A Gaussian Prior for Smoothing Maximum Entropy Models

Stanley F. Chen Ronald Rosenfeld

February 1999
CMU-CS-99-108

School of Computer Science
Carnegie Mellon University
Pittsburgh, PA 15213

Abstract

In certain contexts, maximum entropy (ME) modeling can be viewed as maximum likelihood training for exponential models, and like other maximum likelihood methods is prone to overfitting of training data. Several smoothing methods for maximum entropy models have been proposed to address this problem, but previous results do not make it clear how these smoothing methods compare with smoothing methods for other types of related models. In this work, we survey previous work in maximum entropy smoothing and compare the performance of several of these algorithms with conventional techniques for smoothing n-gram language models. Because of the mature body of research in n-gram model smoothing and the close connection between maximum entropy and conventional n-gram models, this domain is well-suited to gauge the performance of maximum entropy smoothing methods. Over a large number of data sets, we find that an ME smoothing method proposed to us by Lafferty [1] performs as well as or better than all other algorithms under consideration. This general and efficient method involves using a Gaussian prior on the parameters of the model and selecting maximum *a posteriori* instead of maximum likelihood parameter values. We contrast this method with previous n-gram smoothing methods to explain its superior performance.

Keywords: language models, maximum entropy, smoothing

1 Introduction

Maximum entropy (ME) modeling has been successfully applied to a wide range of domains, including language modeling as well as many other natural language tasks [2, 3, 4, 5]. For many problems, this type of modeling can be viewed as maximum likelihood (ML) training for exponential models, and like other maximum likelihood methods is prone to overfitting of training data. While several smoothing methods for maximum entropy models have been proposed to address this problem [6, 1, 7, 5], previous results do not make it clear how these smoothing methods compare with smoothing methods for other types of related models.

However, there has a been great deal of research in smoothing n-gram language models, and it can be shown that maximum entropy n-gram models are closely related to conventional n-gram models. Consequently, this domain is well-suited to gauging the performance of maximum entropy smoothing methods relative to other smoothing techniques.

In this work, we survey previous work in maximum entropy smoothing and compare the performance of several of these algorithms with conventional techniques for smoothing n-gram language models. Evaluating the perplexity of each method over a large number of data sets, we find that the ME smoothing method proposed to us by Lafferty [1] performs as well as or better than all other algorithms under consideration. In this method, a Gaussian prior on model parameters is applied and maximum *a posteriori* instead of maximum likelihood parameter values are selected. While simple and efficient, this method exhibits all of the behaviors that have been observed by Chen and Goodman to be beneficial for n-gram smoothing [8].

In the remainder of this section, we present an introduction to maximum entropy modeling and discuss why smoothing ME models is necessary. In Section 2, we introduce n-gram language models and summarize previous work on smoothing these models. We list the desirable properties of smoothing algorithms observed by Chen and Goodman. In Section 3, we introduce maximum entropy n-gram models and discuss their relationship with conventional n-gram models. In Section 4, we survey previous work in smoothing maximum entropy models and in Section 5, we present Lafferty's Gaussian prior method. We contrast this method with smoothing algorithms for conventional n-gram models and show that it satisfies all of criteria of Chen and Goodman. In Section 6, we present results of experiments comparing a number of maximum entropy and conventional smoothing techniques on n-gram language modeling. Finally, in Section 7 we discuss our conclusions.

1.1 Maximum Entropy Modeling

Consider the task of estimating a probability distribution $q(x)$ over a finite set $x \in \Omega$ given some training data set $X = \{x_1, \ldots, x_N\}$. Intuitively, our task is to find a distribution $q(x)$ similar to the empirical distribution $\tilde{p}(x)$ given by the training data

$$\tilde{p}(x) = \frac{c_X(x)}{N}$$

where $c_X(x)$ denotes the number of times x occurs in X and where N is the size of the X. In the extreme case, we can take $q(x)$ to be identical to $\tilde{p}(x)$, but this will typically lead to overfitting to the training data. Instead, it would be better to require that $q(x)$ match only those properties of $\tilde{p}(x)$ that we deem to be significant and that can be reliably estimated from the training data.

For example, consider $x = (w_1, w_2)$ where w_1 and w_2 are English words, and let the training data X be the list of consecutive word pairs, or *bigrams*, that occur in some large corpus of English text. Thus, the task is estimating the frequency of English bigrams. Consider a bigram that does not occur in the training data, say PIG DOG. We have $\tilde{p}(\text{PIG DOG}) = 0$, but intuitively we want $q(\text{PIG DOG}) > 0$ since this bigram has *some* chance of occurring. This is an example of a property of $\tilde{p}(x)$ that we do not deem significant and thus do not want to match exactly with $q(x)$. However, let us assume that we observe that the word THE occurs with frequency 0.05 in the training data, *i.e.*,

$$\sum_{w_2} \tilde{p}(\text{THE } w_2) = \sum_{w_1} \tilde{p}(w_1 \text{ THE}) = 0.05 \quad .$$

Because of the abundance of the word THE, this is presumably an accurate estimate of this frequency and it seems reasonable to require that our selected distribution $q(x)$ satisfies the analogous constraint

$$\sum_{w_2} q(\text{THE } w_2) = \sum_{w_1} q(w_1 \text{ THE}) = 0.05 \quad . \tag{1}$$

More generally, we can select a number of nonnegative random variables or *features* $\vec{f} = \{f_1(x), \ldots, f_F(x)\}$ and require that the expected value of each feature over the model $q(x)$ is equal to that of the empirical distribution $\tilde{p}(x)$:

$$\sum_x q(x) f_i(x) = \sum_x \tilde{p}(x) f_i(x), \quad i = 1, \ldots, F \quad . \tag{2}$$

The constraint expressed in equation (1) can be expressed with two such features,

$$f_j(w_1, w_2) = \begin{cases} 1 & \text{if } w_j = \text{THE} \\ 0 & \text{otherwise} \end{cases}$$

for $j = 1, 2$.

The constraints given in equation (2) do not generally specify a unique model $q(x)$, but a set of models $Q_{\vec{f}}$. The *maximum entropy* principle states that we should select the model $q(x) \in Q_{\vec{f}}$ with the largest entropy $H(q) = -\sum_x q(x) \log q(x)$ [9]. Intuitively, models with high entropy are more uniform and correspond to assuming less about the world. The maximum entropy model can be interpreted as the model that assumes only the knowledge that is represented by the features derived from the training data, and nothing else.

The maximum entropy paradigm has many elegant properties [2, 3]. The maximum entropy model is unique and can be shown to be an exponential model of the form

$$q_{\text{ME}}(x) = \frac{1}{Z_\Lambda} \exp\left(\sum_{i=1}^{F} \lambda_i f_i(x)\right) \tag{3}$$

where $Z_\Lambda = \sum_x \exp(\sum_{i=1}^F \lambda_i f_i(x))$ is a normalization factor and $\Lambda = (\lambda_1, \ldots, \lambda_F)$ are the parameters of the model. Furthermore, the maximum entropy model is also the maximum likelihood model in the class of exponential models given by equation (3).[1] Finally, the log-likelihood of the training

[1]These properties hold when constraining feature expectations to be equal to those found on a training set. When constraining expectations to alternate values, the maximum entropy model will not be the maximum likelihood model, and the ME model will not exist if the constraints are inconsistent.

data is concave in the model parameters Λ, and thus it is relatively easy to find the unique maximum entropy/maximum likelihood model using algorithms such as generalized iterative scaling [10] or improved iterative scaling [3].

While models with high entropy tend to be rather uniform or smooth and we may only constrain properties of $q(x)$ we consider significant, a maximum entropy model can still overfit training data even with small numbers of constraints. For example, consider constraints on the frequency of the word MATEO and the bigram SAN MATEO, and assume that the word MATEO only occurs after the word SAN in the training data. Then, we will have

$$\sum_{w_1} q(w_1 \text{ MATEO}) = \sum_{w_1} \tilde{p}(w_1 \text{ MATEO}) = \tilde{p}(\text{SAN MATEO})$$

and

$$q(\text{SAN MATEO}) = \tilde{p}(\text{SAN MATEO})$$

which implies $q(w_1 \text{ MATEO}) = 0$ for all $w_1 \neq \text{SAN}$. Intuitively, we want $q(x) > 0$ for all $x \in \Omega$ since all bigrams have some chance of occurring. Zero probabilities lead to infinite loss in log-loss objective functions and can lead to poor performance in many applications, *e.g.*, when $q(x)$ represents a language model to be used in speech recognition. Thus, it is desirable to *smooth* maximum entropy models, or adjust parameter values away from their maximum likelihood estimates.

2 Smoothing N-Gram Language Models

While there has been relatively little work in smoothing maximum entropy models, there has been a great deal of work in smoothing n-gram language models. A *language model* is a probability distribution $q(s)$ over word sequences s that models how often each sequence s occurs as a sentence. Language models have many applications, including speech recognition, machine translation, and spelling correction [11, 12, 13].

For a word sequence $s = w_1 \cdots w_l$, we can express its probability $\Pr(s)$ as

$$\begin{aligned}\Pr(s) &= \Pr(w_1) \times \Pr(w_2|w_1) \times \cdots \times \Pr(w_l|w_1 \cdots w_{l-1}) \times \Pr(\text{END}|w_1 \cdots w_l) \\ &= \prod_{i=1}^{l+1} \Pr(w_i|w_1 \cdots w_{i-1})\end{aligned}$$

where the token $w_{l+1} = \text{END}$ signals the end of the sentence. The most widely-used language models, by far, are n-gram language models. In an n-gram model, we make the approximation that the identity of a word depends only on past words through the identity of the last $n-1$ words, giving us

$$\Pr(s) = \prod_{i=1}^{l+1} \Pr(w_i|w_1 \cdots w_{i-1}) \approx \prod_{i=1}^{l+1} \Pr(w_i|w_{i-(n-1)}^{i-1})$$

where the notation w_i^j denotes the sequence $w_i \cdots w_j$ and where $w_{-n+2}, \ldots, w_0$ are all taken to be some distinguished beginning-of-sentence token.

The maximum likelihood estimate $q_{\text{ML}}(w_i|w_{i-(n-1)}^{i-1})$ of the probabilities $\Pr(w_i|w_{i-(n-1)}^{i-1})$ over some training data X can be calculated by simply counting how often the token w_i follows the *history* or context $w_{i-(n-1)}^{i-1}$ and dividing by the total number of times the history occurs, *i.e.*,

$$q_{\mathrm{ML}}(w_i|w_{i-(n-1)}^{i-1}) = \frac{c_X(w_{i-(n-1)}^{i})}{c_X(w_{i-(n-1)}^{i-1})} = \frac{c_X(w_{i-(n-1)}^{i})}{\sum_{w_i} c_X(w_{i-(n-1)}^{i})} \quad .$$

However, the maximum likelihood estimates of these probabilities typically lead to overfitting, and instead it is desirable to use smoothed estimates of these values. For example, one simple smoothing technique is to linearly interpolate the maximum likelihood estimate of the n-gram probability $q_{\mathrm{ML}}(w_i|w_{i-(n-1)}^{i-1})$ with an estimate of the $(n-1)$-gram probability $\Pr(w_i|w_{i-(n-2)}^{i-1})$ [14, 15]:

$$q_{\mathrm{int}}(w_i|w_{i-(n-1)}^{i-1}) = \lambda\, q_{\mathrm{ML}}(w_i|w_{i-(n-1)}^{i-1}) + (1-\lambda)\, q_{\mathrm{int}}(w_i|w_{i-(n-2)}^{i-1}), \quad 0 \le \lambda \le 1 \quad . \tag{4}$$

The lower-order estimate can be defined analogously, and the recursion can end with a unigram or uniform distribution. Since the lower-order distributions are less sparsely estimated from the training data, their interpolation generally reduces overfitting. A large number of other smoothing methods for n-gram models have been proposed, *e.g.*, [16, 8, 14, 17, 18, 19].

We present a brief overview of past work in n-gram model smoothing. One basic observation is that the maximum likelihood estimate of the probability of an n-gram that does not occur in the training data is zero and is thus too low, and consequently the ML probabilities of n-grams with nonzero counts are generally too high. This dichotomy motivates the following framework for expressing smoothing methods, which can be used to express most existing smoothing techniques [18]:

$$q_{\mathrm{sm}}(w_i|w_{i-(n-1)}^{i-1}) = \begin{cases} \alpha(w_i|w_{i-(n-1)}^{i-1}) & \text{if } c_X(w_{i-(n-1)}^{i}) > 0 \\ \gamma(w_{i-(n-1)}^{i-1})\, q_{\mathrm{sm}}(w_i|w_{i-(n-2)}^{i-1}) & \text{if } c_X(w_{i-(n-1)}^{i}) = 0 \end{cases} \quad . \tag{5}$$

That is, if an n-gram $w_{i-(n-1)}^{i}$ occurs in the training data, the estimate $\alpha(w_i|w_{i-(n-1)}^{i-1})$ is used; this estimate is generally a discounted version of the maximum likelihood estimate. Otherwise, we back off to a scaled version of the $(n-1)$-gram distribution $q_{\mathrm{sm}}(w_i|w_{i-(n-2)}^{i-1})$, where the lower-order distribution is typically defined analogously to the higher-order distribution. The scaling factor $\gamma(w_{i-(n-1)}^{i-1})$ is chosen to assure that each conditional distribution sums to 1. The algorithm described by equation (4) can be placed in this framework with the following relations:

$$\begin{aligned} \alpha(w_i|w_{i-(n-1)}^{i-1}) &= q_{\mathrm{int}}(w_i|w_{i-(n-1)}^{i-1}) \\ \gamma(w_{i-(n-1)}^{i-1}) &= 1-\lambda \\ q_{\mathrm{sm}}(w_i|w_{i-(n-2)}^{i-1}) &= q_{\mathrm{int}}(w_i|w_{i-(n-2)}^{i-1}) \quad . \end{aligned}$$

There are three primary distinctions between smoothing algorithms: whether an algorithm is *interpolated* or *backed-off*, what type of discounting is applied to the ML estimate to calculate $\alpha(w_i|w_{i-(n-1)}^{i-1})$, and how lower-order distributions are computed.

In interpolated models, the probability estimate $\alpha(w_i|w_{i-(n-1)}^{i-1})$ of an n-gram $w_{i-(n-1)}^{i}$ with nonzero count depends on the probability assigned to the corresponding $(n-1)$-gram $w_{i-(n-2)}^{i}$, as in equation (4). In backed-off models, the probability estimate of an n-gram with nonzero count is determined while ignoring information from lower-order distributions. Interpolated models include Jelinek-Mercer smoothing [14] and Witten-Bell smoothing [16]; backed-off models include Katz smoothing [17], absolute discounting [19], and Kneser-Ney smoothing [18].

To describe the different types of discounting, we write $\alpha(w_i|w_{i-(n-1)}^{i-1})$ as

$$\alpha(w_i|w_{i-(n-1)}^{i-1}) = \frac{c_X(w_{i-(n-1)}^{i}) - d(w_{i-(n-1)}^{i})}{c_X(w_{i-(n-1)}^{i-1})} + \beta(w_{i-(n-1)}^{i})$$

where $d(w_{i-(n-1)}^{i})$ can be viewed as the discount in count space from the ML estimate and where $\beta(w_{i-(n-1)}^{i})$ is the contribution from lower-order distributions. The value $\beta(w_{i-(n-1)}^{i})$ is zero for backed-off models and typically $\gamma(w_{i-(n-1)}^{i-1})q_{\text{sm}}(w_i|w_{i-(n-2)}^{i-1})$ for interpolated models. In *linear discounting*, the discount $d(w_{i-(n-1)}^{i})$ is taken to be proportional to the original count $c_X(w_{i-(n-1)}^{i})$, as in equation (4) where the discount is $(1-\lambda)\cdot c_X(w_{i-(n-1)}^{i})$. In *absolute discounting*, $d(w_{i-(n-1)}^{i})$ is taken to be a constant $0 \le D \le 1$. In Good-Turing discounting, the discount is calculated using the Good-Turing estimate [20], a theoretically-motivated discount that has been shown to be accurate in non-sparse data situations [17, 21]. A brief description of the Good-Turing estimate is given in Section 4.1. Jelinek-Mercer smoothing and Witten-Bell smoothing use linear discounting; Kneser-Ney smoothing uses absolute discounting, and Katz smoothing and Church-Gale smoothing [21] use Good-Turing discounting.

The final major distinction between smoothing algorithms is how the lower-order probability estimates are calculated. While most smoothing methods define the lower-order model $q_{\text{sm}}(w_i|w_{i-(n-2)}^{i-1})$ analogously to the higher-order model $q_{\text{sm}}(w_i|w_{i-(n-1)}^{i-1})$, in Kneser-Ney smoothing a different approach is taken. The $(n-1)$-gram model is chosen to satisfy certain constraints derived from the training data, namely

$$\sum_{w_{i-(n-1)}} c_X(w_{i-(n-1)}^{i-1})q_{\text{sm}}(w_i|w_{i-(n-1)}^{i-1}) = c_X(w_{i-(n-2)}^{i}) \tag{6}$$

for all $(n-1)$-grams $w_{i-(n-2)}^{i}$. This constraint can be rephrased as: The expected number of times $w_{i-(n-2)}^{i}$ occurs in the training data given the model $q_{\text{sm}}(w_i|w_{i-(n-1)}^{i-1})$ and the history frequencies $c_X(w_{i-(n-1)}^{i-1})$ should equal the actual number of times it occurs. Kneser-Ney smoothing can be applied recursively to lower-order distributions, in which case the constraints (6) are not satisfied exactly. Instead, the right-hand side of the constraints are discounted with absolute discounting.

Chen and Goodman [8] provide an extensive comparison of all of the widely-used smoothing techniques. They evaluate each algorithm on a wide range of training sets through its *perplexity* on test data. The perplexity $\text{PP}_q(X')$ of a model q on a test set X' is the reciprocal of the geometric average probability that the model assigns to each word in the test set. They also use the derivative measure *cross-entropy* $H_q(X') = \log_2 \text{PP}_q(X')$, which can be interpreted as the average number of bits needed to code each word in the test set using the model q. Chen *et al.* [8, 22] also conducted experiments investigating how the cross-entropy of a language model is related to its performance when used in a speech recognition system. They found a strong linear correlation between the two metrics when comparing models that only differ in smoothing.

In terms of perplexity, Chen and Goodman found that Kneser-Ney smoothing and variations consistently outperform all other algorithms. More specifically, they present four main conclusions:

- The factor that affects performance the most is the use of a modified lower-order distributions as in Kneser-Ney smoothing. This is the primary reason for the excellent performance of this algorithm.

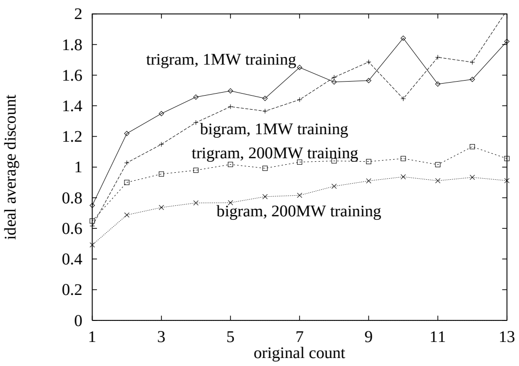

Figure 1: Ideal average discount for n-grams with given count in training data for 1 million word training set and 200 million word training set, bigram and trigram models

- Absolute discounting is superior to linear discounting. For n-grams with a given count r in the training data, they calculate the average discount in count space $d(w_{i-(n-1)}^i)$ from the ML estimate that would cause the expected number of these n-grams in a test set to be equal to their actual number (assuming $\beta(w_{i-(n-1)}^i) = 0$). This ideal average discount is displayed in Figure 1 for counts $r \leq 13$ for two training sets for bigram and trigram models. From this graph, we see why a fixed discount works well. While Good-Turing discounting is actually better than absolute discounting at predicting the *average* discount, it has yet to be used in such a way as to predict the correct discounts in individual distributions well.

- Interpolated models outperform backed-off models when considering performance on just n-grams with low counts in the training data. This is because lower-order models provide valuable information for estimating the probabilities of n-grams with low counts.

- Adding free parameters to an algorithm and optimizing these parameters on held-out data can improve the performance of an algorithm.

Based on these observations, Chen and Goodman propose an algorithm named *modified Kneser-Ney smoothing* that is found to outperform all other methods considered. It is an interpolated variation of Kneser-Ney smoothing with an augmented version of absolute discounting. Instead of using a single discount D for all n-grams, three separate discounts D_1, D_2, and D_{3+} are used for n-grams with one count, two counts, and three or more counts, respectively. This is motivated by the observation that the ideal discount for one-counts and two-counts is significantly smaller than the ideal discount of larger counts, as shown in Figure 1.

3 Maximum Entropy N-Gram Models

We can construct language models very similar to conventional n-gram models within the maximum entropy framework. The maximum entropy models described in Section 1.1 are joint models; to create the conditional distributions used in conventional n-gram models we use the framework introduced by Brown *et al.* [23]. Instead of estimating a joint distribution $q(x)$ over samples x, we estimate a conditional distribution $q(y|x)$ over samples (x, y). Instead of constraints as given by equation (2), we have constraints of the form

$$\sum_{x,y} \tilde{p}(x) q(y|x) f_i(x, y) = \sum_{x,y} \tilde{p}(x, y) f_i(x, y) \quad . \tag{7}$$

This can be interpreted as replacing $q(x, y)$ in the joint formulation with $\tilde{p}(x)q(y|x)$. That is, we assume that history frequencies $\tilde{p}(x)$ are taken from the training data, and we only estimate conditional probabilities. Conditional ME models share many of the same properties as joint models, including being maximum likelihood models, and have computational and performance advantages over joint models in language modeling [24, 5]. A conditional maximum entropy model has the form

$$q_{\mathrm{ME}}(y|x) = \frac{1}{Z_\Lambda(x)} \exp(\sum_{i=1}^{F} \lambda_i f_i(x, y)) \quad . \tag{8}$$

To construct a maximum entropy n-gram model, we take $x = w_{i-(n-1)}^{i-1}$ to be the history and $y = w_i$ to be the following word. For each m-gram $\theta = w_{i-(m-1)}^{i}$ with $m = 1, \ldots, n$ that occurs in the training data, we include a constraint that forces the conditional expectation of θ according to q to be the same as its frequency in the training data. The corresponding features $f_\theta(x, y)$ are

$$f_\theta(x, y) = \begin{cases} 1 & \text{if } (x, y) = w_{i-(n-1)}^{i} \text{ ends in } \theta \\ 0 & \text{otherwise} \end{cases} \quad .$$

Substituting these features into equation (7) and simplifying, we arrive at constraints of the form

$$\sum_{w_{i-(n-1)}^{i} : \mathrm{suffix}(w_{i-(n-1)}^{i}) = \theta} \tilde{p}(w_{i-(n-1)}^{i-1}) q(w_i | w_{i-(n-1)}^{i-1}) = \tilde{p}(\theta) \quad . \tag{9}$$

In fact, the only solution to these constraints is $q(y|x) = q_{\mathrm{ML}}(y|x)$. Consequently, the maximum entropy model is identical to the maximum likelihood n-gram model and it will be beneficial to smooth the estimates of the model parameters $\Lambda = \{\lambda_\theta\}$.

Remarkably, the set of models given by equation (8) with n-gram features is identical to the set of models described by equation (5), which we used to express most existing smoothing algorithms for conventional n-gram models. To see this, let us define a set of m-gram models $q_{\mathrm{ME}}(w_i|w_{i-(m-1)}^{i-1})$ for $m = 1, \ldots, n$ as in equation (8), where each m-gram model only contains features corresponding to word sequences up to length m, and where all models share the same parameter set Λ. Then, to describe the maximum entropy model form in terms of equation (5), we take

$$\begin{aligned}
\alpha(w_i|w_{i-(n-1)}^{i-1}) &= q_{\mathrm{ME}}(w_i|w_{i-(n-1)}^{i-1}) \\
&= \exp(\lambda_{w_{i-(n-1)}^{i}}) \times \frac{Z_\Lambda(w_{i-(n-2)}^{i-1})}{Z_\Lambda(w_{i-(n-1)}^{i-1})} \times q_{\mathrm{ME}}(w_i|w_{i-(n-2)}^{i-1}) \\
\gamma(w_i|w_{i-(n-1)}^{i-1}) &= \frac{Z_\Lambda(w_{i-(n-2)}^{i-1})}{Z_\Lambda(w_{i-(n-1)}^{i-1})} \\
q_{\mathrm{sm}}(w_i|w_{i-(n-2)}^{i-1}) &= q_{\mathrm{ME}}(w_i|w_{i-(n-2)}^{i-1}) \quad .
\end{aligned}$$

For any model of the form (5), we can choose λ_θ using the above equations starting from lower-order models to higher-order to construct an equivalent exponential model.[2] Because smoothing λ_θ estimates in maximum entropy n-gram models and smoothing conventional n-gram models both consider the same class of models, these two tasks are closely related.

4 Smoothing Maximum Entropy Models

In this section and Section 5, we survey previous work in maximum entropy model smoothing. Here, we discuss *Good-Turing discounting*, *fuzzy maximum entropy*, and *fat constraints*; in the following section, we present the Gaussian prior method proposed by Lafferty. In describing these methods, we sometimes use the joint maximum entropy formulation for simplicity. All of these techniques apply equally well to conditional ME models; the analogous conditional ME equations can be derived by replacing x with (x, y) and $q(x)$ with $\tilde{p}(x)q(y|x)$.

4.1 Good-Turing Discounting

Good-Turing discounting has been proposed by Lau [7] and Rosenfeld [5] and can be viewed as the maximum entropy analog to Katz smoothing for conventional n-gram models. They observe that the marginals of the model $q(y|x)$ should not be constrained to be exactly those of the empirical distribution $\tilde{p}(y|x)$, but instead target values should be discounted as in conventional n-gram smoothing. Instead of constraints as given by equation (9), they propose the following constraints

$$\sum_{w_{i-(n-1)}^{i}:\text{suffix}(w_{i-(n-1)}^{i})=\theta} \tilde{p}(w_{i-(n-1)}^{i-1})q(w_i|w_{i-(n-1)}^{i-1}) = \tilde{p}_{\text{GT}}(\theta)$$

where $\tilde{p}_{\text{GT}}(\theta)$ is the Good-Turing estimate of the frequency of θ.

The Good-Turing estimate [20] is a theoretically motivated method for estimating the average discount for an event based on its count in the training data. For an event that occurs r times in N samples, in contrast with the maximum likelihood estimate $\frac{r}{N}$, the Good-Turing estimate of the event's true frequency is $\frac{r^*}{N}$ where

$$r^* = \frac{n_{r+1}}{n_r}(r+1)$$

and where n_r is the number of members of the population with exactly r counts. Katz [17] suggests applying this estimate to each joint m-gram distribution, $m = 1, \ldots, n$, separately. Furthermore, as n_r can be very low or zero for large r, Katz proposes a method where n-grams with large counts are not discounted and discounts for low counts are adjusted to compensate. Lau and Rosenfeld use the Katz variation of Good-Turing discounting.

However, when constraining marginals of a model to Good-Turing discounted marginals of the training data, the constraints may no longer be consistent and a maximum entropy model may not exist. For example, in a trigram model consider features that constrain the frequencies of the n-grams TIC TAC TOE and TAC TOE and assume that the word TAC only follows the word TIC in the training data. Then, we will have the constraints

$$\tilde{p}(\text{TIC TAC})q(\text{TOE}|\text{TIC TAC}) = \tilde{p}_{\text{GT}}(\text{TIC TAC TOE})$$

[2] The equivalence is not exact as exponential models cannot express probabilities equal to zero or one. In addition, for the equivalence to hold the unigram model given by equation (5) must assign the same probability to all words not occurring in the training data. This is generally the case with existing smoothing algorithms.

and

$$\sum_{w_{i-2}} \tilde{p}(w_{i-2} \text{ TAC})q(\text{TOE}|w_{i-2} \text{ TAC}) = \tilde{p}(\text{TIC TAC})q(\text{TOE}|\text{TIC TAC}) = \tilde{p}_{\text{GT}}(\text{TAC TOE}) \quad .$$

In general, we will have $\tilde{p}_{\text{GT}}(\text{TIC TAC TOE}) \neq \tilde{p}_{\text{GT}}(\text{TAC TOE})$ since discounts for n-grams of different length are calculated independently; consequently, these constraints will be inconsistent. In practice, there are no dire consequences to having inconsistent constraints. While training algorithms such as iterative scaling may not converge, a reasonable procedure is to stop training once performance on some held-out set stops improving. However, inconsistency is symptomatic of constraints that will lead to poor parameter estimates.

Lau [7] compares the performance of Good-Turing discounting for smoothing ME n-gram models with deleted interpolation [14], a variation of Jelinek-Mercer smoothing, for conventional n-gram models. For a 5 million word training set of Wall Street Journal text, deleted interpolation yielded a perplexity of 225 on a 870,000 word test set. The maximum entropy model yielded a slightly superior perplexity of 221, where all n-grams that occurred only once in the training data were excluded from the ME model. However, later results by Chen and Goodman [8] strongly indicate that other smoothing methods for conventional n-gram models, such as modified Kneser-Ney smoothing, would outperform deleted interpolation by a much larger margin.

4.2 Fuzzy Maximum Entropy

In the fuzzy maximum entropy framework developed by Della Pietra and Della Pietra [25], instead of requiring that constraints are satisfied exactly, a penalty is associated with inexact constraint satisfaction. Finding the maximum entropy model is equivalent to finding the model $q(x)$ satisfying the given constraints that minimizes the Kullback-Leibler distance $D(q \parallel p_{\text{unif}})$ from the uniform model $p_{\text{unif}}(x)$. In fuzzy maximum entropy, the objective function is taken to be

$$D(q \parallel p_{\text{unif}}) + \tau U(q) \tag{10}$$

where $U(\cdot)$ is a penalty function minimized when constraints are satisfied exactly and τ is a weighting parameter. Della Pietra and Della Pietra suggest a penalty function of the form

$$U(q) = \frac{1}{2}\sum_{i=1}^{F} \frac{1}{\sigma_i^2}\left[\sum_x q(x)f_i(x) - \sum_x \tilde{p}(x)f_i(x)\right]^2 \quad .$$

This penalty function can be interpreted as the logarithm of a Gaussian distribution with diagonal covariance centered around the target constraint values. The variance σ_i^2 associated with feature $f_i(x)$ can be estimated from the empirical distribution of $f_i(x)$ in the training data. Della Pietra and Della Pietra describe a variant of generalized iterative scaling that can be used to find the optimal model under this objective function [26].

We can interpret this algorithm from the viewpoint of maximum *a posteriori* (MAP) estimation. In MAP estimation, we attempt to find the model q with the highest posterior probability given the training data X:

$$\arg\max_q \Pr(q|X) = \arg\max_q \Pr(q)\Pr(X|q) = \arg\max_q \left[\log \Pr(q) + \log \Pr(X|q)\right] \quad . \tag{11}$$

The MAP objective function has two terms, a prior term *log* $\Pr(q)$ and a likelihood term *log* $\Pr(X|q)$. The fuzzy ME objective function given in equation (10) is analogous to the MAP objective function. The first term in equation (10) encourages the model q to have high entropy, and can be thought of as taking the role of a prior distribution favoring more uniform models. The second term encourages q to satisfy the given constraints, *i.e.*, to fit the training data well. This term plays a similar role as the likelihood term in MAP estimation. However, both terms in the fuzzy ME objective function are very different than their MAP counterparts. In Lafferty's Gaussian prior method to be discussed in Section 5, a traditional MAP objective function is used, and we contrast the two approaches in that section.

Lau [7] constructed a fuzzy maximum entropy n-gram model excluding all n-grams with only one count using the data sets described in Section 4.1, yielding a perplexity of 230. This is slightly worse than the perplexities achieved by the deleted interpolation and Good-Turing discounted ME models.

4.3 Fat Constraints

Other methods for relaxing constraints include work by Newman [27] and Khudanpur [28]. In these algorithms, instead of selecting the maximum entropy model over models $q(x)$ that satisfy a set of constraints exactly, they only require that the given marginals of $q(x)$ fall in some range around the target values. Newman suggests a constraint of the form

$$\sum_{i=1}^{F} W_i \left[\sum_x q(x) f_i(x) - \sum_x \tilde{p}(x) f_i(x) \right]^2 \leq \sigma^2$$

with feature weights W_i for the task of estimating power spectra. Khudanpur suggests constraints of the form

$$\alpha_i \leq \sum_x q(x) f_i(x) \leq \beta_i, \quad i = 1, \ldots, F \quad .$$

Both of these approaches can be viewed as instances of the fuzzy maximum entropy framework. Instead of a smooth function, the penalty $U(q)$ is taken to be zero if q satisfies the relaxed constraints and infinite otherwise. These types of methods have yet to be applied to language modeling.

5 A Gaussian Prior

Lafferty [1] proposes applying a Gaussian prior on the parameters Λ to smooth maximum entropy models. This technique has been applied previously in [6]. Recall that a maximum entropy model is the maximum likelihood model among the set of models given by equation (3). By performing maximum *a posteriori* instead of maximum likelihood estimation, we can apply a Gaussian prior centered around $\Lambda = \vec{0}$ to smooth the ML model toward the uniform model, thus hopefully ameliorating overfitting.

More precisely, we can equate finding the maximum entropy model with finding parameters Λ that maximize the log-likelihood $L_X(\Lambda)$ of the training data X

$$L_X(\Lambda) = \sum_x \tilde{p}(x) \log q_\Lambda(x) \quad .$$

With the Gaussian prior, which we take to have diagonal covariance, our objective function $L'_X(\Lambda)$ then takes the form given in equation (11),

$$L'_X(\Lambda) = L_X(\Lambda) + \sum_{i=1}^{F} \log \frac{1}{\sqrt{2\pi\sigma_i^2}} \exp(-\frac{\lambda_i^2}{2\sigma_i^2}) \tag{12}$$

$$= L_X(\Lambda) - \sum_{i=1}^{F} \frac{\lambda_i^2}{2\sigma_i^2} + \mathrm{const}(\Lambda) \quad , \tag{13}$$

where the σ_i^2 are the variances of the Gaussian.

We contrast this objective function with the objective function in equation (10) for fuzzy maximum entropy. Both functions have two terms, one which prefers models that are more uniform and one which prefers models similar to the training data. In the Lafferty framework, the term that prefers models similar to the training data is a log-likelihood value, while the analogous term in fuzzy ME is a sum of squared constraint errors. The suitability of either function ultimately depends on the application. However, for language modeling it has been found that likelihood, or the derivative measure perplexity, is a useful performance metric.

As to the Gaussian prior method used in the Lafferty framework, the analogous term in fuzzy ME that favors more uniform models is $D(q \parallel p_{\mathrm{unif}})$. The former function penalizes models that have many large λ_i values, while the latter function penalizes models far from uniform. While the correct function to use will again depend on the application, we argue that the former function is generally more appealing. For example, a model with a single nonzero λ_i parameter should intuitively receive a small penalty, regardless of how far from uniform this model is. Thus, we hypothesize that the Gaussian prior method is preferable to fuzzy maximum entropy.

The Gaussian prior method adds little computation to existing maximum entropy training algorithms. Since the logarithm of the Gaussian prior is concave, the objective function is still concave in Λ and it is reasonably easy to find the optimal model. We can make a simple modification to improved iterative scaling [3] to find the MAP model. The original update of each λ_i in this algorithm is to take

$$\lambda_i^{(t+1)} \leftarrow \lambda_i^{(t)} + \delta_i^{(t)}$$

where $\delta_i^{(t)}$ satisfies the equation

$$\sum_x \tilde{p}(x) f_i(x) = \sum_x q_{\Lambda^{(t)}}(x) f_i(x) \exp(\delta_i^{(t)} f^\#(x)) \tag{14}$$

and where $f^\#(x) = \sum_i f_i(x)$. With the Gaussian prior, equation (14) is replaced with

$$\sum_x \tilde{p}(x) f_i(x) = \sum_x q_{\Lambda^{(t)}}(x) f_i(x) \exp(\delta_i^{(t)} f^\#(x)) + \frac{\lambda_i^{(t)} + \delta_i^{(t)}}{\sigma_i^2} \quad . \tag{15}$$

As the right-hand side of this equation is strictly monotonic in $\delta_i^{(t)}$, it is relatively easy to find its solution using a search algorithm. We derive this modified update rule in the appendix.

5.1 The Gaussian Prior Method and Conventional N-Gram Smoothing

In Section 2, we listed four factors that were found by Chen and Goodman to significantly affect n-gram smoothing performance. It is informative to assess the Gaussian prior method according to these four criteria.

First, they point out that the modified lower-order distributions of Kneser-Ney smoothing is the primary reason for its superiority among conventional n-gram smoothing algorithms. Recall that these distributions are chosen to satisfy marginal constraints as given in equation (6). However, this set of constraints is *identical* to the corresponding maximum entropy constraints for $(n-1)$-grams, as given by equation (9). Thus, maximum entropy n-gram models have similar modified lower-order distributions as in Kneser-Ney smoothing.

However, MAP models under the Gaussian prior no longer satisfy the ME constraints exactly. Instead, the constraints that are satisfied have the form

$$\sum_x \tilde{p}(x) f_i(x) - \frac{\lambda_i}{\sigma_i^2} = \sum_x q_\Lambda(x) f_i(x) \quad . \tag{16}$$

That is, the empirical expectations $\sum_x \tilde{p}(x) f_i(x)$ are now "discounted" by the amount λ_i/σ_i^2. (In ME n-gram models, most λ_i are positive.) Qualitatively, this is even more desirable than meeting the targets exactly, as empirical frequencies tend to be higher than true frequencies for events with nonzero counts. Analogous behavior is produced with Kneser-Ney smoothing when applied recursively to lower-order distributions. In this case, target counts are discounted through absolute discounting. A derivation of equation (16) is given in the appendix.

Second, Chen and Goodman point out that absolute discounting is superior to the other types of discounting considered, and that using a different discount for one-counts and two-counts and a flat discount thereafter as in modified Kneser-Ney smoothing performs even better. With the Gaussian prior, the discount for an n-gram θ is linear in λ_θ as can be seen from equation (16). As the probability assigned to θ by q_Λ grows exponentially in λ_θ, λ_θ grows logarithmically as a function of the target probability or count. In other words, roughly speaking the Gaussian prior method translates to *logarithmic* discounting. This is a qualitatively appealing model of the ideal average discount displayed in Figure 1 and is more elegant than using multiple flat discounts.[3]

Third, Chen and Goodman report that interpolated models outperform backed-off models on n-grams with low counts, as lower-order models provide valuable information for estimating these probabilities. Happily, the Gaussian prior behaves like an interpolated model as n-gram probability estimates depend on lower-order information. This follows trivially from the observation that the probability q_Λ assigns to an n-gram θ depends on the parameter values $\lambda_{\theta'}$ for all n-grams θ' that are suffixes of θ. However, the Gaussian prior method uses the information from lower-order models in a meaningful way. For any n-gram θ, the Gaussian prior method tends to adjust λ_θ towards zero; when λ_θ is zero, the corresponding feature has no effect on the model, and the lower-order n-gram probability estimate is used. In other words, the prior adjusts n-gram probabilities towards the lower-order probability estimate, as is desirable.

Finally, Chen and Goodman note that additional tunable parameters can improve current smoothing methods. For the Gaussian prior, the natural free parameters are the variances σ_i.

[3] We can contrast a Gaussian prior on Λ parameters with previous work in n-gram smoothing where priors have been applied directly in probability space. MacKay and Peto [29] use a Dirichlet prior and Nádas [30] uses a Beta prior, both resulting in linear discounting which has been shown to perform suboptimally.

In the basic version of the Gaussian prior method that we implemented, we had n free parameters σ_m, $m = 1, \ldots, n$, where all m-grams of the same length were constrained to have the same variance σ_m^2.

As the Gaussian prior method satisfies all of the desiderata listed by Chen and Goodman, it may perform competitively in n-gram smoothing. The experiments in Section 6 show that this is indeed the case.

6 Experiments

To compare the performance of maximum entropy and conventional n-gram smoothing techniques, we ran experiments over many training set sizes using several different text corpora for both bigram and trigram models.

6.1 Methodology

Of the conventional n-gram smoothing techniques, we implemented Katz smoothing [17], which is perhaps the most popular algorithm in practice, and modified Kneser-Ney smoothing [8], which has been shown to outperform all other widely-used techniques. In addition, we implemented the variation of Jelinek-Mercer smoothing given by equation (4) where instead of a single λ parameter a different λ_m is used for each level of the n-gram model. This method does not perform particularly well, but is used as a baseline algorithm for expository purposes. We refer to these three implementations with the mnemonics `katz`, `kneser-ney-mod`, and `baseline`, respectively.

We also implemented several maximum entropy smoothing techniques. For each technique, all λ_θ parameters are initialized to zero and improved iterative scaling is applied to train the model. Iterative scaling is terminated when the perplexity of a held-out set no longer decreases significantly. Cluster expansion [31] is employed to reduce computation. In the implementation `ME-no-smooth`, no smoothing is performed. (Since training is terminated when performance on a held-out set no longer improves, no probabilities will converge to zero as in the case where training is continued to convergence.) The algorithm `ME-disc-katz` is an implementation of Good-Turing discounting as described in Section 4.1. The algorithm `ME-gauss` is an implementation of Lafferty's Gaussian prior method as described in Section 5. As mentioned earlier, this method has n free parameters σ_m, one for each level of the n-gram model.

We used data from four sources: the Brown corpus, which contains text from a number of miscellaneous sources [32]; Wall Street Journal (WSJ) newspaper text [33]; the Broadcast News (BN) corpus, which contains transcriptions of television and radio news shows [34]; and the Switchboard (SWB) corpus, which contains transcriptions of telephone conversations [35]. In each experiment, we selected a training set of a given length from one source, and two held-out sets from the same source. The first held-out set was used to optimize the parameters of each smoothing algorithm, *e.g.*, the σ_m parameters of `ME-gauss` or the discounts D_* of modified Kneser-Ney smoothing. Parameters were selected to minimize the perplexity of the held-out set; Powell's search algorithm [36] was used to perform this search. This held-out set was also used to decide when to terminate iterative scaling for the ME models. The second held-out set was used to evaluate the final perplexity of each smoothing algorithm.

For each data source, we ran experiments using training sets from 100 sentences (about 2,000 words) to around 100,000 sentences (about 2 million words). Held-out sets were 2,500 sentences.

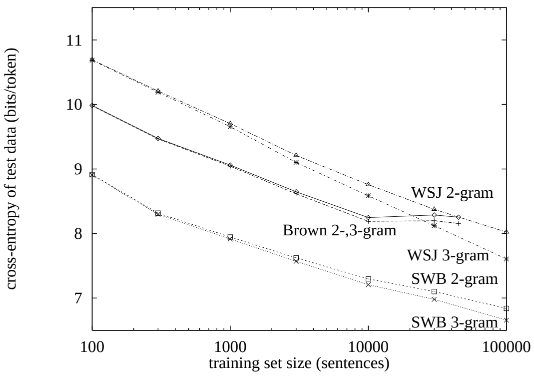

Figure 2: Cross-entropy of baseline smoothing algorithm on test set over multiple training set sizes on Brown, Switchboard, and Wall Street Journal corpora

While training sets for language models may reach hundreds of millions of words in practice, we were unable to consider larger training sets than we did due to computational limitations. Training maximum entropy n-gram models requires a great deal more computation than training conventional n-gram models. In addition, when considering multiple parameter settings in the Powell search (as for the σ_m parameters in `ME-gauss`), the iterative scaling algorithm must be applied separately for each parameter setting. To train a single model using method `ME-gauss` for a 2 million word training set required around six hours of computation on a 400 MHz Pentium II computer. Substantially larger training sets are feasible if parameter optimization is not used.

Our data sets are identical to those used by Chen and Goodman [8] and consequently our results are directly comparable to the analogous results presented by Chen and Goodman. More details of our methodology can be found in that work.

6.2 Results

In Figure 2, we display the cross-entropy of the baseline Jelinek-Mercer smoothing algorithm over a range of training set sizes on several corpora. In the graphs to follow, we display the performance of each algorithm as the difference of its cross-entropy on the test set from the cross-entropy of the baseline method (using the same training set), to facilitate visualization. Each point in the graphs presented here represents a single experiment; for an analysis of the standard error of these observations refer to Chen and Goodman [8]. To give a rough idea of the statistical error involved, in Figures 4 and 5, the difference between `kneser-ney-mod` and `ME-gauss` may not be significant, while the difference between these two algorithms and all of the others almost certainly is for almost every data point.

In Figure 3, we compare the performance of the various maximum entropy smoothing algorithms over multiple training set sizes using Wall Street Journal data. The left graph is for bigram models and the right graph is for trigram models. We see that `ME-no-smooth` is outperformed by the other algorithms by a large margin, demonstrating the necessity of smoothing for maximum entropy models. Of the remaining algorithms, the Gaussian prior method significantly outperformed Good-Turing discounting. Though not shown here, we see similar behavior in experiments on the other

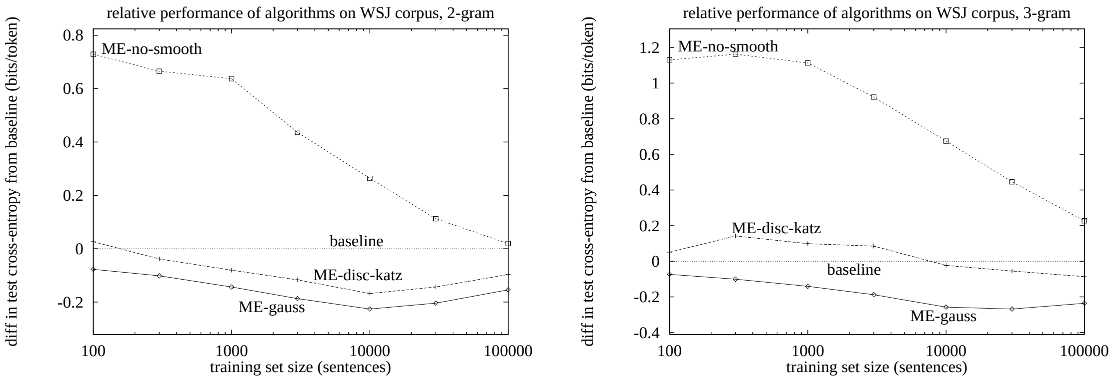

Figure 3: Performance relative to baseline of various maximum entropy smoothing algorithms over multiple training set sizes on the Wall Street Journal corpus, bigram and trigram models

three corpora.[4]

In Figures 4 and 5, we compare the performance of maximum entropy smoothing algorithms with conventional n-gram smoothing algorithms over several corpora.[5] Of the conventional smoothing methods, we see that Katz smoothing generally outperforms the baseline and that modified Kneser-Ney smoothing is significantly better. Of the maximum entropy methods, we see that `ME-disc-katz` performs comparably to Katz smoothing for bigram models, and somewhat worse for trigram models, though it does better on larger data sets. The method `ME-gauss` performs about as well as modified Kneser-Ney smoothing, and is slightly better over most data sets. Thus, the Gaussian prior method performs as well as or better than all other widely-used algorithms for smoothing n-gram models.

To investigate how the logarithmic discounting of the Gaussian prior compares to the multiple absolute discounts of modified Kneser-Ney smoothing, we computed how closely the expected number of certain n-grams in a test set according to each model matched the actual number of those n-grams in the test set. In particular, for all n-grams occurring r times in a 750,000 word training set X for some r, we computed the ratio of the expected number of times these n-grams occurred in a 10,000,000 word test set X' to the actual number of times they occurred:

$$\frac{\sum_{w_{i-(n-1)}^{i}:c_X(w_{i-(n-1)}^{i})=r} c_{X'}(w_{i-(n-1)}^{i-1})q(w_i|w_{i-(n-1)}^{i-1})}{\sum_{w_{i-(n-1)}^{i}:c_X(w_{i-(n-1)}^{i})=r} c_{X'}(w_{i-(n-1)}^{i})}\ .$$

These ratios are displayed for $r < 40$ in Figure 6 for bigram and trigram models. The Gaussian prior achieves ratios closer to the ideal value of one than modified Kneser-Ney smoothing for most r, which is evidence that the Gaussian prior method is superior to multiple flat discounts at predicting correct average discounts.

We also investigated how the number of independent variance parameters used with the Gaussian prior affects performance. In the original implementation `ME-gauss`, a different σ_m is used

[4]We also ran experiments using *complemented* n-grams [5], where each n-gram feature is nonzero only when no longer n-gram feature is nonzero. This resulted in significantly inferior performance.

[5]The large spikes in the Switchboard graphs are discussed by Chen and Goodman [8]. They are caused by a duplicated segment of text in the corresponding training set.

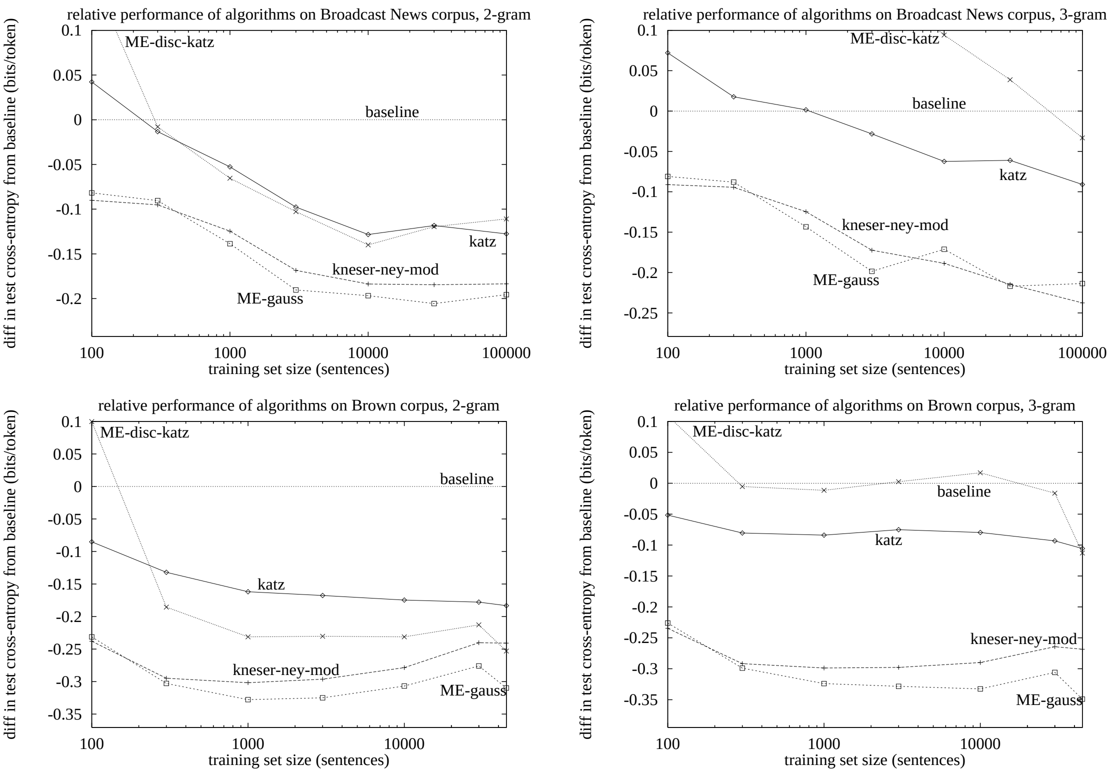

Figure 4: Performance relative to baseline of various smoothing algorithms over multiple training set sizes on the Broadcast News and Brown corpora, bigram and trigram models

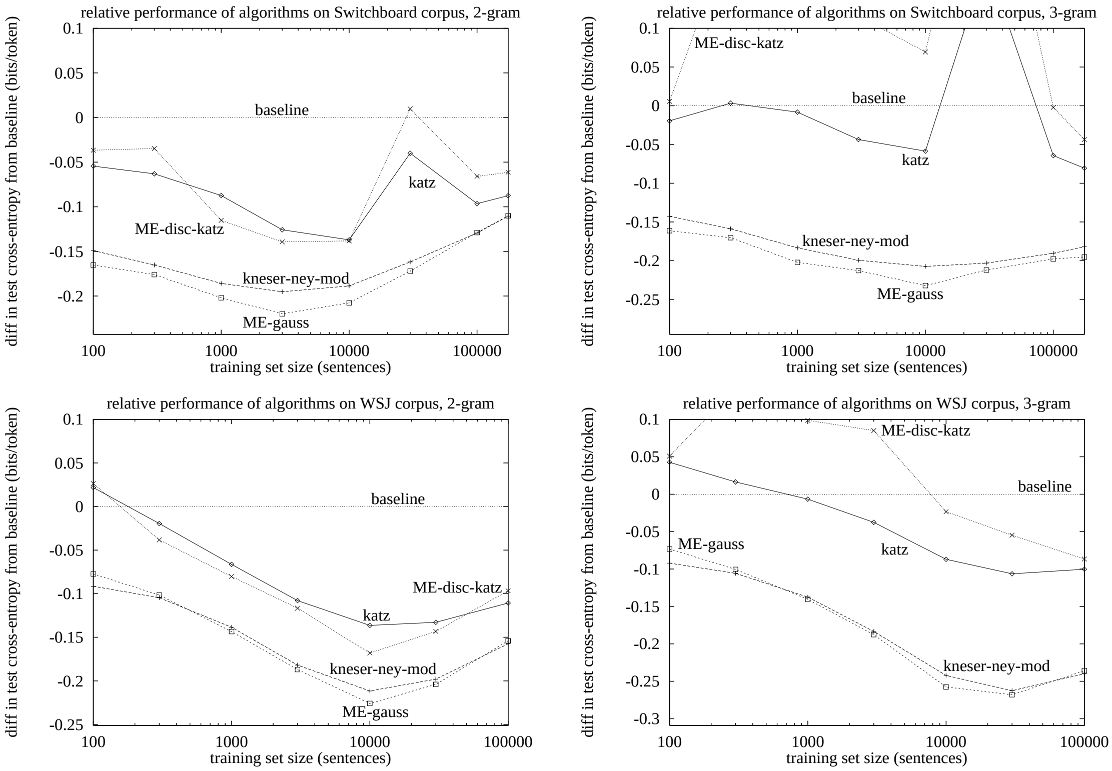

Figure 5: Performance relative to baseline of various smoothing algorithms over multiple training set sizes on the Switchboard and Wall Street Journal corpora, bigram and trigram models

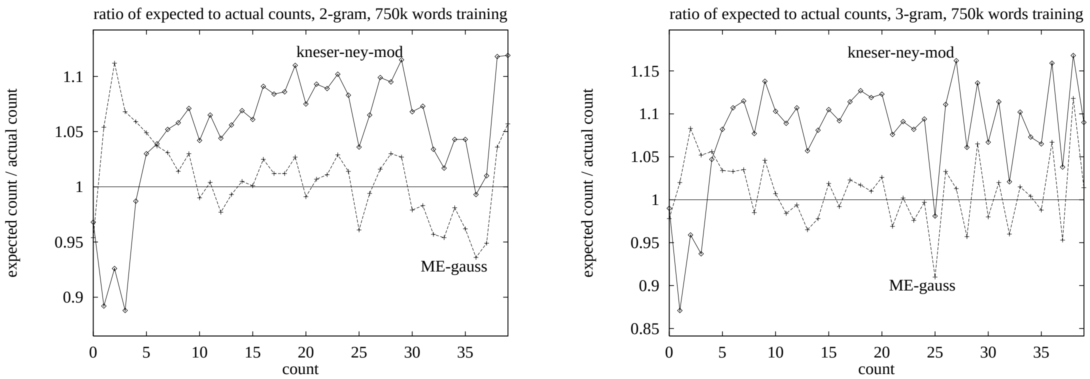

Figure 6: Ratio of expected number to actual number in test set of n-grams with a given count in training data, 750,000 word Wall Street Journal training set, bigram and trigram models

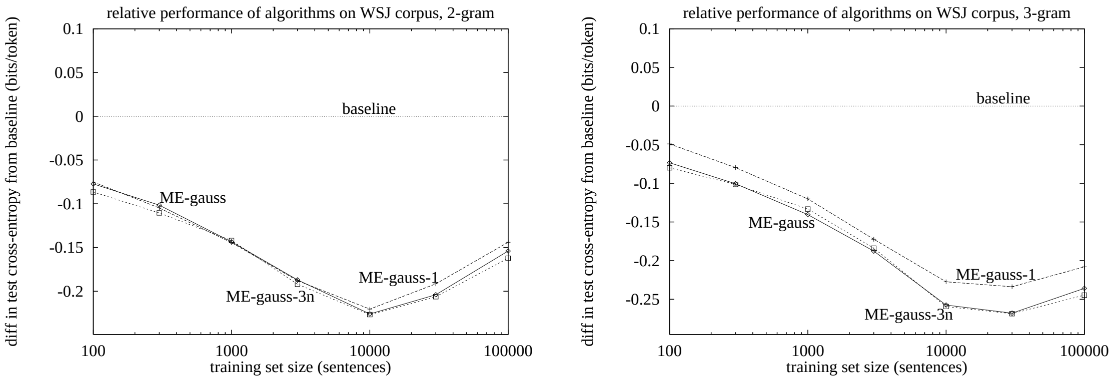

Figure 7: Performance relative to baseline of different parameterizations of Gaussian prior over multiple training set sizes on the Wall Street Journal corpus, bigram and trigram models

for each level of the n-gram model.[6] We also considered using a single σ over the whole model (`ME-gauss-1`), and using three parameters $\sigma_{m,1}$, $\sigma_{m,2}$, and $\sigma_{m,3+}$ for each level of the n-gram model, to be applied to m-grams with 1, 2, or 3 or more counts in the training data, respectively. This latter parameterization (`ME-gauss-3n`) is analogous to the parameterization of modified Kneser-Ney smoothing. The performance of these three variations on the Wall Street Journal corpus is displayed in Figure 7. The variations `ME-gauss` and `ME-gauss-3n` yield almost identical performance, and the variation `ME-gauss-1` performs slightly worse.[7] As having separate variances for each n-gram level leads to improved performance, this is a useful distinction to make. We also investigated many other parameter-tying schemes, but none significantly outperformed this simple technique.

7 Discussion

It has been argued that maximum entropy models do not require smoothing because they are already as uniform or smooth as possible given the constraints. However, maximum entropy models can be viewed as maximum likelihood exponential models, and have similar properties as other maximum likelihood methods. For example, as can be seen in Figure 3, when data is plentiful, smoothing has a smaller effect, and when data is sparse, smoothing is essential.

In many tasks including language modeling, it has been found that superior performance can be achieved by constructing very large models (so parameters are sparsely estimated) and then smoothing them. Thus, for maximum entropy models to be competitive with other techniques in these domains, we need effective maximum entropy smoothing algorithms.

In this work, we showed that a Gaussian prior can be used to smooth maximum entropy n-gram models to achieve performance equal to or superior to that of all other techniques for

[6] The optimal variances $N\sigma_m^2$ for the Gaussian prior found by the Powell search were mostly in the range $1.5 < N\sigma_m^2 < 5$, where N is the size of the training set. Multiplying by N converts the variances from probability space to count space, and discounts are relatively constant in count space over different training set sizes. The discounts tended to grow with data set size and shrink with m.

[7] The reason that a variation with more parameters may not outperform a variation with fewer parameters is due to search errors in parameter optimization.

smoothing n-gram models, a field which has an extensive body of associated research. This is the first clear demonstration that a maximum entropy smoothing method can be as effective as smoothing techniques for other types of models, and makes it possible to construct maximum entropy models in sparse data situations without loss of performance. Furthermore, it adds virtually no computational cost to the maximum entropy training procedure. However, because of the large underlying computational cost of maximum entropy algorithms, building maximum entropy models for very large data sets is still a challenging problem.

While this smoothing method can be expressed very simply, we show that it possesses all of the desirable qualities of n-gram smoothing noted by Chen and Goodman from empirical analysis. In addition, it achieves its excellent performance using fewer parameters than the comparably performing modified Kneser-Ney smoothing.

The Gaussian prior and the Kneser-Ney methods consistently outperform other smoothing techniques. The distinction between these algorithms and the others is their use of modified lower-order distributions as described in Sections 2 and 5.1. These distributions are chosen to satisfy certain marginal constraints derived from the training data. Thus, the use of marginal constraints may be a powerful technique for designing novel smoothing algorithms, whether for language modeling or for other domains. Enforcing marginal constraints would mark a significant departure from traditional techniques used in smoothing.

In addition, the Gaussian prior is a qualitatively different prior than has been used previously in n-gram smoothing. As touched on in Section 5.1, linear discounting can be motivated through a Dirichlet or Beta prior on probabilities [29, 30], but it has been shown to perform poorly. While absolute discounting yields better performance, it is unclear how to elegantly express this technique through a prior distribution. In contrast, the Gaussian prior is applied to λ_θ parameters which are linear in log-probability, and leads to logarithmic discounting. This simple prior yields discounting that is qualitatively and quantitatively similar to the empirical ideal.

Not only can the Gaussian prior be applied to maximum entropy modeling, but it can also be applied in the more general *minimum divergence* paradigm [37, 38]. Maximizing entropy is equivalent to finding the model with the smallest Kullback-Leibler divergence from the uniform distribution. In minimum divergence modeling, one selects the model satisfying the given constraints closest to some default distribution $q_0(x)$. The model $q_0(x)$ can be used to express prior knowledge about the domain. Minimum divergence models have the form

$$q_{\mathrm{MD}}(x) = \frac{1}{Z_\Lambda}\, q_0(x)\, \exp(\sum_{i=1}^{F} \lambda_i f_i(x)) \quad .$$

The analysis in Section 5 applies to these models without modification.

Maximum entropy modeling has advantages over competing approaches in terms of elegance, generality, and performance, and the Gaussian prior is a powerful tool for smoothing general ME models. Whether the Gaussian prior proves superior to other algorithms in domains other than n-gram modeling is still an open empirical question. In n-gram models, no features partially overlap each other, and this is not the case in general. In addition, how parameters should be tied in other domains has yet to be explored.[8] Nonetheless, our results and analysis justify the choice of a Gaussian prior for use in n-gram modeling, and strongly suggest its use in other situations as well.

[8]With n-gram models, we found that a single variance σ for the whole model worked quite well, though using separate σ_m for each level of the n-gram model worked slightly better. However, this partitioning is not applicable in general.

A Derivation of Modified Constraints and Modified Iterative Scaling for the Gaussian Prior

In this section, we derive the modified constraints given in equation (16) and the modified update for improved iterative scaling given in equation (15) for the Gaussian prior method suggested by Lafferty. We use the conditional ME formulation. For further details about improved iterative scaling such as proof of convergence, refer to [3].

To derive the modified constraints, we take the partial derivatives of the objective function given in equation (13) with respect to the parameters λ_i and set them to zero.

$$
\begin{aligned}
L'_X(\Lambda) &= \sum_{x,y} \tilde{p}(x,y) \log q_\Lambda(y|x) - \sum_{i=1}^{F} \frac{\lambda_i^2}{2\sigma_i^2} + \mathrm{const}(\Lambda) \\
&= \sum_{x,y} \tilde{p}(x,y) \sum_i \lambda_i f_i(x,y) - \sum_{x,y} \tilde{p}(x,y) \log \sum_{y'} \exp(\sum_i \lambda_i f_i(x,y')) - \sum_{i=1}^{F} \frac{\lambda_i^2}{2\sigma_i^2} + \mathrm{const}(\Lambda) \\
\frac{\partial L'_X(\Lambda)}{\partial \lambda_i} &= \sum_{x,y} \tilde{p}(x,y) f_i(x,y) - \sum_{x,y} \tilde{p}(x,y) \sum_{y'} \frac{\exp(\sum_i \lambda_i f_i(x,y'))}{Z_\Lambda(x)} f_i(x,y') - \frac{\lambda_i}{\sigma_i^2} \\
&= \sum_{x,y} \tilde{p}(x,y) f_i(x,y) - \sum_{x,y} \tilde{p}(x,y) \sum_{y'} q_\Lambda(y'|x) f_i(x,y') - \frac{\lambda_i}{\sigma_i^2} \\
&= \sum_{x,y} \tilde{p}(x,y) f_i(x,y) - \sum_{x} \tilde{p}(x) \sum_{y'} q_\Lambda(y'|x) f_i(x,y') \sum_y \tilde{p}(y|x) - \frac{\lambda_i}{\sigma_i^2} \\
&= \sum_{x,y} \tilde{p}(x,y) f_i(x,y) - \sum_{x,y} \tilde{p}(x) q_\Lambda(y|x) f_i(x,y) - \frac{\lambda_i}{\sigma_i^2}
\end{aligned}
$$

Equation (16) follows simply from the last line.

The derivation of the modified improved iterative scaling update is identical to the original derivation except for the presence of extra terms for the prior. In each iteration, we try to find $\Delta = \{\delta_i\}$ that maximizes the increase in the objective function:

$$
\begin{aligned}
L'_X(\Lambda+\Delta) - L'_X(\Lambda) &= \sum_{x,y} \tilde{p}(x,y) \sum_i \delta_i f_i(x,y) - \\
\sum_x \tilde{p}(x) \log \sum_y q_\Lambda(y|x) \exp(\sum_i \delta_i f_i(x,y)) &- \frac{1}{2\sigma_i^2} \sum_i (2\lambda_i \delta_i + \delta_i^2) \quad .
\end{aligned}
$$

As it is not clear how to maximize this function directly, we find an auxiliary function $B(\Delta)$ that we *can* maximize that bounds this function from below. We would like the maximum of $B(\Delta)$ to be larger than zero whenever Λ is not optimal, *i.e.*, whenever Λ does not satisfy the constraints in equation (16).

Using the inequality $\log x \leq x - 1$, we get

$$
\begin{aligned}
L'_X(\Lambda+\Delta) - L'_X(\Lambda) &\geq \sum_{x,y} \tilde{p}(x,y) \sum_i \delta_i f_i(x,y) + 1 - \\
\sum_x \tilde{p}(x) \sum_y q_\Lambda(y|x) \exp(\sum_i \delta_i f_i(x,y)) &- \frac{1}{2\sigma_i^2} \sum_i (2\lambda_i \delta_i + \delta_i^2) \ = \ A(\Delta) \quad .
\end{aligned}
$$

Substituting in $f^{\#}(x,y) = \sum_i f_i(x,y)$ and applying Jensen's inequality, we arrive at

$$A(\Delta) \geq \sum_{x,y} \tilde{p}(x,y) \sum_i \delta_i f_i(x,y) + 1 -$$

$$\sum_x \tilde{p}(x) \sum_y q_\Lambda(y|x) \sum_i \frac{f_i(x,y)}{f^{\#}(x,y)} \exp(\delta_i f^{\#}(x,y)) - \frac{1}{2\sigma_i^2} \sum_i (2\lambda_i \delta_i + \delta_i^2) = B(\Delta) \quad .$$

Taking the partial derivative of $B(\Delta)$ with respect to δ_i, we get

$$\frac{\partial B(\Delta)}{\partial \delta_i} = \sum_{x,y} \tilde{p}(x,y) f_i(x,y) - \sum_x \tilde{p}(x) \sum_y q_\Lambda(y|x) f_i(x,y) \exp(\delta_i f^{\#}(x,y)) - \frac{\lambda_i + \delta_i}{\sigma_i^2} \quad .$$

Equation (15) follows by setting these derivatives to zero. Notice that $B(\vec{0}) = 0$ and that $\nabla B(\vec{0}) = \vec{0}$ only if Λ satisfies the constraints. It follows that the maximum of $B(\Delta)$ will be larger than zero when Λ is not optimal, as desired.

References

[1] J. Lafferty, personal communication, 1997.

[2] A. Berger, S. Della Pietra, and V. Della Pietra, "A maximum entropy approach to natural language processing," *Computational Linguistics*, vol. 22, no. 1, pp. 39–71, 1996.

[3] S. Della Pietra, V. Della Pietra, and J. Lafferty, "Inducing features of random fields," *IEEE Trans. on Pattern Analysis and Machine Intelligence*, vol. 19, no. 3, 1997.

[4] A. Ratnaparkhi, *Maximum Entropy Models for Natural Language Ambiguity Resolution*, Ph.D. thesis, University of Pennsylvania, Philadelphia, PA, 1998.

[5] R. Rosenfeld, "A maximum entropy approach to adaptive statistical language modeling," *Computer, Speech, and Language*, vol. 10, pp. 187–228, 1996, longer version published as "Adaptive Statistical Language Modeling: A Maximum Entropy Approach," Ph.D. thesis, Computer Science Department, Carnegie Mellon University, TR CMU-CS-94-138, Apr. 1994.

[6] A. Berger and R. Miller, "Just-in-time language modeling," in *ICASSP-98*, Seattle, WA, 1998.

[7] R. Lau, "Adaptive statistical language modelling," M.S. thesis, Department of Electrical Engineering and Computer Science, Massachusetts Institute of Technology, Cambridge, MA, 1994.

[8] S. F. Chen and J. Goodman, "An empirical study of smoothing techniques for language modeling," Tech. Rep. TR-10-98, Harvard University, 1998.

[9] E. Jaynes, "Information theory and statistical mechanics," *Physics Reviews*, vol. 106, pp. 620–630, 1957.

[10] J. Darroch and D. Ratcliff, "Generalized iterative scaling for log-linear models," *The Annals of Mathematical Statistics*, vol. 43, pp. 1470–1480, 1972.

[11] L. R. Bahl, F. Jelinek, and R. L. Mercer, "A maximum likelihood approach to continuous speech recognition," *IEEE Trans. on Pattern Analysis and Machine Intelligence*, vol. PAMI-5, no. 2, pp. 179–190, Mar. 1983.

[12] P. F. Brown, J. Cocke, S. A. Della Pietra, V. J. Della Pietra, F. Jelinek, J. D. Lafferty, R. L. Mercer, and P. S. Roossin, "A statistical approach to machine translation," *Computational Linguistics*, vol. 16, no. 2, pp. 79–85, Jun. 1990.

[13] M. Kernighan, K. Church, and W. Gale, "A spelling correction program based on a noisy channel model," in *Proc. of the Thirteenth International Conf. on Computational Linguistics*, 1990, pp. 205–210.

[14] F. Jelinek and R. L. Mercer, "Interpolated estimation of Markov source parameters from sparse data," in *Proc. of the Workshop on Pattern Recognition in Practice*, Amsterdam, The Netherlands: North-Holland, May 1980.

[15] P. F. Brown, S. A. Della Pietra, V. J. Della Pietra, J. C. Lai, and R. L. Mercer, "An estimate of an upper bound for the entropy of English," *Computational Linguistics*, vol. 18, no. 1, pp. 31–40, Mar. 1992.

[16] T. C. Bell, J. G. Cleary, and I. H. Witten, *Text Compression*, Prentice Hall, Englewood Cliffs, N.J., 1990.

[17] S. M. Katz, "Estimation of probabilities from sparse data for the language model component of a speech recognizer," *IEEE Trans. on Acoustics, Speech and Signal Processing*, vol. ASSP-35, no. 3, pp. 400–401, Mar. 1987.

[18] R. Kneser and H. Ney, "Improved backing-off for m-gram language modeling," in *Proc. of the IEEE International Conf. on Acoustics, Speech and Signal Processing*, 1995, vol. 1, pp. 181–184.

[19] H. Ney, U. Essen, and R. Kneser, "On structuring probabilistic dependences in stochastic language modeling," *Computer, Speech, and Language*, vol. 8, pp. 1–38, 1994.

[20] I. Good, "The population frequencies of species and the estimation of population parameters," *Biometrika*, vol. 40, no. 3 and 4, pp. 237–264, 1953.

[21] K. W. Church and W. A. Gale, "A comparison of the enhanced Good-Turing and deleted estimation methods for estimating probabilities of English bigrams," *Computer, Speech, and Language*, vol. 5, pp. 19–54, 1991.

[22] S. F. Chen, D. Beeferman, and R. Rosenfeld, "Evaluation metrics for language models," in *DARPA Broadcast News Transcription and Understanding Workshop*, 1998.

[23] P. Brown, S. Della Pietra, V. Della Pietra, R. Mercer, A. Nadas, and S. Roukos, "A maximum penalized entropy construction of conditional log-linear language and translation models using learned features and a generalized csiszar algorithm," Internal IBM Report, 1992.

[24] R. Lau, R. Rosenfeld, and S. Roukos, "Adaptive language modeling using the maximum entropy principle," in *Proc. of the ARPA Workshop on Human Language Technology*, 1993, pp. 108–113.

[25] S. Della Pietra and V. Della Pietra, "Statistical modeling by maximum entropy," unpublished report, 1993.

[26] R. Lau, R. Rosenfeld, and S. Roukos, "Trigger-based language models: A maximum entropy approach," in *Proc. of ICASSP-93*, Apr. 1993, pp. II–45 – II–48.

[27] W. Newman, "Extension to the maximum entropy method," *IEEE Trans. on Information Theory*, vol. IT-23, no. 1, pp. 89–93, Jan. 1977.

[28] S. Khudanpur, "A method of maximum entropy estimation with relaxed constraints," in *1995 Johns Hopkins University Language Modeling Workshop Proc.*, Center for Language and Speech Processing, Johns Hopkins University, Baltimore, 1995.

[29] D. J. C. MacKay and L. C. Peto, "A hierarchical Dirichlet language model," *Natural Language Engineering*, vol. 1, no. 3, pp. 1–19, 1995.

[30] A. Nadas, "Estimation of probabilities in the language model of the IBM speech recognition system," *IEEE Trans. on Acoustics, Speech and Signal Processing*, vol. ASSP-32, no. 4, pp. 859–861, Aug. 1984.

[31] J. Lafferty and B. Suhm, "Cluster expansions and iterative scaling for maximum entropy language models," in *Maximum Entropy and Bayesian Methods*, K. Hanson and R. Silver, Eds. Kluwer Academic Publishers, 1995.

[32] H. Kucera and W. Francis, *Computational Analysis of Present-Day American English*, Brown University Press, Providence R.I., 1967.

[33] R. M. Stern, "Specification of the 1995 ARPA hub 3 evaluation: Unlimited vocabulary NAB news baseline," in *Proc. of the DARPA Speech Recognition Workshop*, 1996, pp. 5–7.

[34] A. Rudnicky, "Hub 4: Business Broadcast News," in *Proc. of the DARPA Speech Recognition Workshop*, 1996, pp. 8–11.

[35] J. Godfrey, E. Holliman, and J. McDaniel, "SWITCHBOARD: Telephone speech corpus for research and development," in *Proc. of ICASSP-92*, 1992, vol. I, pp. 517–520.

[36] W. Press, B. Flannery, S. Teukolsky, and W. Vetterling, *Numerical Recipes in C*, Cambridge University Press, Cambridge, 1988.

[37] S. Kullback, *Information Theory and Statistics*, Wiley, New York, 1959.

[38] D. Beeferman, A. Berger, and J. Lafferty, "A model of lexical attraction and repulsion," in *Proc. of the ACL*, Madrid, Spain, 1997.