vocabulary items, such as punctuation, capitalization and numeric characters. More importantly, no speech recognizer is perfect, and spoken language is rather different from written language. Although planned, low-noise speech (such as dictation, or a news bulletin read from a script) can be recognized with a word error rate (*WER*) of less than 10%, speech which is conversational, in a noisy (or otherwise cluttered) acoustic environment or from a different domain may suffer a *WER* in excess of 40%. Additionally, the natural unit seems to be the phrase, rather than the sentence, and phenomena such as disfluencies, corrections and repetitions are common. It could thus be argued that statistical approaches, that typically operate with limited context and very little notion of grammatical constructs, are more robust than grammar-based approaches. Spoken NE identification was first demonstrated by Kubala *et al.* [24], who applied the model of [4] to the output of a broadcast news speech recognizer. An important conclusion of that work was that the error of an NE identifier degraded linearly with *WER*, with the largest errors due to missing and spuriously tagged names.

5.1 Finite State Model

In this section we outline a statistical framework for NE identification [17], which is closely related to that of Bikel *et al.* [4] and Palmer *et al.* [31].

Formulation. First, let V denote a vocabulary and C be a set of name classes. We consider that V is similar to a vocabulary for conventional speech recognition systems (*i.e.*, typically containing tens of thousands of words, and no case information or other characteristics). When there is no ambiguity, these named entities are referred to as 'name(s)'. As a convention here, a class <other> is included in C for those words not belonging to any of the specified names. Because each name may consist of one word or a sequence of words, we also include a marker <+> in C, implying that the corresponding word is a part of the same name as the previous word. The following example is taken from a human-generated reference transcription for the 1997 *Hub–4* evaluation data:

$$\text{at the } \underbrace{\text{ronald reagan center}}_{<\text{organization}>} \text{ in } \underbrace{\text{simi valley}}_{<\text{location}>} \; \underbrace{\text{california}}_{<\text{location}>}$$

The corresponding class sequence is

<other> <+> <organization> <+> <+> <other> <location> <+> <location>

because 'simi valley' and 'california' are considered two different names.

Class information may be interpreted as a word attribute (the left model of figure 7). Formally, we define a class-word token $<c,w> \in C \times V$ and consider a joint probability model

$$p(<c,w>_1,\ldots,<c,w>_m) = \prod_{i=1\ldots m} p(<c,w>_i \mid <c,w>_1,\ldots,<c,w>_{i-1}) \qquad (34)$$

that generates a sequence of class-word tokens $\{<c,w>_1,\ldots,<c,w>_m\}$. This formulation is best viewed as a straightforward extension to standard n-gram language modelling having implicit class transition[9].

[9] Denoting $e = <c,w>$, formulation (34) is identical to n-gram language models.

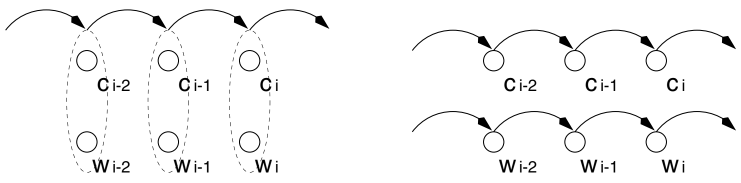


Fig. 7. Topologies for NE models. The left model assumes that class information is a word attribute. The right model explicitly models word-word and class-class transitions.

Unfortunately, this approach would not work as well as other statistical approaches can do. This is an example of a data sparsity problem that is observed in almost every aspect of spoken language processing. Although NE models cannot accommodate a complete set of parameters, a successful recovery of name expressions is heavily dependent on an existence of higher order n-grams. The implicit class transition approach contributes adversely to the data sparsity problem because it causes the set of possible tokens to increase in size from $|V|$ to $|C \times V|$.

Alternatively, word-word and class-class transitions may be explicitly formulated (the right model of figure 7). Then we consider a probability model

$$p(c_1, w_1, \dots, c_m, w_m) = \prod_{i=1\dots m} p(c_i, w_i \mid c_1, w_1, \dots, c_{i-1}, w_{i-1}) \tag{35}$$

that generates a sequences of words $\{w_1, \dots, w_m\}$ and a corresponding sequence of NE classes $\{c_1, \dots, c_m\}$. It is a state machine, but it cannot be considered as an HMM, as the probabilities are conditioned both on the previous word and on the previous class. It compensates for the fundamental sparseness of n-gram tokens in a vocabulary set; a bigram level modelling of form (35) outperforms a trigram model of form (34) [17].

Formulation (35) treats class and word tokens independently. Using bigram level constraints, it is reduced to

$$p(c_1, w_1, \dots, c_m, w_m) = \prod_{i=1\dots m} p(c_i, w_i \mid c_{i-1}, w_{i-1}) \ . \tag{36}$$

The right side of (36) may be decomposed as

$$p(c_i, w_i \mid c_{i-1}, w_{i-1}) = p(w_i \mid c_i, c_{i-1}, w_{i-1}) \cdot p(c_i \mid c_{i-1}, w_{i-1}) \ . \tag{37}$$

The conditioned current word probability $p(w_i \mid c_i, c_{i-1}, w_{i-1})$ and the current class probability $p(c_i \mid c_{i-1}, w_{i-1})$ are in the same form as a conventional n-gram, hence can be estimated from annotated text data[10].

[10] There exists an alternative approach to decomposing the right side of (36):

$$p(c_i, w_i \mid c_{i-1}, w_{i-1}) = p(c_i \mid w_i, c_{i-1}, w_{i-1}) \cdot p(w_i \mid c_{i-1}, w_{i-1}) \ .$$

Theoretically, if the ‘true’ conditional probability can be estimated, decompositions by (37) and above should produce identical results. This ideal case does not occur, and various discounting and smoothing techniques will cause further differences between two decompositions.

Smoothing Techniques. The amount of annotated data available is orders of magnitude smaller than the amount of text data typically used to estimate n-gram models for large vocabulary speech recognition. Smoothing the ML probability estimates is therefore essential to avoid zero probabilities, in which more specific models are smoothed with progressively less specific models. The following smoothing path can be chosen for the first term on the right side of (37):

$$p(w_i \mid c_i, c_{i-1}, w_{i-1}) \longrightarrow p(w_i \mid c_i, c_{i-1}) \longrightarrow p(w_i \mid c_i) \longrightarrow p(w_i) \longrightarrow \frac{1}{|W|} \quad .$$

$|W|$ is the size of the possible vocabulary that includes both observed and unobserved words from the training text data (*i.e.*, $|W|$ is sufficiently greater than $|V|$)[11]. Similarly, the smoothing path for the current class probability (the final term in (37)) may be

$$p(c_i \mid c_{i-1}, w_{i-1}) \longrightarrow p(c_i \mid c_{i-1}) \longrightarrow p(c_i) \quad .$$

This assumes that each class occurs sufficiently in training text data; otherwise, further smoothing to some constant probability may be required.

Given the smoothing path, the current word probability may be computed using an interpolation method described in section 2:

$$\begin{aligned} p(w_i|c_i, c_{i-1}, w_{i-1}) &= \hat{f}(w_i|c_i, c_{i-1}, w_{i-1}) + \{1 - \alpha(c_i, c_{i-1}, w_{i-1})\} \cdot p(w_i|c_i, c_{i-1}) \\ \text{where} \quad \alpha(c_i, c_{i-1}, w_{i-1}) &= \sum_{w_i \in E(c_{i-1}, w_{i-1}, c_i, w_i)} \hat{f}(w_i \mid c_i, c_{i-1}, w_{i-1}) \end{aligned} \tag{38}$$

where $\hat{f}(w_i \mid c_i, c_{i-1}, w_{i-1})$ is a discounted relative frequency, $\alpha(c_i, c_{i-1}, w_{i-1})$ is a non-zero probability estimate and $E(c_{i-1}, w_{i-1}, c_i, w_i)$ implies the event such that current class c_i and word w_i occur after previous class c_{i-1} and word w_{i-1}.

Alternatively, the back-off smoothing method of [23] could be applied:

$$\begin{aligned} p(w_i \mid c_i, c_{i-1}, w_{i-1}) &= \begin{cases} \hat{f}(w_i \mid c_i, c_{i-1}, w_{i-1}) & \text{if } E(c_{i-1}, w_{i-1}, c_i, w_i) \text{ exists} \\ \beta(c_i, c_{i-1}, w_{i-1}) \cdot p(w_i \mid c_i, c_{i-1}) & \text{otherwise} \end{cases} \\ \text{where} \quad \beta(c_i, c_{i-1}, w_{i-1}) &= \frac{1 - \alpha(c_i, c_{i-1}, w_{i-1})}{1 - \sum_{w_i \in E(c_{i-1}, w_{i-1}, c_i, w_i)} \hat{f}(w_i \mid c_i, c_{i-1})} \end{aligned} \tag{39}$$

This second decomposition alone would not work as well as the initial decomposition. Crudely speaking, it calculates the distribution over classes for each word; consequently it would reduce accuracy for uncommon words with less reliable probability estimates. Decomposition by (37) makes a more balanced decision because it relies on the distribution over words for each class, and usually there are orders of magnitude fewer classes than words.

[11] Smoothing to $p(w_i \mid c_i, c_{i-1})$ would probably produce a better results than smoothing to $p(w_i \mid c_i, w_{i-1})$, since the former could be more accurately estimated from the annotated training data.

where $\beta(c_i, c_{i-1}, w_{i-1})$ is a back-off factor. Using standard discounting techniques described in section 2, discounted relative frequencies and non-zero probability estimates can be obtained from the training data[12].

Given a sequence of words $\{w_1, \dots, w_m\}$, named entities can be identified by searching the Viterbi path such that

$$<\hat{c}_1 \dots \hat{c}_m> = \underset{c_1 \dots c_m}{\operatorname{argmax}} \, p(c_1, w_1, \dots, c_m, w_m) \ . \tag{40}$$

Although the smoothing scheme should handle novel words well, the introduction of conditional probabilities for <unk> (which represents those words not included in the vocabulary V) may be used to model unknown words directly. In practice, this is achieved by setting a certain cutoff threshold when estimating discounting probabilities. Those words that occur less than this threshold are treated as <unk> tokens. This does not imply that smoothing is no longer needed, but that conditional probabilities containing the <unk> token may occasionally pick up the context correctly without smoothing with weaker models. The drawback is that some uncommon words are lost from the vocabulary.

5.2 Experiment

NE identification systems are evaluated using an unseen set of evaluation data; the hypothesized NEs are compared with human annotated NE tags in a reference transcription[13]. In this situation there are two possible types of error: *type*, where an item is tagged as the wrong kind of entity and *extent*, where the wrong number of word tokens are tagged. A third error type *content* arises from speech recognition errors. These three error types each contribute $1/3$ to the overall error count, and precision (*P*) and recall (*R*) can be calculated in a usual way. A weighted harmonic mean (*P&R*), sometimes referred to as the F-measure [45], is often calculated as a single summary statistic:

$$P\&R = \frac{2RP}{R+P} \ .$$

P&R has the disadvantage of deweighting missing and spurious identification errors compared with incorrect identification errors [27]. The slot error rate (*SER*) is an alternative measure that weights the three types of identification error equally. Analogous to *WER*, *SER* may be obtained by:

$$SER = \frac{I+M+S}{C+I+M}$$

where C, I, M, and S denote the numbers of correct, incorrect, missing, and spurious identifications. Using this notation, precision and recall scores may be calculated as $R = C/(C+I+M)$ and $P = C/(C+I+S)$, respectively.

[12] The weaker models—$p(w_i \mid c_i, c_{i-1})$, $p(w_i \mid c_i)$, and $p(w_i)$—may be smoothed in a way analogous to that used for $p(w_i \mid c_i, c_{i-1}, w_{i-1})$. Approaches to discounting and combining with different level models are similar when handling the conditioned current class probabilities, *i.e.*, $p(c_i \mid c_{i-1}, w_{i-1})$, $p(c_i \mid c_{i-1})$, and $p(c_i)$.

[13] Inter-annotator agreement for reference transcriptions is around 97–98% [39].

Table 5 shows NE identification scores by the statistical finite state machine approach. The model parameters with explicit class-class and word-word transitions were derived from the relatively small amount of training data (one million words). Like other language modelling problems, a simple way to improve the performance is to increase the amount of training data. As a final note, it should not be under-stated that an appropriate choice and implementation of discounting/smoothing strategies is very important, particularly because a more complex model structure is being trained with less data, compared with conventional language models for speech recognition systems.

Table 5. The table compares NE identification scores on hand transcription (no word error) and speech recognizer output (21% *WER*). A finite state model was derived from the *Hub–4* training data, consisting of about one million words of transcripts having manual NE annotation. It selected 17,560 words (from those occurring more than once in the training data) as a vocabulary and the rest (those occurring exactly once — nearly 10,000 words) were replaced by the <unk> token. The combined Good-Turing/absolute discounting scheme was applied, followed by back-off smoothing. Further detail may be found in [17].

	WER	*R*	*P*	*P&R*	*SER*
hand transcription	.000	.863	.922	.892	.187
recognizer output	.210	.729	.823	.773	.381

6 Summary

This chapter first outlined the basics of statistical language models (*e.g.*, estimating from sparse training data, encapsulating varying contents and styles of spoken language), then discussed a more recent area for applying n-gram based approaches such as named entity extraction.

Constraints by the Markov assumption, that enforces the local structure of (spoken) language, achieve success only to some extent, but their fundamental brittleness may be alleviated by incorporating richer, and more linguistically motivated models. Chelba and Jelinek have addressed the use of probabilistic dependency grammar [6], where the probability of each word is estimated from several other words that, unlike conventional n-gram models, are not necessarily those immediately preceeding the word.

Another area, that is not discussed in this chapter but worth noting, is the maximum entropy approach. It is a conceptually clean way to model information from multiple sources. It is recently applied to areas such as parsing [34], statistical machine translation [2], [32], and the incorporation of trigger word pair constraints into an n-gram language model [40]. Although computationally challenging, series of successful applications seem to indicate the potential of the framework.

References

1. J. Aberdeen, J. Burger, D. Day, L. Hirschman, P. Robinson, and M. Vilain. MITRE: Description of the Alembic system used for MUC-6. In *Proceedings of the 6th Message Understanding Conference (MUC-6)*, pages 141–1552, November 1995.
2. Adam L. Berger, Stephen A. Della Pietra, , and Vincent J. Della Pietra. A maximum entropy approach to natural language processing. *Computational Linguistics*, 22(1):39–71, March 1996.
3. Michael W. Berry, Susan T. Dumais, and Gavin W. O'Brien. Using linear algebra for intelligent information retrieval. *SIAM Review*, 37(4):573–595, 1995.
4. Daniel M. Bikel, Richard Schwartz, and Ralph M. Weischedel. An algorithm that learns what's in a name. *Machine Learning*, 211–231:1999, 34.
5. Lou Burnard. Users reference guide, British National Corpus version 1.0. Oxford University Computing Service, May 1995.
6. Ciprian Chelba and Frederick Jelinek. Structured language modeling. *Computer Speech and Language*, 14:283–332, 2000.
7. Nancy Chinchor, Patty Robinson, and Erica Brown. Hub-4 named entity task definition (version 4.8). SAIC - http://www.nist.gov/speech/hub4_98/hub4_98.htm, August 1998.
8. Kenneth W. Church and William A. Gale. A comparison of the enhanced Good-Turing and deleted estimation methods for estimating probabilities of English bigrams. *Computer Speech and Language*, 5:19–54, 1991.
9. Kenneth W. Church and William A. Gale. Poisson mixtures. *Natural Language Engineering*, 1(2):163–190, June 1995.
10. P. R. Clarkson and A. J. Robinson. Language model adaptation using mixtures and an exponentially decaying cache. In *Proceedings of ICASSP-97*, volume 2, pages 799–802, April 1997.
11. Scott Deerwester, Susan T. Dumais, George W. Furnas, Thomas K. Landauer, and Richard Harshman. Indexing by latent semantic analysis. *Journal of the Society for Information Science*, 41(6):391–407, 1990.
12. Morris H. DeGroot. *Optimal Statistical Decisions*. McGraw Hill, New York, NY, 1970.
13. A. P. Dempster, N. M. Laird, and D. B. Rubin. Maximum likelihood from incomplete data via the EM algorithm. *Journal of the Royal Statistical Society, series B*, 39(1):1–38, 1977.
14. Marcello Federico, Mauro Cettolo, Fabio Brugnara, and Giuliano Antoniol. Language modelling for efficient beam-search. *Computer Speech and Language*, 9:353–379, 1995.
15. W. B. Frakes and R. Baeza-Yates. *Information Retrieval: Data Structures and Algorithms*. Prentice Hall, Englewood Cliffs, NJ, 1992.
16. Yoshihiko Gotoh and Steve Renals. Topic-based mixture language modelling. *Natural Language Engineering*, 5(4):355–375, December 1999.
17. Yoshihiko Gotoh and Steve Renals. Information extraction from broadcast news. *Philosophical Transactions of the Royal Society, series A*, 358:1295–1310, April 2000.
18. Stephen P. Harter. A probabilistic approach to automatic keyword indexing (part 1). *Journal of the American Society for Information Sceince*, 26(4):197–206, July 1975.
19. J. Hobbs, D. Appelt, J. Bear, D. Israel, M. Kameyama, M. Stickel, and M. Tyson. FASTUS: A cascaded finite state transducer for extracting information from natural language text. In *Finite State Language Processing*, pages 381–406. MIT Press, 1997.
20. F. Jelinek and R. L. Mercer. Interpolated estimation of Markov source parameters from sparse data. In *Proceedings of the Workshop: Pattern Recognition in Practice*, pages 381–397, May 1980.
21. Frederick Jelinek. Up from trigrams! The struggle for improved language models. In *Proceedings of Eurospeech-91*, volume 3, pages 1037–1040, September 1991.

22. Frederick Jelinek. *Statistical Methods for Speech Recognition*. MIT Press, Cambridge, MA, 1997.
23. Slava M. Katz. Estimation of probabilities from sparse data for the language model component of a speech recognizer. *IEEE Transactions on Acoustics, Speech and Signal Processing*, 35(3):400–401, March 1987.
24. Francis Kubala, Richard Schwartz, Rebecca Stone, and Ralph Weischedel. Named entity extraction from speech. In *Proceedings of DARPA Broadcast News Transcription and Understanding Workshop*, 1998.
25. R. Kuhn and R. De Mori. A cache-based natural language model for speech recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 12(6):570–583, June 1990.
26. Stephen A. Lowe. The beta-binomial mixture model for word frequencies in documents with applications to information retrieval. In *Proceedings of Eurospeech-99*, volume 6, pages 2443–2446, September 1999.
27. John Makhoul, Francis Kubala, Richard Schwartz, and Ralph Weischedel. Performance measures for information extraction. In *Proceedings of DARPA Broadcast News Workshop*, pages 249–252, February 1999.
28. Christopher D. Manning and Hinrich Schütze. *Foundation of Statistical Natural Language Processing*. MIT Press, Cambridge, MA, 1999.
29. A. Mikheev, C. Grover, and M. Moens. Description of the LTG system used for MUC-7. In *Proceedings of the 7th Message Understanding Conference (MUC-7)*, 1998.
30. Hermann Ney, Ute Essen, and Reinhard Kneser. On the estimation of 'small' probabilities by leaving-one-out. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 17(12):1202–1212, December 1995.
31. David D. Palmer, Mari Ostendorf, and John D. Burger. Robust information extraction from automatically generated speech transcriptions. *Speech Communication*, 32(1/2):95–109, September 2000.
32. K. A. Papineni, S. Roukos, and R. T. Ward. Maximum likelihood and discriminative training of direct translation models. In *Proc. IEEE ICASSP*, pages 189–192, 1998.
33. Athanasios Papoulis. *Probability, Random Variables, and Stochastic Processes*. McGraw Hill, New York, NY, 2nd edition, 1984.
34. Adwait Ratnaparkhi. Learning to parse natural language with maximum entropy models. *Machine Learning*, 34, 1999.
35. Richard A. Redner and Homer F. Walker. Mixture densities, maximum likelihood and the EM algorithm. *SIAM Review*, 26(2):195–239, April 1984.
36. S. Renals, D. Abberley, D. Kirby, and T. Robinson. Indexing and retrieval of broadcast news. *Speech Communication*, 32(1/2):5–20, September 2000.
37. S. E. Robertson and K. Spärck Jones. Simple, proven approaches to text retrieval. Technical Report TR356, University of Cambridge, Computer Laboratory, 1997. http://www.ftp.cl.cam.ac.uk/ftp/papers/reports/.
38. S. E. Robertson and S. Walker. Some simple effective approximations to the 2-Poisson model for probabilistic weighted retrieval. In *Proceedings of SIGIR-94*, pages 232–241, 1994.
39. Patricia Robinson, Erica Brown, John Burger, Nancy Chinchor, Aaron Douthat, Lisa Ferro, and Lynette Hirschman. Overview: Information extraction from broadcast news. In *Proceedings of DARPA Broadcast News Workshop - Herndon, VA*, pages 27–30, February 1999.
40. Ronald Rosenfeld. A maximum entropy approach to adaptive statistical language modeling. *Computer Speech and Language*, 10:187–228, 1996.
41. C. E. Shannon. A mathematical theory of communication. *Bell System Technical Journal*, 27(3,4):379–423 and 623–656, 1948.
42. C. E. Shannon. Prediction and entropy of printed English. *Bell System Technical Journal*, 30(1):50–64, January 1951.

43. Elizabeth Shriberg, Andreas Stolcke, Dilek Hakkani Tür, and GükhanTür. Prosody modeling for automatic sentence and topic segmentation from speech. *Speech Communication*, 32(1/2):127–154, September 2000.
44. K. Spärck Jones, S. Walker, , and S. E. Robertson. A probabilistic model of information retrieval: Development and status. Technical report, Technical Report TR446, University of Cambridge, Computer Laboratory - Available from http://www.ftp.cl.cam.ac.uk/ftp/papers/reports/, 1998.
45. C. J. van Rijsbergen. *Information Retrieval*. Butterworths, London, 2nd edition, 1979.
46. T. Wakao, R. Gaizauskas, , and Y. Wilks. Evaluation of an algorithm for the recognition and classification of proper names. In *Proceedings of COLING-96*, pages 418–423, 1996.
47. P. C. Woodland, T. Hain, G. L. Moore, T. R. Niesler, D. Povey, A. Tuerk, and E. W. D. Whittaker. The 1998 HTK broadcast news transcription system: Development and results. In *Proceedings of DARPA Broadcast News Workshop*, pages 265–270, February 1999.
48. C. T. Yu and G Salton. Effective information retrieval using term accuracy. *Communications of the ACM*, 20:135–142, 1977.