2013-04-16

Probability of Belonging to a Language

Kevin Michael Brooks Cook
Brigham Young University - Provo

Probability of Belonging to a Language

Kevin M. B. Cook

A thesis submitted to the faculty of
Brigham Young University
in partial fulfillment of the requirements for the degree of

Master of Science

Kevin D. Seppi, Chair
Eric K. Ringger
Robert P. Burton

Department of Computer Science

Brigham Young University

April 2013

ABSTRACT

Probability of Belonging to a Language

Kevin M. B. Cook
Department of Computer Science, BYU
Master of Science

Conventional language models estimate the probability that a word sequence within a chosen language will occur. By contrast, the purpose of our work is to estimate the probability that the word sequence belongs to the chosen language. The language of interest in our research is comprehensible well-formed English. We explain how conventional language models assume what we refer to as a *degree of generalization*, the extent to which a model generalizes from a given sequence. We explain why such an assumption may hinder estimation of the probability that a sequence belongs. We show that the probability that a word sequence belongs to a chosen language (represented by a given sequence) can be estimated by avoiding an assumed degree of generalization, and we introduce two methods for doing so: *Minimal Number of Segments (MINS)* and *Segment Selection*. We demonstrate that in some cases both MINS and Segment Selection perform better at distinguishing sequences that belong from those that do not than any other method we tested, including Good-Turing, interpolated modified Kneser-Ney, and the Sequence Memoizer.

Keywords: degree of generalization, language model, Minimal Number of Segments (MINS), probability of belonging, Segment Selection, word sequence

Contents

Chapter 1

Introduction

Language models estimate the probability of a sequence of words. Word sequence probability is a key component in many applications, including speech recognition, machine translation, and text compression.

Conventional language models estimate the probability that a word sequence within a chosen language will occur. By contrast, the purpose of our work is to estimate the probability that the word sequence belongs to the chosen language. The language of interest in our research is comprehensible well-formed English.

In speech recognition and in machine translation it may be desirable to estimate the probability that a sequence belongs to a chosen language, in order to produce text output that belongs to that language. Certainly the use of language models in such systems can help to produce output that belongs, but only to the extent that an estimate of the probability that a sequence occurs is useful as an estimate of the probability that the sequence belongs. Other applications involving the recognition of word sequences, such as optical character recognition and handwriting recognition, may also benefit from an estimate of the probability that a word sequence belongs to a chosen language.

1.1 Language Models and Smoothing

The primary difference among language models is the method of smoothing they employ. Smoothing is a means of generalizing from training data. Language models are often named after their method of smoothing [8] [9]. The names Good-Turing and Kneser-Ney each reference both a means of

smoothing and a language model. Bayesian models, such as the Sequence Memoizer, have been introduced as a more principled approach to smoothing [17] [19].

Assuming smoothing parameters of a language model is a way of assuming what we refer to as a *degree of generalization*, which is a more general notion than smoothing describing the extent to which a model generalizes from a given sequence. To illustrate what we mean by a degree of generalization, consider a model that assumes a high degree of generalization, a simple unigram (i.e., bag-of-words) language model. This model assumes that each word in a sequence is independent of previous words. If this model were trained on the introduction of this paper and used to generate a sequence, it might stochastically generate the following sequence: *"language a already probability sequence the the"*. On the other hand, consider a model that assumes a very low degree of generalization, one that assigns all probability to the sequence that is the training data and no probability to any other sequence. If this second model were again trained on the introduction of this paper and used to generate a sequence, it would generate the introduction of this paper.

Language models assume a degree of generalization in various ways. N-gram models, such as Good-Turing and Kneser-Ney, set aside probability for unseen words, set aside probability for words that are unseen in a given context, and make Markov independence assumptions, ignoring context earlier than a set number of recent words. The Sequence Memoizer does not make traditional Markov assumptions but does assume a degree of generalization using what are known as discount parameters.

Conventional language models estimate the probability that a word sequence occurs given a language, rather than the probability that the sequence belongs to the language. An example may help to clarify the distinction between the probability of occurring and the probability of belonging: Suppose again that we wish to model the language of this paper. We will use the introduction of this paper as an example sequence to represent the language and then discuss the estimation of probabilities for sequences which occur later in this paper. We begin with the sequence *"the probability"*. Since this sequence occurs several times in the example sequence, a conventional language model will assign high probability to seeing it again later in this paper.

Now consider the sequence which is the introduction itself. Since that entire sequence occurs only once in the example sequence (itself), a conventional language model will assign relatively low probability to seeing it again later in this paper. (It would seem highly unlikely to find such an oddly structured and repetitive paper.) Although we would say that the introduction has a low probability of occurring (again), we would also say that it has a high probability of belonging to the language: the introduction belongs to the language of this paper by definition.

A conventional model assumes that a sequence belongs to a language and uses that assumption to estimate the probability that the sequence occurs. To see that a typical language model assumes that a sequence belongs to the language from which the model was trained, consider that withholding a portion of training data to smooth a conventional language model embodies the assumption that the withheld portion is representative of an unseen sequence, which implies that training data is representative of the sequence, or that the sequence belongs to the language represented by training data.

If a model assumes that a sequence belongs to a language, then the probability that the sequence belongs to that language is by definition equal to one, because it is already assumed to be true. Since we are interested in estimating the probability that a sequence belongs, we want to avoid assuming that it belongs. Since the justification for assuming a degree of generalization rests on the assumption that the sequence belongs, then we also want to avoid assuming a degree of generalization, which reasoning motivates the thesis statement (see Chapter 3), which is that an estimate of the probability that a word sequence belongs to a chosen language (represented by a given sequence) can be constructed by avoiding an assumed degree of generalization.

1.2 Probability of Belonging

We introduce two strategies for avoiding an assumed degree of generalization to construct an estimate of the probability that a word sequence belongs to a chosen language. Our language of interest can be considered a potentially infinite set of potentially infinite word sequences. The probability we are interested in estimating is the probability that a word sequence belongs to this

set. If training data includes a sequence known to belong to this set, then the probability that the sequence belongs is one. Otherwise, this probability is unknown, since training data may not include all sequences in the set. An estimate of such an unknown probability is necessarily subjective. We state the assumptions inherent in each of the strategies we employ to estimate the probability that a word sequence belongs to a chosen language.

We estimate this probability given an example sequence representative of the language. Given that the example sequence belongs, we know the following: if the example sequence is used to generate a sequence using the lowest possible degree of generalization, namely no generalization, then the generated sequence also belongs, because it is identical to the example sequence. We also know that knowledge of the degree of generalization used to generate a sequence can affect knowledge of whether the sequence belongs or not, since we know that it does belong if no generalization is used.

1.2.1 MINS

In Chapter 4 we describe our first strategy for avoiding an assumed degree of generalization, which is to search for the lowest degree of generalization which could possibly be used to generate a sequence from an example sequence, and then to use that degree to estimate a probability for the sequence.

We assume the following: when that lowest possible degree is high, then the probability is low that the sequence belongs.

1.2.2 Segment Selection

In Chapter 5 we describe our second strategy for avoiding an assumed degree of generalization, which is essentially to consider a set of models which assume a broad range of degrees of generalization, from very low to very high. Each model estimates the probability of a sequence, given its assumed degree of generalization. Each degree of generalization has the same prior probability, in other words, one degree is not assumed to be more probable than another. Held-out data is not used

to find a most likely degree of generalization. The probability that the sequence is generated from an example sequence, without assuming a degree of generalization, is estimated by marginalizing over all degrees of generalization, the sum of the products of the probability estimated by each model times the prior probability of the degree of generalization assumed by the model.

We assume the following: when the probability is low that the sequence is generated from the example sequence without assuming a degree of generalization, then the probability is low that the sequence belongs.

1.3 Experiments

In Chapter 6, we validate our thesis, demonstrating the effectiveness of MINS and Segment Selection by using each method to distinguish word sequences belonging to the language of interest from those that do not. We also show that in some cases, both MINS and Segment Selection perform better at making this distinction than any of the other methods we tested.

Chapter 2

Related Work

We explain how our work relates to text classification, various string similarity measures, and discriminative language models.

2.1 Text Classification

We discuss similarities and differences between our problem of estimating the probability that a word sequence belongs to a chosen language and the problem of text classification.

Text classification is the problem of inferring the class of a document, which appears equivalent to the problem of inferring the language to which a word sequence belongs, since a word sequence can be considered a text document and a language can be considered a class. However, the difference between the two problems is a type vs. token distinction: Consider the difference between a word sequence (i.e., the type) and an instance of a word sequence (i.e., the token). Our problem is concerned with the word sequence itself, not a specific instance thereof. For example, we are interested in estimating the probability that the sequence *"reports confirm suspicions"* belongs to our language of interest, comprehensible well-formed English. We are not concerned with any specific instance of that sequence, such as one which may appear in political news or another which may appear in a scientific paper. By contrast, a typical classification task is to infer the source of a specific instance of a word sequence, such as to infer whether it comes from a political news source, or whether it comes from a scientific journal. Estimating the probability that an instance of a word sequence belongs to a language is different than our problem, which is to estimate the probability that the sequence of words, regardless of source, belongs to the language.

To illustrate this type vs. token distinction, consider language identification, the task of classifying documents by language, such as English or German. In such a task it may be desirable to classify documents by language (concerned with document type), rather than by source (concerned with document token). For example, if an English article is published in a German magazine, we may want to identify the article as English, rather than as German. If so, then it may be useful to estimate the probability that the article belongs to the English language. However, if it is desirable to classify documents by source, then it may be useful to estimate the probability that an English article appears in the German magazine, or to estimate the probability that a word sequence occurs in the German magazine regardless of the language to which the word sequence belongs.

Although a text document may be equivalent to a word sequence, in text classification it is common to represent a document as a bag-of-words, a vector of word counts, which representation is obviously not a word sequence.

The class of interest in our work is our language of interest. We are interested in distinguishing sequences that belong to this class from those that do not, given only positive training examples. The Positive Naive Bayes classifier (PNB) [6] is an example of a one-class (or unary) classifier. This classifier requires an estimate of the prior probability of the positive class. It also requires unlabeled data, documents which are representative of the entire space of possible documents, both those that belong and those that do not. The classifier uses bag-of-words models to estimate the probability that a document occurs, both in the positive and negative classes. Bayes law is then used to estimate the probability that the document instance belongs, in order to make a classification decision. Generally, the PNB is not applied to the problem of estimating the probability that a word sequence instance belongs to our language of interest, or any similar language. One reason for not doing so may be the sparse data problem: it is difficult to acquire sufficient data to accurately estimate the probability that a sequence occurs, especially when considering longer contexts.

2.2 String Similarity

Various string similarity measures may be useful for estimating the probability that a sequence belongs. Our thesis is that this probability can be estimated by avoiding an assumed degree of generalization. In order for us to validate our thesis with a string similarity measure, we must use one which does not assume a degree of generalization.

We describe various ways in which string similarity measures assume a degree of generalization:

2.2.1 Cosine Distance

Cosine distance assumes a high degree of generalization by assuming that a string can be adequately represented by a vector of word counts.

Consider a long string generated from a simple unigram language model. The cosine distance between that string and the example (or training) string from which the model was trained is likely to be very low, since cosine distance ignores word order, only considering the frequency of individual words.

2.2.2 Edit Distance

Edit distance assumes a high degree of generalization by assuming that edit operators are defined at the scope of one word or character.

Consider a string created by splitting a source string in half, and then concatenating the two halves in opposite order. If edit distance is used to calculate similarity between the two strings, the strings can be aligned so that they overlap by at least 50%. However, the difference with the remaining portion of either string is measured in terms of one edit per word or character, rather than in terms of a single substring edit.

As a further example, consider a string created by duplicating some substring in a source string. If edit distance is used to calculate similarity, the portion of the source string which does not align with the substring has a distance measured in terms of one edit per word or character, rather

than in terms of one or two long deletions. If the source string is long and the substring is short, the edit distance will be great, in spite of the fact that the substring belongs to the source string by definition.

2.2.3 N-grams

It is possible to treat a word n-gram as a single feature, or word, to calculate similarity. Any method which limits the length of word n-grams to be considered, assumes a degree of generalization by doing so. The same is true of character n-grams [1, 3, 10, 12].

2.2.4 Text Compression

Text compression has been used as a means for producing a string similarity metric [2]. One way that text compression methods can assume a degree of generalization is by bounding context length. A method has been published to consider unbounded context length for text compression [5], however a degree of generalization is assumed using escape probabilities, somewhat analogous to discount parameters of the Sequence Memoizer.

2.3 Discriminative Language Models

Discriminative estimation of language models [15] has been proposed as a means of reducing speech recognition error rates. Such models are used to rank word sequences in an attempt to directly reduce error rates. Models are trained using a set of acoustic sequences together with corresponding transcriptions.

We suppose that an estimate of the probability that a word sequence belongs may be used to rank word sequences in an attempt to reduce speech recognition error rates, which may be the subject of future work. However, we do not require transcribed acoustic sequences in order to estimate this probability.

Chapter 3

Thesis Statement

An estimate of the probability that a word sequence belongs to a chosen language (represented by a given sequence) can be constructed by avoiding an assumed degree of generalization.

Chapter 4

Minimal Number of Segments (MINS)

Our first strategy for avoiding an assumed degree of generalization is to search for a minimum possible degree of generalization. To implement this strategy, we define a sequence measure called Minimal Number of Segments (MINS) and denoted as $mins(\mathbf{x} \rightarrow \mathbf{y})$ as the minimal number of segments required to generate a sequence of characters $\mathbf{y}$ by concatenating segments found in a sequence of characters $\mathbf{x}$, where each segment is a contiguous sequence of characters. If it is not possible to do so, then the number is undefined. We first motivate the measure, and then describe a method to calculate the measure.

4.1 Probability of Belonging

We estimate a probability that $\mathbf{y}$ belongs to a language l, given that $\mathbf{x}$ belongs to l, by the exponentiation of the product of the minimal number of segments minus one and an arbitrary negative constant k:

$$\hat{P}_{mins}(\mathbf{y} \in l | \mathbf{x} \in l, k) \equiv \begin{cases} \exp\left(k(mins(\mathbf{x} \rightarrow \mathbf{y}) - 1)\right) & \text{if } mins \text{ is defined} \\ 0 & \text{otherwise} \end{cases} \tag{4.1}$$

We posit that this estimate (Equation 4.1) is useful for distinguishing sequences that belong from those that do not, because the calculation of the MINS measure does not assume any relationship between $\mathbf{y}$ and l, such as assuming that $\mathbf{y}$ belongs to l, that $\mathbf{y}$ is generated from $\mathbf{x}$ (known to belong to l), or that $\mathbf{y}$ is generated from $\mathbf{x}$ with an assumed degree of generalization. Rather, MINS is a measure of a minimal degree of generalization required to generate $\mathbf{y}$ from $\mathbf{x}$.

The reason we subtract one from MINS is so that when $\mathbf{y} = \mathbf{x}$, the probability that $\mathbf{y}$ belongs to l is one, which agrees with the fact that $\mathbf{x} \in l$ is given. It is also true that when $\mathbf{y}$ is a subsequence of $\mathbf{x}$, that $\hat{P}_{mins}$ equals one, which agrees with a reasonable assumption that if $\mathbf{x}$ belongs then a subsequence of $\mathbf{x}$ also belongs. (If a subsequence of $\mathbf{x}$ does not belong, then in that case $\hat{P}_{mins}$ is a very poor estimate.) Note that $0 \leq \hat{P}_{mins} \leq 1$, since $mins$ (when defined) is a positive number equal to or greater than one and k is negative, and the exponentiation of any number equal to or less than zero results in a value within that interval.

In order to use MINS to estimate the probability of a sequence, it is necessary to choose a value of k. As long as the value of k is the same over all sequences, the value of k affects neither the ranking of sequences by probability, nor the ratio of the log probability of one sequence to the log probability of another sequence, which ratio we use in Chapter 6 to demonstrate usefulness of MINS for distinguishing sequences that belong from those that do not.

4.2 The Method

The method to calculate $mins(\mathbf{x} \rightarrow \mathbf{y})$ is shown in Figure 4.1. Beginning at position $j = 1$ in $\mathbf{y}$, and proceeding to the right, the method searches $\mathbf{x}$ for the longest matching segment. It then increments j by the length of this segment and repeats this process, starting at j in $\mathbf{y}$, searching $\mathbf{x}$ for the longest matching segment. The method terminates when the end of $\mathbf{y}$ is reached. If successful, the method returns the number of segments.

Theorem 1. *The method described above (shown in Figure 4.1) finds the minimal number of segments* $mins(\mathbf{x} \rightarrow \mathbf{y})$.

> *Proof.* To see that the method finds the minimal number of segments, observe that by definition when the search is begun at position j, the maximum length match is of length v and that it is not possible for any segment in $\mathbf{x}$ to match the longer segment in $\mathbf{y}$ beginning at position j extending to and including position $j + v$. The last position matched is $j + v - 1$. Suppose a search begins before position j. Further suppose this

Input: $\mathbf{x}$, $\mathbf{y}$

Output: $mins$ *if defined*

Set $j = 1$ and $mins = 0$

Repeat until $j = \|\mathbf{y}\| + 1$ or until FAIL

    if $\mathbf{y}[j]$ exists in $\mathbf{x}$

        $v = \max v' : \mathbf{y}[j..(j + v' - 1)] \subseteq \mathbf{x}$
        $j = j + v$
        $mins = mins + 1$

    else FAIL

if FAIL then $mins$ is undefined

else return $mins$

Figure 4.1: Method to calculate $mins(\mathbf{x} \rightarrow \mathbf{y})$.

search finds a match that extends to and includes position $j + v$. This search would then have also found a match beginning at position j and extending to and including position $j + v$, which is already known not to exist. It is possible for a search to begin before position j and end at position $j + v - 1$, but this does not affect the minimal number of segments required, since all positions prior to j have either already been matched, or in the case of $j = 1$ do not exist. If the search were to begin at any position after j, there would be no guarantee that position j would match. □

4.2.1 Searching

Calculation of $mins$ involves multiple searches of $\mathbf{x}$. To facilitate search, we index $\mathbf{x}$ using a suffix tree, which can be done on-line in $O(\|\mathbf{x}\|)$ time and space [18], as mentioned earlier.

To search $\mathbf{x}$ for the length of a longest matching segment, we follow the method in Figure 4.2. The method begins at the root of the tree. It starts at position j in $\mathbf{y}$ and matches as many characters as possible, following matching branches in the suffix tree. (The edges connecting nodes in a suffix tree represent sequences of one or more characters, as shown in Figure 4.3.)

Each search requires time in $O(v)$. The calculation of $mins$ requires time in $O(\|\mathbf{y}\|)$.

Input: $\mathbf{y}$, *suffix tree of* $\mathbf{x}$

Output: v

Set $v = 0$

Set *node* to root of suffix tree of $\mathbf{x}$

Repeat until FINISHED

    If exists branch $b : b[1] = \mathbf{y}[j + v]$ then

        $k = 1$

        Repeat until DONE

            If $b[1 + k] = \mathbf{y}[j + v + k]$ then $k = k + 1$
            else $v = v + k$ and DONE

        If $k = \|b\|$ then

            Set *node* to *child*, following branch b

        else FINISHED

    else FINISHED

Return v

Figure 4.2: Method to search for the length of a longest matching segment

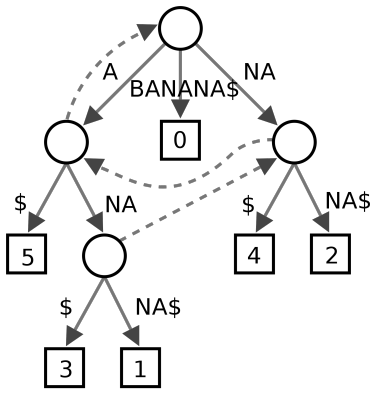


Figure 4.3: Illustration of a suffix tree for the sequence BANANA (from wikipedia.org)

Chapter 5

Segment Selection

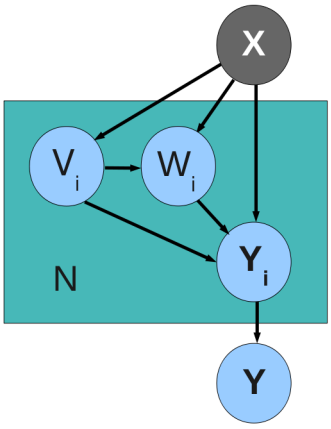
Our second strategy for avoiding an assumed degree of generalization is to marginalize over a broad range of degrees of generalization. To implement this strategy, we describe a probabilistic model for discrete sequence data called Segment Selection, to estimate what we refer to as the *marginal*, which is the probability that the sequence is generated from a given sequence without assuming a degree of generalization. We use the marginal to estimate the probability that a sequence belongs to the language represented by the given sequence, by assuming a relationship between these two probabilities, between the marginal and the probability that the sequence belongs. We first explain our motivation for assuming that relationship, and then describe the model in detail along with methods for inference.

5.1 The Model

We introduce the Segment Selection model (Figure 5.1) in order to estimate the marginal probability of an arbitrary word sequence $\mathbf{a}$, the probability that $\mathbf{Y} = \mathbf{a}$, given that $\mathbf{Y}$ is generated from a given sequence $\mathbf{x}$ without assuming a degree of generalization. No assumption is made that $\mathbf{a}$ is generated from $\mathbf{x}$. We refer to the marginal probability of $\mathbf{a}$ as $P(\mathbf{Y} = \mathbf{a}|\mathbf{x})$, or simply as the marginal $P(\mathbf{y}|\mathbf{x})$.

5.1.1 Probability of Belonging

To estimate the probability that $\mathbf{a}$ belongs to language l, we assume that the probability that $\mathbf{a}$ belongs to l, given that $\mathbf{x}$ belongs to l, is equal to the ratio of the marginal probability of $\mathbf{a}$ to the



$\mathbf{x}$ Given sequence of characters

N Number of segments
$N|c \sim U(1, c)$ where c is an arbitrarily large constant

i Segment index
$(1 \leq i \leq n)$

V_i Length of segment i in characters
$V_i|\mathbf{x} \sim U(1, \|\mathbf{x}\|)$

$\mathbf{V}$ Segmentation
$\mathbf{V}|n = [V_1, \ldots, V_i, \ldots, V_n]$

W_i Offset, starting position of segment i in $\mathbf{x}$
$W_i|\mathbf{x}, v_i \sim U(1, \|\mathbf{x}\| - v_i + 1)$

$\mathbf{W}$ Segment offsets
$\mathbf{W}|n = [W_1, \ldots, W_i, \ldots, W_n]$

$\mathbf{Y_i}$ Segment i
$\mathbf{Y_i}|\mathbf{x}, v_i, w_i = \mathbf{x}[w_i..(w_i + v_i - 1)]$

$\mathbf{Y}$ Sequence of characters generated from $\mathbf{x}$
$\mathbf{Y}|\mathbf{y_1}..\mathbf{y_n}$ = Concatenation of segments $\mathbf{y_1}..\mathbf{y_n}$

Figure 5.1: Segment Selection model

marginal probability of $\mathbf{x}$:

$$P(\mathbf{a} \in l|\mathbf{x} \in l) = \frac{P(\mathbf{Y} = \mathbf{a}|\mathbf{x})}{P(\mathbf{Y} = \mathbf{x}|\mathbf{x})} \tag{5.1}$$

We expect that this assumption (Equation 5.1) will be useful for distinguishing sequences in l from those not in l, because no assumption is made that $\mathbf{a}$ is in l, by not assuming a degree of generalization, as explained in Chapter 2.

Note that when $\mathbf{a} = \mathbf{x}$ it follows that $P(\mathbf{Y} = \mathbf{a}|\mathbf{x})/P(\mathbf{Y} = \mathbf{x}|\mathbf{x}) = 1$, which agrees with the fact that $\mathbf{x} \in l$ is given. However, when $\mathbf{a}$ is a subsequence of $\mathbf{x}$ it may be true that $P(\mathbf{a} \in l|\mathbf{x} \in l) < 1$ which would contradict a reasonable argument that $\mathbf{a} \in l$, because it is a subsequence of $\mathbf{x}$. Also, a probability must be equal to or less than one. We do not know of any cases where $P(\mathbf{Y} = \mathbf{a}|\mathbf{x}) > P(\mathbf{Y} = \mathbf{x}|\mathbf{x})$. We do know that if $\mathbf{x}$ = *"b"* and $\mathbf{a}$ is a sequence of more than one *"b"*, then $P(\mathbf{Y} = \mathbf{a}|\mathbf{x}) = P(\mathbf{Y} = \mathbf{x}|\mathbf{x})$, which can be shown by summing the joint probability $P(\mathbf{v}, \mathbf{y}|\mathbf{x})$ (Equation 5.18) over all possible segmentations $\mathbf{v}$.

Equation 5.1 does not assume that the marginal represents the true distribution of sequences in l, rather it assumes that the marginal represents the weight of evidence supporting the conclusion that $\mathbf{a} \in l$ given that all that is known about l is that $\mathbf{x} \in l$.

5.1.2 Generative Process

The model shows a generative process that produces a sequence of characters $\mathbf{y}$, given $\mathbf{x}$. $\mathbf{y}$ is generated by concatenating segments $\mathbf{y_1}..\mathbf{y_n}$. Each segment $\mathbf{y_i}$ is independently drawn from $\mathbf{x}$ using length v_i and offset w_i.

The number of segments n is drawn uniformly from 1 to c, where c is an arbitrarily large constant. This constant is used to define a finite universe of discourse. The maximum n which could possibly result in producing a $\mathbf{y}$ which equals $\mathbf{a}$ is $\|\mathbf{a}\|$. It is assumed that all sequences $\mathbf{a}$, that will ever be observed, will satisfy the condition $\|\mathbf{a}\| < c$. Assuming this condition implies that it is possible to draw any n which could possibly result in $\mathbf{y} = \mathbf{a}$. As long as this condition is met, the choice of c is not important, because it does not affect the marginal probability of $\mathbf{a}$ relative to the marginal probability of any other $\mathbf{a}$.

Length v_i is drawn uniformly from 1 to $\|\mathbf{x}\|$. The vector $\mathbf{v} = [v_1, \ldots, v_i, \ldots, v_n]$ contains all segment lengths and represents a segmentation of $\mathbf{y}$.

Offset w_i is drawn uniformly from all possible positions in $\mathbf{x}$, identifying the starting position of a segment of length v_i. The vector $\mathbf{w} = [w_1, \ldots, w_i, \ldots, w_n]$ contains all segment offsets. Length v_i is generated first, then offset w_i.

An example of Segment Selection is shown in Figure 5.2. Segments of length 7, 1, 10, and 5 are drawn from $\mathbf{x}$ to generate $\mathbf{y}$.

5.1.3 Degree of Generalization

Each value of $\mathbf{V}$ represents a degree of generalization. The value $\mathbf{v} = [\|\mathbf{x}\|]$ represents the lowest possible degree, no generalization. In this case, there is only one segment in $\mathbf{y}$, and that segment is $\mathbf{x}$. The value $\mathbf{v} = [1, 1, 1, \ldots]$, a vector of c ones, represents the highest possible degree of

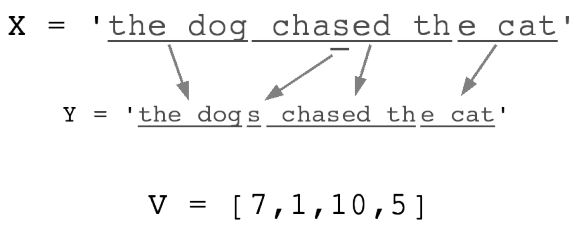


Figure 5.2: Example of Segment Selection

generalization. This is because there are a maximal number of independently drawn segments, each of which is only a single character in length.

The model avoids assuming a degree of generalization by avoiding preference for larger or smaller numbers of segments n, within the finite universe of discourse, and by avoiding preference for larger or smaller segment lengths v_i, which are possible to draw from $\mathbf{x}$.

The reason that w_i is drawn after v_i, is to ensure that v_i is always drawn uniformly from 1 to $\|\mathbf{x}\|$. If offset w_i were drawn first, then the range of possible v_i would be limited by the choice of w_i. For example, it would only be possible to draw segment length $\|\mathbf{x}\|$ when $w_i = 1$. Drawing w_i before v_i would have the effect of assuming a degree of generalization; shorter segments would be preferred.

Since the model assigns zero probability that $\mathbf{y} = \mathbf{a}$, if $\mathbf{a}$ contains a character which is not seen in $\mathbf{x}$, it is necessary to ensure that all characters which are found in $\mathbf{a}$ are also found in $\mathbf{x}$, in order to use the model to estimate the marginal. The way we meet this requirement is to encode both $\mathbf{x}$ and $\mathbf{a}$ in a sufficiently small character set. For example, there are thousands of words in the English language, but relatively few English characters. By encoding words as a sequence of characters, it is possible to represent a great variety of words using only a small set of characters.[1]

5.1.4 Comparison to Conventional Models

In a conventional language model, a degree of generalization is assumed which maximizes the probability that a held-out portion of training data is generated from the rest of training data.

[1]This principle can be applied to an extreme: by encoding words in binary, it is possible to represent any word using only two characters.

Typically, this degree of generalization is relatively high, since the held-out portion is usually very different from the other training data. Put in terms of Segment Selection, in order for the held-out portion to be generated from the rest of training data, a large number of short segments would typically be required.

Segment Selection differs from conventional language models in that a high degree of generalization is not preferred over a low degree of generalization, for the purpose of estimating the probability that a sequence belongs. Not preferring a high degree of generalization over a low degree of generalization means that sequences which can be generated with a low degree of generalization and which are likely to belong to the language of the training data, such as a copy of the training data itself, will not be assigned lower probability by the model simply because a low degree of generalization is not expected.

Some conventional language models interpolate between various orders of n-gram models [11]. Such models are somewhat similar to Segment Selection from the perspective of combining component models, each assuming a different degree of generalization. Conventional interpolation is used to fine-tune the degree of generalization assumed, rather than to avoid assuming a degree of generalization.

5.2 Estimating the Marginal

In this section, we outline our approach to estimating the marginal, $P(\mathbf{y}|\mathbf{x})$.

We apply the chain rule to factor the joint probability of the variables in the model given $\mathbf{x}$ as follows:

$$P(n, \mathbf{v}, \mathbf{w}, \mathbf{y}|\mathbf{x}) = P(n|\mathbf{x})P(\mathbf{v}|\mathbf{x}, n)P(\mathbf{w}|\mathbf{x}, n, \mathbf{v})P(\mathbf{y}|\mathbf{x}, n, \mathbf{v}, \mathbf{w}) \tag{5.2}$$

It is straightforward, but intractable, to calculate $P(\mathbf{y}|\mathbf{x})$ by summing $P(n, \mathbf{v}, \mathbf{w}, \mathbf{y}|\mathbf{x})$ over all possible assignments of N, $\mathbf{V}$ and $\mathbf{W}$. Rather, in the following sections we show how to efficiently estimate the marginal using Gibbs sampling. First, we marginalize out N and $\mathbf{W}$ to

obtain the joint distribution $P(\mathbf{v}, \mathbf{y}|\mathbf{x})$. Then we use Gibbs sampling to obtain samples from the posterior, $P(\mathbf{v}|\mathbf{y}, \mathbf{x})$. Finally, we use samples from the posterior to estimate the marginal $P(\mathbf{y}|\mathbf{x})$.

5.3 Collapsing the Model

Before applying Gibbs sampling, it is convenient to marginalize out the variables N and $\mathbf{W}$. This is done by summing $P(n, \mathbf{v}, \mathbf{w}, \mathbf{y}|\mathbf{x})$ (Equation 5.2) over all values of N and $\mathbf{W}$:

$$P(\mathbf{v}, \mathbf{y}|\mathbf{x}) = \sum_{n} \sum_{\mathbf{w}} P(n|\mathbf{x}) P(\mathbf{v}|\mathbf{x}, n) P(\mathbf{w}|\mathbf{x}, n, \mathbf{v}) P(\mathbf{y}|\mathbf{x}, n, \mathbf{v}, \mathbf{w}) \tag{5.3}$$

The last two terms in Equation 5.3 assume a given $\mathbf{v}$. If $\mathbf{v}$ is given then $n = \|\mathbf{v}\|$. We re-write these last two terms omitting the given n, since $\mathbf{v}$ is already given:

$$P(\mathbf{v}, \mathbf{y}|\mathbf{x}) = \sum_{n} \sum_{\mathbf{w}} P(n|\mathbf{x}) P(\mathbf{v}|\mathbf{x}, n) P(\mathbf{w}|\mathbf{x}, \mathbf{v}) P(\mathbf{y}|\mathbf{x}, \mathbf{v}, \mathbf{w}) \tag{5.4}$$

Re-arranging terms:

$$P(\mathbf{v}, \mathbf{y}|\mathbf{x}) = \left(\sum_{n} P(n|\mathbf{x}) P(\mathbf{v}|\mathbf{x}, n) \right) \left(\sum_{\mathbf{w}} P(\mathbf{w}|\mathbf{x}, \mathbf{v}) P(\mathbf{y}|\mathbf{x}, \mathbf{v}, \mathbf{w}) \right) \tag{5.5}$$

Which is equivalent to:

$$P(\mathbf{v}, \mathbf{y}|\mathbf{x}) = P(\mathbf{v}|\mathbf{x}) P(\mathbf{y}|\mathbf{x}, \mathbf{v}) \tag{5.6}$$

The joint $P(\mathbf{v}, \mathbf{y}|\mathbf{x})$ equals the prior $P(\mathbf{v}|\mathbf{x})$ times the likelihood $P(\mathbf{y}|\mathbf{x}, \mathbf{v})$. In the following three sections we derive an equation for each of these three probabilities: joint, prior, and likelihood. Note that all three are conditioned upon $\mathbf{x}$.

5.4 The Prior

The prior $P(\mathbf{v}|\mathbf{x})$ is found by marginalizing out N:

$$P(\mathbf{v}|\mathbf{x}) = \sum_{n=1}^{c} P(n|\mathbf{x})P(\mathbf{v}|\mathbf{x}, n) \tag{5.7}$$

Note that the distribution of N does not depend upon $\mathbf{x}$, so $P(n|\mathbf{x}) = P(n)$ (see Figure 5.1). Note also that $P(n) = \frac{1}{c}$. We refer to $\frac{1}{c}$ as ϵ, an arbitrarily small constant:

$$P(n|\mathbf{x}) = P(n) = \frac{1}{c} = \epsilon \tag{5.8}$$

The probability of drawing a segment length v_i given $\mathbf{x}$ is (see Figure 5.1):

$$P(v_i|\mathbf{x}) = \|\mathbf{x}\|^{-1} \tag{5.9}$$

Each segment length draw is conditionally independent of all others, given $\mathbf{x}$. The probability of drawing $\mathbf{v}$, given $\mathbf{x}$ and $\|\mathbf{v}\| = n$, is the product of the individual segment length draws shown in Equation 5.9. Thus, the probability of $\mathbf{v}$ given $\mathbf{x}$ and n is shown below:

$$P(\mathbf{v}|\mathbf{x}, n) = \begin{cases} \|\mathbf{x}\|^{-\|\mathbf{v}\|} & \text{if } \|\mathbf{v}\| = n \\ 0 & \text{otherwise} \end{cases} \tag{5.10}$$

Combining Equations 5.7 through 5.10 gives the prior $P(\mathbf{v}|\mathbf{x})$. Note that there is one non-zero term in the sum of Equation 5.7, the term for which $\|\mathbf{v}\| = n$, thus:

$$P(\mathbf{v}|\mathbf{x}) = \epsilon\|\mathbf{x}\|^{-\|\mathbf{v}\|} \tag{5.11}$$

5.5 The Likelihood

The likelihood $P(\mathbf{y}|\mathbf{x},\mathbf{v})$ is equal to the product of the individual segment draws $P(\mathbf{y_i}|\mathbf{x},v_i)$ $(1 \leq i \leq n)$, because each $\mathbf{y_i}$ is conditionally independent of all other $\mathbf{y_{j(j\neq i)}}$, given $\mathbf{x}$.)

$$P(\mathbf{y}|\mathbf{x},\mathbf{v}) = \prod_{i=1}^{\|\mathbf{v}\|} P(\mathbf{y_i}|\mathbf{x},v_i) \tag{5.12}$$

The probability of drawing a segment $\mathbf{y_i}$, given $\mathbf{x}$ and v_i, is calculated by marginalizing out w_i:

$$P(\mathbf{y_i}|\mathbf{x},v_i) = \sum_{w_i=1}^{\|\mathbf{x}\|} P(w_i|\mathbf{x},v_i)P(\mathbf{y_i}|\mathbf{x},v_i,w_i) \tag{5.13}$$

The probability of offset w_i, given $\mathbf{x}$ and v_i, is uniform over all positions in $\mathbf{x}$ identifying segments of length v_i:

$$P(w_i|\mathbf{x},v_i) = \frac{1}{\|\mathbf{x}\| - v_i + 1} \tag{5.14}$$

The probability of $\mathbf{y_i}$, given $\mathbf{x}$, v_i, and w_i, is one or zero depending on whether the segment in $\mathbf{x}$ beginning at offset w_i of length v_i equals $\mathbf{y_i}$ or not:

$$P(\mathbf{y_i}|\mathbf{x},v_i,w_i) = \begin{cases} 1 & \text{if } \mathbf{y_i} = \mathbf{x}[w_i..(w_i+v_i-1)] \\ 0 & \text{otherwise} \end{cases} \tag{5.15}$$

The probability of drawing a segment $\mathbf{y_i}$, given $\mathbf{x}$ and v_i, is the sum of Equation 5.15 times Equation 5.14 over all w_i (see Equation 5.13), which is the number of times that particular segment occurs in $\mathbf{x}$, divided by the total number of possible segments of length v_i in $\mathbf{x}$:

$$P(\mathbf{y_i}|\mathbf{x},v_i) = \frac{\|\{w_i|\mathbf{y_i} = \mathbf{x}[w_i..(w_i+v_i-1)]\}\|}{\|\mathbf{x}\| - v_i + 1} \tag{5.16}$$

Combining Equations 5.12 and 5.16 gives the likelihood $P(\mathbf{y}|\mathbf{x}, \mathbf{v})$:

$$P(\mathbf{y}|\mathbf{x}, \mathbf{v}) = \prod_{i=1}^{\|\mathbf{v}\|} \frac{\|\{w_i|\mathbf{y_i} = \mathbf{x}[w_i..(w_i + v_i - 1)]\}\|}{\|\mathbf{x}\| - v_i + 1} \tag{5.17}$$

5.6 The Joint

Equation 5.6 shows that the joint $P(\mathbf{v}, \mathbf{y}|\mathbf{x})$ is the prior $P(\mathbf{v}|\mathbf{x})$ (Equation 5.11) times the likelihood $P(\mathbf{y}|\mathbf{x}, \mathbf{v})$ (Equation 5.17):

$$P(\mathbf{v}, \mathbf{y}|\mathbf{x}) = \epsilon\|\mathbf{x}\|^{-\|\mathbf{v}\|} \prod_{i=1}^{\|\mathbf{v}\|} \frac{\|\{w_i|\mathbf{y_i} = \mathbf{x}[w_i..(w_i + v_i - 1)]\}\|}{\|\mathbf{x}\| - v_i + 1} \tag{5.18}$$

We use Equation 5.18 in the next section to derive the complete conditional for Gibbs sampling distributions.

5.7 Gibbs Sampling

Gibbs sampling [7] [14] is applied in order to obtain samples from the posterior $P(\mathbf{v}|\mathbf{x}, \mathbf{y})$. In the next section, we show how to use these samples to estimate the marginal $P(\mathbf{y}|\mathbf{x})$.

Rather than sampling $\mathbf{V}$ directly, it is convenient to sample a vector $\mathbf{H}$ of binary variables H_k corresponding to each point k between characters of $\mathbf{y}$. The total number K of these variables is equal to $\|\mathbf{y}\| - 1$.

$$\mathbf{H} = [H_1, \cdots, H_k, \cdots, H_K] \in [0, 1]^K \tag{5.19}$$

Each variable H_k is one if point k is at a boundary between segments and zero otherwise:

$$H_k = \begin{cases} 1 & \text{if } k \text{ is a segment boundary} \\ 0 & \text{otherwise} \end{cases} \tag{5.20}$$

There is a one-to-one correspondence between $\mathbf{H}$ and $\mathbf{V}$. Vectors $\mathbf{h}$ and $\mathbf{v}$ are equivalent segmentations of $\mathbf{y}$. The notation $\mathbf{y}(l_i, r_i)$ refers to the segment $\mathbf{y_i}$ which has its left boundary at l_i and its right boundary at r_i, where $h_{l_i} = 1$, $h_{r_i} = 1$, and $h_k = 0$ for all $l_i < k < r_i$:

$$\mathbf{y}(l_i, r_i) = \mathbf{y_i} \tag{5.21}$$

Segment length v_i written in terms of boundaries l_i and r_i is:

$$v_i = r_i - l_i \tag{5.22}$$

The notation $\psi(l_i, r_i)$ refers to the probability $P(\mathbf{y_i}|\mathbf{x}, v_i)$ (see Equation 5.16) in terms of boundaries l_i and r_i:

$$\psi(l_i, r_i) = \frac{\|\{w_i|\mathbf{y}(l_i, r_i) = \mathbf{x}[w_i..(w_i + (r_i - l_i) - 1)]\}\|}{\|\mathbf{x}\| - (r_i - l_i) + 1} \tag{5.23}$$

The number of segments $\|\mathbf{v}\|$ is equal to the number of boundaries in $\mathbf{h}$ plus one, which we refer to as $n_{\mathbf{h}}$, the number of segments given $\mathbf{h}$:

$$\|\mathbf{v}\| = \|\{h_k|(1 \leq k \leq K) \wedge (h_k = 1)\}\| + 1 = n_{\mathbf{h}} \tag{5.24}$$

The joint probability $P(\mathbf{v}, \mathbf{y}|\mathbf{x})$ (see Equation 5.18) is written in terms of $\psi(l_i, r_i)$ and $n_{\mathbf{h}}$ as follows:

$$P(\mathbf{v}, \mathbf{y}|\mathbf{x}) = P(\mathbf{h}, \mathbf{y}|\mathbf{x}) = \epsilon \prod_{i=1}^{n_{\mathbf{h}}} \frac{\psi(l_i, r_i)}{\|\mathbf{x}\|} \tag{5.25}$$

Samples from the posterior are obtained by sampling H_k given $\mathbf{x}$, $\mathbf{y}$, and values for all other variables in $\mathbf{H}$, which distribution is known as the complete conditional. First we derive the complete conditional for sampling one variable H_k at a time. Later we specify the complete conditional for sampling two variables H_k and H_{k+1} at a time.

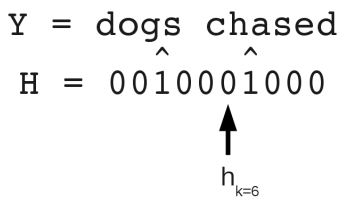


Figure 5.3: Sampling H_6.

During sampling, only local segments are affected. A segment is considered local if $l_i \leq k \leq r_i$. Sampling $H_k = 0$ results in a single local segment, $\mathbf{y}(l_i, r_i)$. Sampling $H_k = 1$ results in two local segments, one to the left of k, denoted as $\mathbf{y}(l_i, k)$ and one to the right, $\mathbf{y}(k, r_{i+1})$. The number of segments n can change, depending on the value of H_k. Segment index i refers to an arbitrary segment; $i + 1$ refers to the subsequent segment.

An example is shown in Figure 5.3: $\mathbf{y}$ is the string *"dogs chased"*; $\mathbf{h}$ is the previous segmentation, resulting in three segments, *"dog"*, *"s ch"*, and *"ased"*. The sampled point is H_6, at $k = 6$ between *"c"* and *"h"*. There is only one local segment, the second segment; it contains the sampled point. If $H_k = 0$, then there is no change to the previous segmentation. If $H_k = 1$, then the middle segment *"s ch"* is split into two local segments, *"s c"* and *"h"*. The number of segments is therefore incremented from three to four. The first and last segments are unchanged, but the index i of the last segment is also incremented from three to four. When H_7 is sampled, its value may change from one to zero. If so, then the two local segments on either side of $k = 7$ will merge into a single segment, reducing the number of segments by one.

The complete conditional is derived as follows: (Variable $\mathbf{H}_{\setminus k}$ refers to all variables in $\mathbf{H}$ except H_k.)

$$P(h_k|\mathbf{h}_{\setminus k}, \mathbf{y}, \mathbf{x}) = \frac{P(h_k, \mathbf{h}_{\setminus k}, \mathbf{y}|\mathbf{x})}{P(\mathbf{h}_{\setminus k}, \mathbf{y}|\mathbf{x})} \tag{5.26}$$

The denominator in the above equation is calculated by marginalizing out H_k:

$$P(\mathbf{h}_{\setminus k}, \mathbf{y}|\mathbf{x}) = P(h_k = 0, \mathbf{h}_{\setminus k}, \mathbf{y}|\mathbf{x}) + P(h_k = 1, \mathbf{h}_{\setminus k}, \mathbf{y}|\mathbf{x}) \tag{5.27}$$

Combining Equations 5.26 and 5.27 gives:

$$P(h_k|\mathbf{h}_{\setminus k},\mathbf{y},\mathbf{x}) = \frac{P(h_k,\mathbf{h}_{\setminus k},\mathbf{y}|\mathbf{x})}{P(h_k=0,\mathbf{h}_{\setminus k},\mathbf{y}|\mathbf{x}) + P(h_k=1,\mathbf{h}_{\setminus k},\mathbf{y}|\mathbf{x})} \tag{5.28}$$

The joint probability (Equation 5.25) of option $h_k = 0$ is epsilon times terms relating to all segments prior to the local segment, times terms relating to the local segment, times terms relating to all subsequent segments:

$$P(h_k=0,\mathbf{h}_{\setminus k},\mathbf{y}|\mathbf{x}) = \epsilon \left(\prod_{n'=1}^{n-1} \frac{\psi(l_{n'},r_{n'})}{\|\mathbf{x}\|}\right)\left(\frac{\psi(l_i,r_i)}{\|\mathbf{x}\|}\right)\left(\prod_{n''=n+1}^{n_\mathbf{h}} \frac{\psi(l_{n''},r_{n''})}{\|\mathbf{x}\|}\right) \tag{5.29}$$

The joint probability of option $h_k = 1$ is epsilon times terms relating to all segments prior to local segments, times terms relating to local segments, times terms relating to all subsequent segments:

$$P(h_k=1,\mathbf{h}_{\setminus k},\mathbf{y}|\mathbf{x}) = \epsilon \left(\prod_{n'=1}^{n-1} \frac{\psi(l_{n'},r_{n'})}{\|\mathbf{x}\|}\right)\left(\frac{\psi(l_i,k)}{\|\mathbf{x}\|}\right)\left(\frac{\psi(k,r_{i+1})}{\|\mathbf{x}\|}\right)\left(\prod_{n''=n+2}^{n_\mathbf{h}} \frac{\psi(l_{n''},r_{n''})}{\|\mathbf{x}\|}\right) \tag{5.30}$$

The probability of $h_k = 0$ given $\mathbf{h}_{\setminus k}$, $\mathbf{y}$, and $\mathbf{x}$ is found by combining Equations 5.28, 5.29, and 5.30: (Terms relating to non-local segments cancel.)

$$P(h_k=0|\mathbf{h}_{\setminus k},\mathbf{y},\mathbf{x}) = \frac{\epsilon\frac{\psi(l_i,r_i)}{\|\mathbf{x}\|}}{\epsilon\frac{\psi(l_i,r_i)}{\|\mathbf{x}\|} + \epsilon\frac{\psi(l_i,k)}{\|\mathbf{x}\|}\cdot\frac{\psi(k,r_{i+1})}{\|\mathbf{x}\|}} \tag{5.31}$$

Multiplying numerator and denominator by $\frac{\|\mathbf{x}\|^2}{\epsilon}$ gives:

$$P(h_k=0|\mathbf{h}_{\setminus k},\mathbf{y},\mathbf{x}) = \frac{\|\mathbf{x}\|\psi(l_i,r_i)}{\|\mathbf{x}\|\psi(l_i,r_i) + \psi(l_i,k)\psi(k,r_{i+1})} \tag{5.32}$$

Similarly, the probability of $h_k = 1$ given $\mathbf{h}_{\setminus k}$, $\mathbf{y}$, and $\mathbf{x}$ is:

$$P(h_k=1|\mathbf{h}_{\setminus k},\mathbf{y},\mathbf{x}) = \frac{\psi(l_i,k)\psi(k,r_{i+1})}{\|\mathbf{x}\|\psi(l_i,r_i) + \psi(l_i,k)\psi(k,r_{i+1})} \tag{5.33}$$

Combining Equations 5.32 and 5.33 yields the complete conditional for sampling H_k:

$$P(h_k|\mathbf{h}_{\setminus k}, \mathbf{y}, \mathbf{x}) = \begin{cases} \frac{\|\mathbf{x}\|}{Z}\psi(l_i, r_i) & \text{if } h_k = 0 \\ \frac{1}{Z}\psi(l_i, k)\psi(k, r_{i+1}) & \text{if } h_k = 1 \end{cases} \tag{5.34}$$

Where:

$$Z = \|\mathbf{x}\|\psi(l_i, r_i) + \psi(l_i, k)\psi(k, r_{i+1}) \tag{5.35}$$

Equations 5.34 and 5.35 specify the distribution needed for Gibbs sampling, when sampling one variable at a time. It is the distribution over all possible values of a subset of latent variables (namely H_k), given values for all other latent variables (namely $\mathbf{H}_{\setminus k}$). It is shown in [7] and [14] that repeatedly sampling all latent variables in this manner results in a Markov chain of samples which converges in distribution to the true posterior distribution over latent variables. Convergence in distribution is guaranteed given a sufficiently large number of samples. In order to get good approximations in a short period of time it is necessary for the sampler to be efficient. To improve efficiency we sample more than one variable at a time, as described next.

5.7.1 Block Sampling

An inefficient sampler is one which gets stuck for long periods of time in one mode of a multi-modal distribution. Sampling only one point H_k at a time may result in an inefficient sampler. For example, suppose that k represents a segment boundary. Suppose further that sampling $H_k = 0$ is not possible because segment $\mathbf{y}(l_i, r_i)$ does not exist in $\mathbf{x}$. Suppose also that the joint probability of a boundary at k is high, and that moving that boundary from k to $k + 1$ also results in high joint probability. Suppose also that a boundary at k and $k + 1$ results in low joint probability. If this is the case, then it is likely that the sampler gets stuck where the boundary is at k or $k + 1$, and that the boundary does not move freely between the two high probability segmentations.

In order to allow segment boundaries to move more easily, block sampling is employed: two adjacent points are sampled at the same time, both H_k and H_{k+1}. Sampling begins by sampling

H_1 together with H_2 and then by sampling H_2 together with H_3 and so on until sampling H_{K-1} together with H_K:

$$((H_1, H_2), (H_2, H_3), \cdots, (H_k, H_{k+1}), \cdots, (H_{K-1}, H_K)) \tag{5.36}$$

Sampling (1,0) results in two local segments, one to the left of k, denoted as $\mathbf{y}(l_i, k)$ and one to the right, $\mathbf{y}(k, r_{i+1})$. Sampling (0,1) also results in two local segments, one to the left of $k+1$, denoted as $\mathbf{y}(l_i, k+1)$ and one to the right, $\mathbf{y}(k+1, r_{i+1})$. Sampling (1,1) results in three local segments, $\mathbf{y}(l_i, k)$, $\mathbf{y}(k, k+1)$, and $\mathbf{y}(k+1, r_{i+2})$. Sampling (0,0) results in a single local segment, $\mathbf{y}(l_i, r_i)$. The complete conditional for the block sampler is shown below:

$$P(h_k, h_{k+1} | h_1..h_{k-1}, h_{k+2}..h_K, \mathbf{y}, \mathbf{x}) = \begin{cases} \frac{1}{Z}\psi(l_i, k)\psi(k, r_{i+1}) & \text{if } h_k = 1 \text{ and } h_{k+1} = 0 \\ \frac{1}{Z}\psi(l_i, k+1)\psi(k+1, r_{i+1}) & \text{if } h_k = 0 \text{ and } h_{k+1} = 1 \\ \frac{1}{Z\|\mathbf{x}\|}\psi(l_i, k)\psi(k, k+1)\psi(k+1, r_{i+2}) & \text{if } h_k = 1 \text{ and } h_{k+1} = 1 \\ \frac{\|\mathbf{x}\|}{Z}\psi(l_i, r_i) & \text{if } h_k = 0 \text{ and } h_{k+1} = 0 \end{cases} \tag{5.37}$$

Where:

$$\begin{aligned} Z = {} & \psi(l_i, k)\psi(k, r_{i+1}) + \\ & \psi(l_i, k+1)\psi(k+1, r_{i+1}) + \\ & \|\mathbf{x}\|^{-1}\psi(l_i, k)\psi(k, k+1)\psi(k+1, r_{i+2}) + \\ & \|\mathbf{x}\|\psi(l_i, r_i) \end{aligned} \tag{5.38}$$

5.7.2 Transition Operators

In Section 5.8 we describe how we use Gibbs sampling to estimate the marginal. To facilitate that discussion, we first define a transition operator which is a process for sampling all pairs of adjacent points in $\mathbf{H}$. Equations 5.37 and 5.38 describe how each pair of adjacent points (H_k, H_{k+1})

is sampled. We define a transition operator $T(\mathbf{v}^{'} \leftarrow \mathbf{v})$ for a transition from one segmentation $\mathbf{v}$ to a possibly different segmentation $\mathbf{v}^{'}$ by sampling all pairs of points, from first to last as shown in List 5.36. We also define a reverse transition operator $\tilde{T}(\mathbf{v} \leftarrow \mathbf{v}^{'})$ for transitioning from one segmentation to its predecessor by sampling all pairs of points in reverse order. Both T and $\tilde{T}$ are needed for the method we employ to estimate the marginal, described in the following section.

The transition shown below is from $\mathbf{v}$ to $\mathbf{v}^{'}$. Segmentation $\mathbf{H}^{'} = [h_1^{'}, \cdots, h_k^{'}, \cdots, h_K^{'}]$ is an alternate but equivalent form of $\mathbf{v}^{'}$. The first step is to calculate the probability that h_1 transitions to $h_1^{'}$. This is done by marginalizing out the choice of h_2, summing over both possibilities $h_2 = 0$ and $h_2 = 1$. Then, the probability that each subsequent h_k transitions to $h_k^{'}$ is calculated in a similar manner, given that prior variables $h_1..h_{k-1}$ have previously transitioned to $h_1^{'}..h_{k-1}^{'}$, and that subsequent variables $h_{k+2}..h_K$ have yet to transition:

$$P(h_k^{'}|h_1^{'}..h_{k-1}^{'}, h_{k+2}..h_K, \mathbf{y}, \mathbf{x}) = \sum_{b=0}^{1} P(h_k^{'}, h_{k+1} = b|h_1^{'}..h_{k-1}^{'}, h_{k+2}..h_K, \mathbf{y}, \mathbf{x}) \tag{5.39}$$

The final step is to calculate the probability that the last two variables, h_{K-1} and h_K, transition to $h_{K-1}^{'}$ and $h_K^{'}$. The probability of a transition is the product of the probabilities of all steps within the transition:

$$T(\mathbf{v}^{'} \leftarrow \mathbf{v}) = \left[\prod_{k=1}^{K-2} P(h_k^{'}|h_1^{'}..h_{k-1}^{'}, h_{k+2}..h_K, \mathbf{y}, \mathbf{x})\right] P(h_{K-1}^{'}, h_K^{'}|h_1^{'}..h_{K-2}^{'}, \mathbf{y}, \mathbf{x}) \tag{5.40}$$

If $\mathbf{v}^{'}$ is known, then T evaluates to the probability of transitioning from $\mathbf{v}$ to $\mathbf{v}^{'}$. If $\mathbf{v}^{'}$ is not known, then T represents a process for drawing $\mathbf{v}^{'}$, given $\mathbf{v}$. $\mathbf{v}^{'}$ is drawn one $h_k^{'}$ at a time according to the distribution over each $h_k^{'}$ shown above.

$\tilde{T}$ is a reverse transition operator obtained by reversing the sampling steps in T:

$$P(h_{k+1}|h_1^{'}..h_{k-1}^{'}, h_{k+2}..h_K, \mathbf{y}, \mathbf{x}) = \sum_{b=0}^{1} P(h_k^{'} = b, h_{k+1}|h_1^{'}..h_{k-1}^{'}, h_{k+2}..h_K, \mathbf{y}, \mathbf{x}) \tag{5.41}$$

$$\tilde{T}(\mathbf{v} \leftarrow \mathbf{v}^{'}) = \left[\prod_{k=2}^{K-1} P(h_{k+1}|h_1^{'}..h_{k-1}^{'}, h_{k+2}..h_K, \mathbf{y}, \mathbf{x})\right] P(h_1, h_2|h_3..h_K, \mathbf{y}, \mathbf{x}) \tag{5.42}$$

5.7.3 Efficiency

Sampling involves calculating $\psi(l_i, r_i)$ which requires segment counts from $\mathbf{x}$ (Equation 5.23). To calculate and store segment counts, we use a suffix tree, which can be constructed on-line in $O(\|\mathbf{x}\|)$ time and space [18]. A path from root to a leaf specifies one or more unique segments, occurring only once in $\mathbf{x}$. A path from root to an interior node specifies one or more segments which have the same number of occurrences in $\mathbf{x}$ as the number of leaves that descend from the interior node. Rather than repeatedly calculating the number of leaves descending from interior nodes, this is done once, and the results are stored in the nodes of the tree. Counting is done using depth-first search. Before exiting each node, a count is calculated as the sum of counts of all child nodes. The time and space required to calculate and store segment counts is also linear in the length of $\mathbf{x}$. The time required to retrieve an individual segment count from a tree node is linear in the length of the segment.

5.8 Chib-style Estimation

We can see from Bayes law that the marginal is equal to the joint divided by the posterior:

$$P(\mathbf{y}|\mathbf{x}) = \frac{P(\mathbf{v}|\mathbf{x})P(\mathbf{y}|\mathbf{x}, \mathbf{v})}{P(\mathbf{v}|\mathbf{x}, \mathbf{y})} \tag{5.43}$$

The equation above can be used to estimate the marginal. The joint can be estimated in closed form. Gibbs sampling can be used to estimate the posterior. Chib [4] describe such a method to estimate a marginal from the output of a Gibbs sampler in a high dimensional latent space. We use the estimator of Murray and Salakhutdinov [13], which builds on Chib's work. The estimator

$\mathbf{v}^*$ A segmentation of preferably high posterior probability
S Number of Markov chain transitions
T Markov chain transition operator

Draw $s \sim U(1, S)$
Draw $\mathbf{v}^{(s)} \sim \tilde{T}(\mathbf{v}^{(s)} \leftarrow \mathbf{v}^*)$
For $s' = (s+1)$ to r
    Draw $\mathbf{v}^{(s')} \sim T(\mathbf{v}^{(s')} \leftarrow \mathbf{v}^{(s'-1)})$
For $s' = (s-1)$ to 1 increment by -1
    Draw $\mathbf{v}^{(s')} \sim \tilde{T}(\mathbf{v}^{(s')} \leftarrow \mathbf{v}^{(s'+1)})$

Figure 5.4: Markov chain procedure

is formally unbiased and overcomes the problem of overestimating in expectation, even when the number of samples is finite.

The estimator requires a special segmentation $\mathbf{v}^*$ of preferably high posterior probability. We find this special segmentation using MINS, described in the following chapter, which finds a segmentation consisting of a minimal number of segments. We use the MINS segmentation to initialize the Gibbs sampler. We then use the first sampled segmentation as the special segmentation $\mathbf{v}^*$.

The estimator also requires a carefully constructed Markov chain procedure, shown in Figure 5.4. S is the total number of steps in a Markov chain of segmentations $\mathbf{v}^{(s)}$ $1 \leq s \leq S$. A starting point, s, is drawn uniformly from 1 to S. Segmentation $\mathbf{v}^{(s)}$ is drawn from the special segmentation $\mathbf{v}^*$ using the reverse transition operator $\tilde{T}$ (Equation 5.42). The chain of segmentations is constructed from $\mathbf{v}^{(s)}$ in both directions, forwards and backwards, using transition operators T (Equation 5.40) and $\tilde{T}$.

The marginal $P(\mathbf{y}|\mathbf{x})$ is estimated as the joint probability of the special segmentation divided by the mean of transition probabilities from each segmentation in the Markov chain to the special segmentation $\mathbf{v}^*$:

$$P(\mathbf{y}|\mathbf{x}) \approx \frac{P(\mathbf{v}^*)P(\mathbf{y}|\mathbf{x}, \mathbf{v}^*)}{\frac{1}{S}\sum_{s'=1}^{S} T(\mathbf{v}^* \leftarrow \mathbf{v}^{(s')})} \tag{5.44}$$

Chapter 6

Experiments

We show that an estimate of the probability that a word sequence belongs to a chosen language (represented by a given sequence) can be constructed by avoiding an assumed degree of generalization.

We begin by demonstrating that perplexity, a standard metric for evaluating language models, although useful for evaluating a model's ability to estimate the probability that a sequence occurs, is not necessarily useful for evaluating a model's ability to estimate the probability that the sequence belongs.

Consequently, we evaluate a model's ability to estimate the probability that a sequence belongs by comparing the model's estimate of the mean log probability of a sequence known to belong with that of a sequence known not to belong.

We also demonstrate that *in some cases*:

- Modeling a sequence of words as a sequence of characters, avoids one type of assumption of a degree of generalization, and improves estimation of the probability that a sequence belongs.
- Both MINS and Segment Selection perform better at distinguishing sequences that belong from those that do not than any other method we tested.

6.1 Data

We define three sequences: AP Train, AP Test, and NYT Test, which we assume belong to our language of interest, which is comprehensible well-formed English. We also define RW, a sequence

of random words, which we assume does not belong to that language. Additionally, we define EW, a sequence of English words.

AP Train and AP Test are drawn from English Associated Press (AP) newswire, containing mostly political, business, and sports news. AP Test contains news stories written shortly after news stories of AP Train. XML tags and non-story documents are excluded. Characters are kept in mixed case, and digits are processed the same as other characters. The size of AP Train is 23 million words; the size of AP Test is 11 million words. A word is defined as a contiguous sequence of non-whitespace characters, which includes punctuation.

NYT is a sequence of 11 million words drawn from New York Times newswire, in the same way as described above for AP newswire.

RW is a sequence of 1.1 million random words, generated by concatenating 100 million characters, each character drawn uniformly from the set of characters seen in AP Test, including spaces.

EW is a sequence of 10 million English words, generated by drawing each word uniformly from the vocabulary of AP Test.

We choose RW to represent sequences that do not belong, rather than EW, because RW contains words which may not belong to the language, as opposed to EW which only contains words from AP Test.

6.1.1 Data Conversion

Experiments are performed to compare the models presented in this paper with a variety of other language models, described in Section 6.2. We configure some of these other models to convert characters to lower case, convert all digits to the digit 5 (could be any token denoting a digit), and convert unseen words (words not seen in training data) to a single unseen word token, following previous research [17] [19].

6.2 Language Models

We describe each language model used in our experiments to estimate the probability of a word sequence.

6.2.1 Good-Turing and Interpolated Modified Kneser-Ney

We use the SRI [16] implementation of Good-Turing (GT) and Interpolated Modified Kneser-Ney (IMKN). We configure both of these models to convert unseen words to a single unseen word token. In other words, they use an open vocabulary model.

Additionally, we perform experiments with variants of GT and IMKN called GTL and IMKNL. These variants are the same as GT and IMKN except that the variants convert all words to lower case and replace all digits with the digit 5, to explore the effects of these types of generalization. The "L" denotes lower-casing.

6.2.2 Sequence Memoizer

We use an implementation of the Sequence Memoizer (SM) [1] with default parameters. The only parameter we set is vocabulary size, which we set to ten times the size of the vocabulary of training data. Unlike GT and IMKN, SM does not convert unseen words to a single unseen word token. Rather, when an unseen word is encountered, the model assigns that word to an available slot in the fixed-size vocabulary. If all vocabulary slots are already filled, the model assigns zero probability to the word.

Additionally, we perform experiments with SMC, which uses the same SM implementation, but treats data as a sequence of characters, rather than as sequence of words. The "C" denotes the emphasis on characters. We set the size of the vocabulary to be the number of unique characters in training data.

[1] www.sequencememoizer.com

6.2.3 Segment Selection

We use our implemention of Segment Selection with two different values of S, the number of steps in the Markov chain of segmentations (see Section 5.8). The first model, SS, sets S equal to one. The second, SS10, sets S equal to ten. We expect that larger values of S will improve estimates of the marginal, however we are limited by practical considerations (see Section 6.6). It is reasonable to expect that a chain of length one is not useful, because of the lack of burn-in in the Markov chain, and yet the first segmentation in the chain is not a random segmentation, nor is it the only segmentation used to estimate the marginal. Section 5.8 explains how the marginal is estimated using segmentations in the Markov chain together with a special segmentation drawn using an initial MINS segmentation. For our purposes, a useful value of S is one which allows us to validate the thesis, which we show in this chapter can be done when S equals one.

6.2.4 MINS

We use our implementation of Minimal Number of Segments (MINS). The choice of negative constant for the MINS log-linear model is not important, since this constant does not affect the log probability of any sequence *relative* to any other sequence. We set this constant to -30.

6.2.5 Modified SS

It is possible to modify SS such that the model estimates a probability of occurring, rather than a probability of belonging, as does a conventional language model. We make such modifications, as described below, to create a family of baseline models which are very similar to SS, yet differ in that they assume a degree of generalization for the purpose of estimating a probability of occurring.

A conventional language modeling assumption for estimating the probability of a sequence is that the length of the sequence is given, which can be seen in the assumption that the sum of probabilities of all possible word sequences of a given length equals one. For example: the sum of probabilities of all possible word sequences of length one equals one, meaning that there is no

remaining probability for longer word sequences. (This assumption is sometimes ignored, as in the case when probabilities of word sequences of different lengths are directly compared.)

We modify SS by incorporating this conventional assumption, that the length of the sequence is given. When sequence length $\|\mathbf{y}\|$ is given, the probability is zero that the number of segments N is greater than $\|\mathbf{y}\|$.

We also modify SS to enable the model to incorporate the conventional assumption that the sequence belongs to the language of training data. When it is given that the sequence belongs, held-out data can be used to assume a degree of generalization, specifically to assume a non-uniform prior over N.

We assume that $\|\mathbf{y}\|$ is given and assume a non-uniform prior over N by modifying the Segment Selection model as follows:

N_λ Intermediate variable

$$N_\lambda|\lambda \sim Poisson(\lambda)$$

N Number of segments

$$N|n_\lambda, \|\mathbf{y}\| \sim \begin{cases} n_\lambda + 1 & \text{if } n_\lambda \leq \|\mathbf{y}\| - 1 \\ U(1, \|\mathbf{y}\|) & \text{otherwise} \end{cases}$$

According to the modified model, for $1 \leq n \leq \|\mathbf{y}\|$:

$$P(n|\lambda, \|\mathbf{y}\|) = \frac{\lambda^{n-1}}{(n-1)!}e^{-\lambda} + \frac{1 - e^{-\lambda}\sum_{i=0}^{\|\mathbf{y}\|-1}\frac{\lambda^i}{i!}}{\|\mathbf{y}\|} \tag{6.1}$$

Otherwise $P(n|\lambda, \|\mathbf{y}\|) = 0$.

The first term in Equation 6.1 is the value of the Poisson distribution at $n - 1$. The second term is used to uniformly distribute probability mass (from one to $\|\mathbf{y}\|$) which is in the tail of the Poisson distribution at points equal to or greater than $\|\mathbf{y}\|$.

Since the sum of segment lengths must equal the length of $\mathbf{y}$, there are $\|\mathbf{y}\| - 1$ choose $\|\mathbf{v}\| - 1$ different possible segmentations when $\|\mathbf{v}\| = n$, and the prior (see Equation 5.11) becomes

Equation 6.1 divided by the number of possible segmentations. For $1 \leq \|\mathbf{v}\| \leq \|\mathbf{y}\|$:

$$P(\mathbf{v}|\lambda, \|\mathbf{y}\|) = \left(\frac{\lambda^{\|\mathbf{v}\|-1}}{(\|\mathbf{v}\|-1)!} e^{-\lambda} + \frac{1 - e^{-\lambda} \sum_{i=0}^{\|\mathbf{y}\|-1} \frac{\lambda^i}{i!}}{\|\mathbf{y}\|} \right) \binom{\|\mathbf{y}\|-1}{\|\mathbf{v}\|-1}^{-1} \tag{6.2}$$

The joint (see Equation 5.25) and the complete conditional (see Equations 5.37 and 5.38) are derived as before, using the prior shown above. In terms of $\mathbf{h}$ ($\mathbf{h}$ and $\mathbf{v}$ are equivalent segmentations of y, and $n_{\mathbf{h}}$ is the number of segments given $\mathbf{h}$) we write the prior shown above as a function $\pi(n_{\mathbf{h}}, \lambda, \|\mathbf{y}\|)$. For $1 \leq n_{\mathbf{h}} \leq \|\mathbf{y}\|$:

$$\pi(n_{\mathbf{h}}, \lambda, \|\mathbf{y}\|) = \left(\frac{\lambda^{n_{\mathbf{h}}-1}}{(n_{\mathbf{h}}-1)!} e^{-\lambda} + \frac{1 - e^{-\lambda} \sum_{i=0}^{\|\mathbf{y}\|-1} \frac{\lambda^i}{i!}}{\|\mathbf{y}\|} \right) \binom{\|\mathbf{y}\|-1}{n_{\mathbf{h}}-1}^{-1} \tag{6.3}$$

Using this function to denote the prior, the joint for modified SS is:

$$P(\mathbf{h}, \mathbf{y}|\mathbf{x}, \lambda, \|\mathbf{y}\|) = \pi(n_{\mathbf{h}}, \lambda, \|\mathbf{y}\|) \prod_{i=1}^{n_{\mathbf{h}}} \psi(l_i, r_i) \tag{6.4}$$

To denote the complete conditional we add subscripts to $n_{\mathbf{h}}$, identifying the values of h_k and h_{k+1}. For example, if $h_k = 1$ and $h_{k+1} = 0$, then we denote $n_{\mathbf{h}}$ as $n_{\mathbf{h}(\mathbf{10})}$. Using this notation, the complete conditional for modified SS is:

$$P(h_k, h_{k+1}|h_1..h_{k-1}, h_{k+2}..h_K, \mathbf{y}, \mathbf{x}, \lambda) =$$
$$\begin{cases} \frac{1}{Z}\pi(n_{\mathbf{h}(10)}, \lambda, \|\mathbf{y}\|)\psi(l_i, k)\psi(k, r_{i+1}) & \text{if } h_k = 1 \text{ and } h_{k+1} = 0 \\ \frac{1}{Z}\pi(n_{\mathbf{h}(01)}, \lambda, \|\mathbf{y}\|)\psi(l_i, k+1)\psi(k+1, r_{i+1}) & \text{if } h_k = 0 \text{ and } h_{k+1} = 1 \\ \frac{1}{Z}\pi(n_{\mathbf{h}(11)}, \lambda, \|\mathbf{y}\|)\psi(l_i, k)\psi(k, k+1)\psi(k+1, r_{i+2}) & \text{if } h_k = 1 \text{ and } h_{k+1} = 1 \\ \frac{1}{Z}\pi(n_{\mathbf{h}(00)}, \lambda, \|\mathbf{y}\|)\psi(l_i, r_i) & \text{if } h_k = 0 \text{ and } h_{k+1} = 0 \end{cases} \tag{6.5}$$

Where:

$$
\begin{aligned}
Z = {} & \pi(n_{\mathbf{h}(10)}, \lambda, \|\mathbf{y}\|)\psi(l_i, k)\psi(k, r_{i+1}) + \\
& \pi(n_{\mathbf{h}(01)}, \lambda, \|\mathbf{y}\|)\psi(l_i, k+1)\psi(k+1, r_{i+1}) + \\
& \pi(n_{\mathbf{h}(11)}, \lambda, \|\mathbf{y}\|)\psi(l_i, k)\psi(k, k+1)\psi(k+1, r_{i+2}) + \\
& \pi(n_{\mathbf{h}(00)}, \lambda, \|\mathbf{y}\|)\psi(l_i, r_i)
\end{aligned} \tag{6.6}
$$

Samples obtained from the complete conditional above are used to estimate the probability $P(\mathbf{y}|\mathbf{x}, \lambda, \|\mathbf{y}\|)$ following the procedure described in Section 5.8, which probability is used directly to estimate the probability that a sequence occurs, meaning that Equation 5.1 is not used.

We set the mean λ of the Poisson to various values in order to study the effects of assuming a degree of generalization in the context of SS. Each modified SS model — L1K, L1M, L2M, L4M, L8M, L16M — is named after the value of λ assumed by the model. For example, L8M is a modified SS model for which λ equals eight million, meaning that when applied to AP Test (11 million words) the model expects a relatively large number of relatively short segments (a high degree of generalization).

6.2.6 COUNT

COUNT is a simple baseline model which counts the number of unseen words in a sequence. The log probability reported by the model is the number of unseen words times an arbitrary negative constant, which we set to -400. (The choice of negative constant is not important since this constant does not affect the log probability of any sequence *relative* to any other sequence.)

6.2.7 ALL

ALL is an even simpler baseline model, which converts all words to a single token, which token is always predicted. The log probability reported by the model is always zero.

6.2.8 Words vs Characters

GT, IMKN, and SM all model word sequences directly, in contrast to SS, modified SS, and SMC which model word sequences by modeling sequences of characters. Comparing these two types of models is intended to demonstrate the effects of assuming a degree of generalization by assuming probability for unseen words.

6.3 Perplexity

Perplexity is a standard metric for evaluating language models [11]. It is a measure of how well a model estimates probability of occurring, but we show in this section that it is not always suitable as a measure of how well a model estimates probability of belonging. Perplexity of word sequence S is calculated by exponentiating the negative mean log probability of sequence S according to model m:

$$perplexity(S, m) = \exp\left(\frac{-\log P_m(S)}{\|S\|}\right) \tag{6.7}$$

For all models, even those that model word sequences by modeling sequences of characters, mean log probability is calculated by dividing log probability according to the model by the number of words in S, not by the number of characters in S.

Table 6.1 shows perplexity of AP Test according to each model, where each model is trained on AP Train. Low perplexity corresponds to better performance.

Because models GT, GTL, IMKN, IMKNL, and ALL each convert unseen words to a single unseen word token, the probabilities assigned by these models do not sum to one over all possible word sequences of the same length, as is necessary to estimate a true probability distribution over word sequences for the purpose of estimating the probability that any sequence of a given length occurs. For example according to ALL, the sum of probabilities of any two sequences equals two.

Because models MINS, SS, and S10 each estimate a probability that a word sequence belongs, these models do not estimate a probability distribution over all possible word sequences,

Model	Perplexity	Model	Perplexity
GT	718	L1K	455,688
GTL	566	L1M	8,010
IMKN	477	L2M	5,806
IMKNL	390	L4M	4,617
SM	7,986	L8M	4,421
SMC	538	L16M	6,138
MINS	480,363	COUNT	2,926
SS	2,919,074	ALL	1
SS10	2,919,386		

Table 6.1: Perplexity of AP Test according to each model

but do estimate a probability distribution over two possibilities: does belong and does not belong. (The probability that a sequence does not belong is one minus the probability that it does belong.)

Perplexity for MINS and COUNT is not meaningful alone as presented in Table 6.1, since the choice of negative constant for each model is arbitrary.

Since SS is not designed to estimate probability of occurring, it is not surprising that it does poorly on the perplexity measure.

Of the modified SS models tested, L8M has the lowest perplexity. Notice that the relatively high degree of generalization assumed by this model enables the model to do much better than SS, at estimating the probability that AP Test occurs.

6.3.1 Probability of Belonging

Perplexity is not always suitable for evaluating a model's ability to estimate the probability that a sequence belongs. Table 6.2 shows the mean log probability of each sequence, according to each of the conventional language models and according to the baseline model ALL. Again, each model is trained on AP Train. The AP Train column of this table shows mean log probability of AP Train according to each model, where each model is trained using that same sequence. Mean log probability shown in the AP Test column of this table is related to perplexity shown in Table 6.1 according to Equation 6.7.

Model	AP Train	AP Test	NYT	EW	RW
GT	-3.19	-6.58	-7.66	-14.97	-13.42
GTL	-3.23	-6.34	-7.38	-14.72	-13.42
IMKN	-3.32	-6.17	-6.98	-11.35	-7.09
IMKNL	-3.34	-5.97	-6.78	-11.47	-6.95
SM	-0.00	-8.99	-10.95	-22.51	-18.15
ALL	0	0	0	0	0

Table 6.2: Mean (word) log probability of each sequence, according to the conventional models and the baseline model ALL

The highest mean log probability (and lowest perplexity) of AP Test is achieved by the model ALL. According to that model, the probability of AP Test is one. However, this model assigns this same probability to all other sequences. Clearly, ALL is not useful for distinguishing sequences that belong from those that do not.

The two conventional models with the lowest perplexity (IMKN and IMKNL) are also the two conventional models which make the least distinction between sequences in AP Test (that belong) and sequences in RW (that do not). This lack of distinction is due in part to the fact that these two models assign higher probability to RW than is assigned by any other conventional language model considered. This result suggests that an estimate of the probability that a sequence occurs may not always be useful as an estimate of the probability that the sequence belongs.

Among the various conventional language models, IMKN and IMKNL have lowest perplexity, yet assign lowest (worst) mean log probability to AP Train, which sequence is not only known to belong, but is also observed during training. Ideally, a model which estimates a probability of belonging would assign high probability to a sequence which the model has already observed and which the model already knows to belong. SM assigns much higher mean log probability to AP Train, than do IMKN and IMKNL, yet SM has higher perplexity.

Model	AP Train	AP Test	NYT	EW	RW
SMC	-0.08	-6.29	-7.99	-27.16	-682.18
MINS	0	-13.08	-15.61	-40.93	-1629.33
SS	0	-14.89	-17.65	-45.03	-1672.50
SS10	0	-14.89	-17.65	-45.03	-1672.50
L1K	-0.00	-13.03	-15.77	-42.35	-1372.78
L1M	-0.04	-8.99	-10.66	-27.65	-876.37
L2M	-0.09	-8.67	-10.23	-26.26	-827.36
L4M	-0.17	-8.44	-9.89	-24.98	-779.22
L8M	-0.34	-8.39	-9.73	-23.89	-732.84
L16M	-0.68	-8.72	-9.94	-23.21	-689.98
COUNT	0	-7.98	-16.62	-166.60	-395.16

Table 6.3: Mean (word) log probability of each sequence, according to each other model (not already shown in Table 6.2)

6.4 Distinguishing Sequences

Because perplexity is not always suitable for evaluating a model's ability to estimate the probability that a sequence belongs, we evaluate a model's ability to estimate this probability by comparing the model's estimate of the mean log probability of a sequence known to belong to that of a sequence known not to belong. Table 6.4 shows the ratio r of mean log probability of each sequence to mean log probability of RW:

$$r = \left(\frac{\log P_m(S)}{\|S\|}\right)\left(\frac{\log P_m(RW)}{\|RW\|}\right)^{-1} \tag{6.8}$$

Mean log probabilities not shown in Table 6.2 are shown in Table 6.3.

The ratio of NYT mean log probability to RW mean log probability, according to the IMKNL model is 0.9748, showing relatively little distinction between the two sequences compared to the corresponding COUNT ratio of 0.0421. To understand why IMKNL might make so little distinction between the two sequences, consider that the model assumes up front that RW belongs. The mean log probability of RW according to IMKNL is largely an answer to the question of how probable it is for an unseen word to follow two other unseen words, given the frequency of that occurrence observed in training data.

Model	AP Train/RW	AP Test/RW	NYT/RW	EW/RW
GT	0.2375	0.4901	0.5709	1.1154
GTL	0.2404	0.4723	0.5501	1.0970
IMKN	0.4679	0.8697	0.9847	1.6006
IMKNL	0.4809	0.8581	0.9748	1.6504
SM	0.0000	0.4950	0.6032	1.2400
SMC	0.0001	0.0092	0.0117	0.0398
MINS	**0**	**0.0080**	**0.0096**	**0.0251**
SS	**0**	0.0089	0.0106	0.0269
SS10	**0**	0.0089	0.0106	0.0269
L1K	0.0000	0.0095	0.0115	0.0308
L1M	0.0000	0.0103	0.0122	0.0315
L2M	0.0001	0.0105	0.0124	0.0317
L4M	0.0002	0.0108	0.0127	0.0321
L8M	0.0005	0.0115	0.0133	0.0326
L16M	0.0010	0.0126	0.0144	0.0336
COUNT	**0**	0.0202	0.0421	0.4216
ALL	1	1	1	1

Table 6.4: Ratio of mean log probability of each sequence to mean log probability of RW, according to each model

Model	Perplexity	AP Test/RW
SS	2,919,074	0.0089
L8M	4,421	0.0115

Table 6.5: Comparison of SS and L8M (modified SS)

6.5 Avoiding an Assumed Degree of Generalization

SS and L8M are very similar models, yet L8M estimates probability of occurring by assuming a degree of generalization while SS estimates probability of belonging by avoiding such an assumption. L8M does better than SS at estimating probability of occurring as measured by perplexity of AP Test, while SS does better than L8M at estimating probability of belonging as measured by the ratio of AP Test mean log probability to RW mean log probability, summarized in Table 6.5.

One way to avoid assuming a degree of generalization for unseen words (to assign non-zero probability) is to model a sequence of words as a sequence of characters, which character model can improve estimation of the probability that the sequence belongs. The ratio of mean log probability

of a sequence to RW mean log probability is generally much lower (better) for character models, as opposed to word models. For example, the AP Test/RW ratio according to SMC is 0.0092, while that same ratio according to SM is 0.4950. A reason for this contrast is that word models ignore potentially useful character-level information found in unseen words. All conventional word models we tested assign higher mean log probability to a sequence of randomly generated words (RW) than to a sequence of English words (EW).

Table 6.4 shows that, according to at least one set of experiments and evaluation measure, both MINS and SS perform better than any of the other methods we tested, at distinguishing between sequences that belong and those that do not. For example, the ratio of AP Test mean log probability to RW mean log probability is lower according to MINS and SS than according to any of the other methods.

6.6 Efficiency

The amount of time and space used by a model to score AP Test is shown in Table 6.6. Both GT and IMKN are implemented in C, while the others models are implemented in Java. Of the models shown in this table:

- GT and IMKN use the least time and space.
- SM and SMC use the most time to train.
- SS10 uses the most time to test.
- MINS uses less time and space than all others except GT and IMKN. Note that MINS uses less time and space than Segment Selection by design, since Segment Selection employs MINS to initialize the Gibbs sampler (see Section 5.8).

	Time			
Model	Train	Test	Total	Space
GT and IMKN	1	0	1	1
SM	152	2	154	71
SMC	136	19	155	51
MINS	34	1	35	24
SS	83	23	106	55
SS10	113	59	172	66

Table 6.6: Time (minutes) and space (gigabytes) used by a model

Chapter 7

Conclusion

It is possible to estimate the probability that a word sequence belongs to a chosen language, by avoiding an assumed degree of generalization. We introduce two methods for doing so, MINS and Segment Selection, and evaluate both by distinguishing sequences that belong to the language from those that do not.

It appears that MINS may be more effective than Segment Selection, at estimating this probability, the reason for which may be the subject of future work.

Future work may also include applying MINS and Segment Selection in various areas of research, such as speech recognition, machine translation, information retrieval, bioinformatics, and other areas where it may prove useful to estimate the probability that a sequence of discrete variables belongs to a chosen language.

References

[1] S.F. Altschul, W. Gish, W. Miller, E.W. Myers, and D.J. Lipman. Basic local alignment search tool. *Journal of Molecular Biology*, 215(3):403–410, 1990.

[2] D. Benedetto, E. Caglioti, and V. Loreto. Language trees and zipping. *Physical Review Letters*, 88(4):48702, 2002.

[3] W.B. Cavnar and J.M. Trenkle. N-gram-based text categorization. In *Proceedings of the Symposium on Document Analysis and Information Retrieval*, pages 161–175, 1994.

[4] S. Chib. Marginal likelihood from the Gibbs output. *Journal of the American Statistical Association*, pages 1313–1321, 1995.

[5] J.G. Cleary and W.J. Teahan. Unbounded length contexts for PPM. *The Computer Journal*, 40(2 and 3):67–75, 1997.

[6] F. Denis, R. Gilleron, A. Laurent, and M. Tommasi. Text classification and co-training from positive and unlabeled examples. In *Proceedings of the ICML 2003 workshop: the continuum from labeled to unlabeled data*, pages 80–87, 2003.

[7] S. Geman and D. Geman. Stochastic relaxation, Gibbs distributions, and the Bayesian restoration of images. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 6(2):721–741, 1984.

[8] J. Goodman and S. Chen. An empirical study of smoothing techniques for language modeling. *Computer Speech and Language*, 13(4):359–393, 1999.

[9] J.T. Goodman. A bit of progress in language modeling. *Computer Speech & Language*, 15(4):403–434, 2001.

[10] H. Lodhi, C. Saunders, J. Shawe-Taylor, N. Cristianini, and C. Watkins. Text classification using string kernels. *The Journal of Machine Learning Research*, 2:419–444, 2002.

[11] C.D. Manning and H. Schutze. *Foundations of statistical natural language processing*. MIT Press, 2000.

[12] P. Mcnamee and J. Mayfield. Character n-gram tokenization for European language text retrieval. *Information Retrieval*, 7(1):73–97, 2004.

[13] I. Murray and R. Salakhutdinov. Evaluating probabilities under high-dimensional latent variable models. In *Advances in Neural Information Processing Systems (NIPS'08)*, volume 21, pages 1137–1144, 2009.

[14] J. Pearl. Evidential reasoning using stochastic simulation of causal models. *Artificial Intelligence*, 32(2):245–257, 1987.

[15] B. Roark, M. Saraclar, M. Collins, and M. Johnson. Discriminative language modeling with conditional random fields and the perceptron algorithm. In *Proceedings of the 42nd Annual Meeting of the Association for Computational Linguistics*, page 47, 2004.

[16] A. Stolcke. SRILM - An extensible language modeling toolkit. In *Seventh International Conference on Spoken Language Processing*, volume 3, pages 901–904, 2002.

[17] Y.W. Teh. A hierarchical Bayesian language model based on Pitman-Yor processes. In *Proceedings of the 21st International Conference on Computational Linguistics and the 44th Annual Meeting of the Association for Computational Linguistics*, page 992, 2006.

[18] E. Ukkonen. On-line construction of suffix trees. *Algorithmica*, 14(3):249–260, 1995.

[19] F. Wood, C. Archambeau, J. Gasthaus, L. James, and Y.W. Teh. A stochastic memoizer for sequence data. In *Proceedings of the 26th Annual International Conference on Machine Learning*, 2009.