Inducing Features of Random Fields

Stephen Della Pietra, Vincent Della Pietra, and John Lafferty, *Member, IEEE*

Abstract—**We present a technique for constructing random fields from a set of training samples. The learning paradigm builds increasingly complex fields by allowing potential functions, or features, that are supported by increasingly large subgraphs. Each feature has a weight that is trained by minimizing the Kullback-Leibler divergence between the model and the empirical distribution of the training data. A greedy algorithm determines how features are incrementally added to the field and an iterative scaling algorithm is used to estimate the optimal values of the weights.**

The random field models and techniques introduced in this paper differ from those common to much of the computer vision literature in that the underlying random fields are non-Markovian and have a large number of parameters that must be estimated. Relations to other learning approaches, including decision trees, are given. As a demonstration of the method, we describe its application to the problem of automatic word classification in natural language processing.

Keywords— **Random field, Kullback-Leibler divergence, iterative scaling, maximum entropy, EM algorithm, statistical learning, clustering, word morphology, natural language processing.**

I. Introduction

In this paper we present a method for incrementally constructing random fields. Our method builds increasingly complex fields to approximate the empirical distribution of a set of training examples by allowing potential functions, or features, that are supported by increasingly large subgraphs. Each feature is assigned a weight, and the weights are trained to minimize the Kullback-Leibler divergence between the field and the empirical distribution of the training data. Features are incrementally added to the field using a top-down greedy algorithm, with the intent of capturing the salient properties of the empirical sample while allowing generalization to new configurations. The general problem that the methods we propose address is that of discovering the structure inherent in a set of sample patterns. As one of the fundamental aims of statistical inference and learning, this problem is central to a wide range of tasks including classification, compression, and prediction.

To illustrate the nature of our approach, suppose we wish to automatically characterize spellings of words according to a statistical model; this is the application we develop in Section 5. A field with no features is simply a uniform distribution on ASCII strings (where we take the distribution of string *lengths* as given). The most conspicuous feature of English spellings is that they are most commonly comprised of lower-case letters. The induction algorithm makes this observation by first constructing the field

$$p(\omega) = \frac{1}{Z} e^{\sum_i \lambda_{[\mathrm{a-z}]} \chi_{[\mathrm{a-z}]}(\omega_i)}$$

where χ is an indicator function and the weight $\lambda_{[\mathrm{a-z}]}$ associated with the feature that a character is lower-case is chosen to be approximately 1.944. This means that a string with a lowercase letter in some position is about $7 \approx e^{1.944}$ times more likely than the same string without a lowercase letter in that position. The following collection of strings was generated from the resulting field by Gibbs sampling. (As for all of the examples that will be shown, this sample was generated with annealing, to concentrate the distribution on the more probable strings.)

```
m, r, xevo, ijjiir, b, to, jz, gsr, wq, vf, x, ga,
msmGh, pcp, d, oziVlal, hzagh, yzop, io, advzmxnv,
ijv_bolft, x, emx, kayerf, mlj, rawzyb, jp, ag,
ctdnnnbg, wgdw, t, kguv, cy, spxcq, uzflbbf,
dxtkkn, cxwx, jpd, ztzh, lv, zhpkvnu, l^, r, qee,
nynrx, atze4n, ik, se, w, lrh, hp+, yrqyka'h,
zcngotcnx, igcump, zjcjs, lqpWiqu, cefmfhc, o, lb,
fdcY, tzby, yopxmvk, by, fz,, t, govyccm,
ijyiduwfzo, 6xr, duh, ejv, pk, pjw, l, fl, w
```

The second most important feature, according to the algorithm, is that two adjacent lower-case characters are extremely common. The second-order field now becomes

$$p(\omega) = \frac{1}{Z} e^{\sum_{i\sim j} \lambda_{[\mathrm{a-z}][\mathrm{a-z}]} \chi_{[\mathrm{a-z}][\mathrm{a-z}]}(\omega_{ij}) + \sum_i \lambda_{[\mathrm{a-z}]} \chi_{[\mathrm{a-z}]}(\omega_i)}$$

where the weight $\lambda_{[\mathrm{a-z}][\mathrm{a-z}]}$ associated with adjacent lower-case letters is approximately 1.80.

The first 1000 features that the algorithm induces include the strings s>, <re, ly>, and ing>, where the character “<” denotes beginning-of-string and the character “>” denotes end-of-string. In addition, the first 1000 features include the regular expressions [0-9][0-9] (with weight 9.15) and [a-z][A-Z] (with weight -5.81) in addition to the first two features [a-z] and [a-z][a-z]. A set of strings obtained by Gibbs sampling from the resulting field is shown here:

```
was, reaser, in, there, to, will, ,, was, by,
homes, thing, be, reloverated, ther, which,
conists, at, fores, anditing, with, Mr., proveral,
the, ,, ***, on't, prolling, prothere, ,, mento,
at, yaou, 1, chestraing, for, have, to, intrally,
of, qut, ., best, compers, ***, cluseliment, uster,
of, is, deveral, this, thise, of, offect, inatever,
thifer, constranded, stater, vill, in, thase, in,
youse, menttering, and, ., of, in, verate, of, to
```

These examples are discussed in detail in Section 5.

The induction algorithm that we present has two parts: *feature selection* and *parameter estimation*. The greediness of the algorithm arises in feature selection. In this step each feature in a pool of candidate features is evaluated by estimating the reduction in the Kullback-Leibler divergence that would result from adding the feature to the field. This reduction is approximated as a function of a single parameter, and the largest value of this function is called the *gain* of the candidate. This approximation is one of the key elements of our approach, making it practical to evaluate a large number of candidate features at each stage of the induction algorithm. The candidate with the largest gain is added to the field. In the parameter estimation step, the parameters of the field are estimated using an iterative scaling algorithm. The algorithm we use is a new statistical estimation algorithm

Stephen and Vincent Della Pietra are with Renaissance Technologies, Stony Brook, NY, 11790. E-mail: [sdella,vdella]@rentec.com

John Lafferty is with the Computer Science Department of the School of Computer Science, Carnegie Mellon University, Pittsburgh, PA, 15213. E-mail: lafferty@cs.cmu.edu

that we call *Improved Iterative Scaling*. It is an improvement of the Generalized Iterative Scaling algorithm of Darroch and Ratcliff [12] in that it does not require that the features sum to a constant. The improved algorithm is easier to implement than the Darroch and Ratcliff algorithm, and can lead to an increase in the rate of convergence by increasing the size of the step taken toward the maximum at each iteration. In Section 4 we give a simple, self-contained proof of the convergence of the improved algorithm that does not make use of the Kuhn-Tucker theorem or other machinery of constrained optimization. Moreover, our proof does not rely on the convergence of alternating I-projection as in Csiszár's proof [10] of the Darroch-Ratcliff procedure.

Both the feature selection step and the parameter estimation step require the solution of certain algebraic equations whose coefficients are determined as expectation values with respect to the field. In many applications these expectations cannot be computed exactly because they involve a sum over an exponentially large number of configurations. This is true of the application that we develop in Section 5. In such cases it is possible to approximate the equations that must be solved using Monte Carlo techniques to compute expectations of random variables. The application that we present uses Gibbs sampling to compute expectations, and the resulting equations are then solved using Newton's method.

Our method can be viewed in terms of the *principle of maximum entropy* [19], which instructs us to assume an exponential form for our distributions, with the parameters viewed as Lagrange multipliers. The techniques that we develop in this paper apply to exponential models in general. We formulate our approach in terms of random fields because this provides a convenient framework within which to work, and because our main application is naturally cast in these terms.

Our method differs from the most common applications of statistical techniques in computer vision and natural language processing. In contrast to many applications in computer vision, which involve only a few free parameters, the typical application of our method involves the estimation of thousands of free parameters. In addition, our methods apply to general exponential models and random fields–there is no underlying Markov assumption made. In contrast to the statistical techniques common to natural language processing, in typical applications of our method there is no probabilistic finite-state or push-down automaton on which the statistical model is built.

In the following section we describe the form of the random field models considered in this paper and the general learning algorithm. In Section 3 we discuss the feature selection step of the algorithm and briefly address cases when the equations need to be estimated using Monte Carlo methods. In Section 4 we present the Improved Iterative Scaling algorithm for estimating the parameters, and prove the convergence of this algorithm. In Section 5 we present the application of inducing features of spellings, and finally in Section 6 we discuss the relation between our methods and other learning approaches, as well as possible extensions of our method.

II. The Learning Paradigm

In this section we present the basic algorithm for building up a random field from elementary features. The basic idea is to incrementally construct an increasingly detailed field to approximate a reference distribution $\tilde{p}$. Typically the distribution $\tilde{p}$ is obtained as the empirical distribution of a set of training examples. After establishing our notation and defining the form of the random field models we consider, we present the training problem as a statement of two equivalent optimization problems. We then discuss the notions of a candidate feature and the gain of a candidate. Finally, we give a statement of the induction algorithm.

A. Form of the random field models

Let $G = (E, V)$ be a finite graph with vertex set V and edge set E, and let $\mathcal{A}$ be a finite alphabet. The *configuration space* Ω is the set of all labelings of the vertices in V by letters in $\mathcal{A}$. If $C \subset V$ and $\omega \in \Omega$ is a configuration, then ω_C denotes the configuration restricted to C. A *random field* on G is a probability distribution on Ω. The set of all random fields is nothing more than the simplex Δ of all probability distributions on Ω. If $f : \Omega \to \mathbf{R}$ then the *support* of f, written $\mathrm{supp}(f)$, is the smallest vertex subset $C \subset V$ having the property that whenever $\omega, \omega' \in \Omega$ with $\omega_C = \omega'_C$ then $f(\omega) = f(\omega')$.

We consider random fields that are given by Gibbs distributions of the form

$$p(\omega) = \frac{1}{Z} e^{\sum_C V_C(\omega)}$$

for $\omega \in \Omega$, where $V_C : \Omega \to \mathbf{R}$ are functions with $\mathrm{supp}(V_C) = C$. The field is *Markov* if whenever $V_C \neq 0$ then C is a *clique*, or totally connected subset of V. This property is expressed in terms of conditional probabilities as

$$p(\omega_u \,|\, \omega_v,\ v \neq u) = p(\omega_u \,|\, \omega_v,\ (u, v) \in E)$$

where u and v are arbitrary vertices. We assume that each C is a path-connected subset of V and that

$$V_C(\omega) = \sum_{1 \le i \le n_C} \lambda_i^C f_i^C(\omega) = \lambda^C \cdot f^C(\omega)$$

where $\lambda_i^C \in \mathbf{R}$ and $f_i^C(\omega) \in \{0, 1\}$. We say that the values λ_i^C are the *parameters* of the field and that the functions f_i^C are the *features* of the field. In the following, it will often be convenient to use notation that disregards the dependence of the features and parameters on a vertex subset C, expressing the field in the form

$$p(\omega) = \frac{1}{Z} e^{\sum_i \lambda_i f_i(\omega)} = \frac{1}{Z} e^{\lambda \cdot f(\omega)}.$$

For every random field $(E, V, \{\lambda_i, f_i\})$ of the above form, there is a field $(E', V, \{\lambda_i, f_i\})$ that is Markovian, obtained by completing the edge set E to ensure that for each i, the subgraph generated by the vertex subset $C = \mathrm{supp}(f_i)$ is totally connected.

If we impose the constraint $\lambda_i = \lambda_j$ on two parameters λ_i and λ_j, then we say that these parameters are *tied*. If λ_i and λ_j are tied, then we can write

$$\lambda_i f_i(\omega) + \lambda_j f_j(\omega) = \lambda g(\omega)$$

where $g = f_i + f_j$ is a *non-binary* feature. In general, we can collapse any number of tied parameters onto a single parameter associated with a non-binary feature. Having tied parameters is often natural for a particular problem, but the presence of non-binary features generally makes the estimation of parameters more difficult.

An *automorphism* σ of a graph is a permutation of the vertices that takes edges to edges: $(u, v) \in E$ if and only if $(\sigma u, \sigma v) \in E$. A random field $(E, V, \{\lambda_i, f_i\})$ is said to have *homogeneous features* if for each feature f_i and automorphism σ of the graph $G = (E, V)$, there is a feature f_j such that $f_j(\sigma\omega) = f_i(\omega)$ for $\omega \in \Omega$. If in addition $\lambda_j = \lambda_i$, then the field is said to be *homogeneous*. Roughly speaking, a homogeneous feature contributes the same weight to the distribution no matter where in the graph it appears. Homogeneous features arise naturally in the application of Section 5.

The methods that we describe in this paper apply to exponential models in general; that is, it is not essential that there is an underlying graph structure. However, it will be convenient to express our approach in terms of the random field models described above.

B. Two optimization problems

Suppose that we are given an initial model $q_0 \in \Delta$, a reference distribution $\tilde{p}$, and a set of features $f = (f_0, f_1, \ldots, f_n)$. In practice, it is often the case that $\tilde{p}$ is the empirical distribution of a set of training samples $\omega^{(1)}, \omega^{(2)} \ldots \omega^{(N)}$, and is thus given by

$$\tilde{p}(\omega) = \frac{c(\omega)}{N}$$

where $c(\omega) = \sum_{1 \le i \le N} \delta(\omega, \omega^{(i)})$ is the number of times that configuration ω appears among the training samples.

We wish to construct a probability distribution $q_\star \in \Delta$ that accounts for these data, in the sense that it approximates $\tilde{p}$ but does not deviate too far from q_0. We measure distance between probability distributions p and q in Δ using the Kullback-Leibler divergence

$$D(p \,\|\, q) = \sum_{\omega \in \Omega} p(\omega) \, \log \frac{p(\omega)}{q(\omega)} \,. \tag{1}$$

Throughout this paper we use the notation

$$p[g] = \sum_{\omega \in \Omega} g(\omega) \, p(\omega)$$

for the expectation of a function $g : \mathbf{\Omega} \to \mathbf{R}$ with respect to the probability distribution p. For a function $h : \mathbf{\Omega} \to \mathbf{R}$ and a distribution q, we use both the notation $h \circ q$ and q_h to denote the generalized Gibbs distribution given by

$$q_h(\omega) = (h \circ q)(\omega) = \frac{1}{Z_q(h)} e^{\,h(\omega)} \, q(\omega) \,.$$

Note that $Z_q(h)$ is not the usual partition function. It is a normalization constant determined by the requirement that $(h \circ q)(\omega)$ sums to 1 over ω, and can be written as an expectation:

$$Z_q(h) = q[e^{\,h}] \,.$$

There are two natural sets of probability distributions determined by the data $\tilde{p}$, q_0, and f. The first is the set $\mathcal{P}(f, \tilde{p})$ of all distributions that agree with $\tilde{p}$ as to the expected value of the feature function f:

$$\mathcal{P}(f, \tilde{p}) = \{ p \in \Delta : \; p[f] = \tilde{p}[f] \,\} \,.$$

The second is the set $\mathcal{Q}(f, q_0)$ of generalized Gibbs distributions based on q_0 with feature function f:

$$\mathcal{Q}(f, q_0) = \{ (\lambda \cdot f) \circ q_0 : \; \lambda \in \mathbf{R}^n \,\} \,.$$

We let $\bar{\mathcal{Q}}(f, q_0)$ denote the closure of $\mathcal{Q}(f, q_0)$ in Δ (with respect to the topology it inherits as a subset of Euclidean space).

There are two natural criteria for choosing an element $q_\star$ from these sets:

- *Maximum Likelihood Gibbs Distribution*. Choose $q_\star$ to be a distribution in $\bar{\mathcal{Q}}(f, q_0)$ with maximum likelihood with respect to $\tilde{p}$:

$$q_\star^{\text{ML}} = \mathop{\arg\min}_{q \in \bar{\mathcal{Q}}(f, q_0)} D(\tilde{p} \,\|\, q)$$

- *Maximum Entropy Constrained Distribution*. Choose $q_\star$ to be a distribution in $\mathcal{P}(f, \tilde{p})$ that has maximum entropy relative to q_0:

$$q_\star^{\text{ME}} = \mathop{\arg\min}_{p \in \mathcal{P}(f, \tilde{p})} D(p \,\|\, q_0)$$

Although these criteria are different, they determine the same distribution: $q_\star = q_\star^{\text{ML}} = q_\star^{\text{ME}}$. Moreover, this distribution is the unique element of the intersection $\mathcal{P}(f, \tilde{p}) \cap \bar{\mathcal{Q}}(f, q_0)$, as we discuss in detail in Section 4.1 and Appendix A.

When $\tilde{p}$ is the empirical distribution of a set of training examples $\omega^{(1)}, \omega^{(2)} \ldots \omega^{(N)}$, minimizing $D(\tilde{p} \,\|\, p)$ is equivalent to maximizing the probability that the field p assigns to the training data, given by

$$\prod_{1 \le i \le N} p(\omega^{(i)}) \;=\; \prod_{\omega \in \Omega} p(\omega)^{\,c(\omega)} \;\propto\; e^{-N D(\tilde{p} \,\|\, p)} \,.$$

With sufficiently many parameters it is a simple matter to construct a field for which $D(\tilde{p} \,\|\, p)$ is arbitrarily small. This is the classic problem of *over training*. The idea behind the method proposed in this paper is to incrementally construct a field that captures the salient properties of $\tilde{p}$ by incorporating an increasingly detailed collection of features, allowing generalization to new configurations; the resulting distributions are *not* absolutely continuous with respect to the empirical distribution of the training sample. The maximum entropy framework for parameter estimation tempers the over training problem; however, the basic problem remains, and is out of the scope of the present paper. We now present the random field induction paradigm.

C. Inducing field interactions

We begin by supposing that we have a set of *atomic* features

$$\mathcal{F}_{\text{atomic}} \subset \{ g : \mathbf{\Omega} \longrightarrow \{0, 1\}, \; \text{supp}(g) = v_g \in V \}$$

each of which is supported by a single vertex. We use atomic features to incrementally build up more complicated features. The following definition specifies how we shall allow a field to be incrementally constructed, or *induced*.

Definition 1: Suppose that the field q is given by $q = (\lambda \cdot f) \circ q_0$. The features f_i are called the *active* features of q. A feature g is a *candidate* for q if either $g \in \mathcal{F}_{\text{atomic}}$, or if g is of the form $g(\omega) = a(\omega) f_i(\omega)$ for an atomic feature a and an active feature f_i with supp$(g) \ominus$ supp$(f_i) \in E$. The set of candidate features of q is denoted $\mathcal{C}(q)$.
In other words, candidate features are obtained by conjoining atomic features with existing features. The condition on supports ensures that each feature is supported by a path-connected subset of G.

If $g \in \mathcal{C}(q)$ is a candidate feature of q, then we call the 1-parameter family of random fields $q_{\alpha g} = (\alpha g) \circ q$ the *induction of q by g*. We also define

$$G_q(\alpha, g) = D(\tilde{p} \,\|\, q) - D(\tilde{p} \,\|\, q_{\alpha g}) \,. \tag{2}$$

We think of $G_q(\alpha, g)$ as the improvement that feature g brings to the model when it has weight α. As we show in the following section, $G_q(\alpha, g)$ is $\cap$-convex in α. (We use the suggestive notation $\cap$-convex and $\cup$-convex in place of the less mnemonic concave and convex terminology.) We define $G_q(g)$ to be the greatest improvement that feature g can give to the model while keeping all of the other features' parameters fixed:

$$G_q(g) = \sup_\alpha G_q(\alpha, g) \,.$$

We refer to $G_q(g)$ as the *gain* of the candidate g.

D. Incremental construction of random fields

We can now describe our algorithm for incrementally constructing fields.

Field Induction Algorithm.

Initial Data:

A reference distribution $\tilde{p}$ and an initial model q_0.

Output:

A field $q_\star$ with active features $f_0, \ldots, f_N$ such that $q_\star = \arg\min_{q \in \bar{\mathcal{Q}}(f, q_0)} D(\tilde{p} \,\|\, q)$.

Algorithm:

(0) Set $q^{(0)} = q_0$.

(1) For each candidate $g \in \mathcal{C}(q^{(n)})$ compute the gain $G_{q^{(n)}}(g)$.

(2) Let $f_n = \arg\max_{g \in \mathcal{C}(q^{(n)})} G_{q^{(n)}}(g)$ be the feature with the largest gain.

(3) Compute $q_\star = \arg\min_{q \in \bar{\mathcal{Q}}(f, q_0)} D(\tilde{p} \,\|\, q)$, where $f = (f_0, f_1, \ldots, f_n)$.

(4) Set $q^{(n+1)} = q_\star$ and $n \leftarrow n + 1$, and go to step (1).

This induction algorithm has two parts: *feature selection* and *parameter estimation*. Feature selection is carried out in steps (1) and (2), where the feature yielding the largest gain is incorporated into the model. Parameter estimation is carried out in step (3), where the parameters are adjusted to best represent the reference distribution. These two computations are discussed in more detail in the following two sections.

III. Feature Selection

The feature selection step of our induction algorithm is based upon an approximation. We approximate the improvement due to adding a single candidate feature, measured by the reduction in Kullback-Leibler divergence, by adjusting only the weight of the candidate and keeping all of the other parameters of the field fixed. In general this is only an estimate, since it may well be that adding a feature will require significant adjustments to all of the parameters in the new model. From a computational perspective, approximating the improvement in this way can enable the simultaneous evaluation of thousands of candidate features, and makes the algorithm practical. In this section we present explain the feature selection step in detail.

Proposition 1: Let $G_q(\alpha, g)$, defined in (2), be the approximate improvement obtained by adding feature g with parameter α to the field q. Then if g is not constant, $G_q(\alpha, g)$ is strictly $\cap$-convex in α and attains its maximum at the unique point $\hat{\alpha}$ satisfying

$$\tilde{p}[g] = q_{\hat{\alpha} g}[g] \,. \tag{3}$$

Proof: Using the definition (1) of the Kullback-Leibler divergence we can write

$$\begin{aligned} G_q(\alpha, g) &= \sum_{\omega \in \Omega} \tilde{p}(\omega) \log \frac{Z_q^{-1}(\alpha g)\, e^{\alpha g(\omega)} q(\omega)}{q(\omega)} \\ &= \sum_{\omega \in \Omega} \tilde{p}(\omega) \left(\alpha g(\omega) - \log q\left[e^{\alpha g}\right] \right) \\ &= \alpha \tilde{p}[\,g\,] - \log q\left[e^{\alpha g}\right] \,. \end{aligned}$$

Thus

$$\begin{aligned} \frac{\partial}{\partial \alpha} G_q(\alpha, g) &= \tilde{p}[g] - \frac{q[g e^{\alpha g}]}{q[e^{\alpha g}]} \\ &= \tilde{p}[g] - q_{\alpha g}[g] \,. \end{aligned}$$

Moreover,

$$\begin{aligned} \frac{\partial^2}{\partial \alpha^2} G_q(\alpha, g) &= \frac{q[g e^{\alpha g}]^2}{q[e^{\alpha g}]^2} - \frac{q[g^2 e^{\alpha g}]}{q[e^{\alpha g}]} \\ &= -q_{\alpha g}\left[\left(g - q_{\alpha g}[g]\right)^2\right] \end{aligned}$$

Hence, $\frac{\partial^2}{\partial \alpha^2} G_q(\alpha, g) \leq 0$, so that $G_q(\alpha, g)$ is $\cap$-convex in α. If g is not constant, then $\frac{\partial^2}{\partial \alpha^2} G_q(\alpha, g)$, which is minus the variance of g with respect to $q_{\alpha g}$, is strictly negative, so that $G_q(\alpha, g)$ is strictly convex. □

When g is binary-valued, its gain can be expressed in a particularly nice form. This is stated in the following proposition, whose proof is a simple calculation.

Proposition 2: Suppose that the candidate g is binary-valued. Then $G_q(\alpha, g)$ is maximized at

$$\hat{\alpha} = \log \left(\frac{\tilde{p}[\,g\,](1 - q[\,g\,])}{q[\,g\,](1 - \tilde{p}[\,g\,])} \right)$$

and at this value,

$$G_q(g) = G_q(\hat{\alpha}, g) = D(B_p \,\|\, B_q)$$

where B_p and B_q are Bernoulli random variables given by

$$B_p(1) = \tilde{p}[\,g\,] \qquad B_p(0) = 1 - \tilde{p}[\,g\,]$$
$$B_q(1) = q[\,g\,] \qquad B_q(0) = 1 - q[\,g\,]\,.$$

For features that are not binary-valued, but instead take values in the non-negative integers, the parameter $\hat{\alpha}$ that solves (3) and thus maximizes $G_q(\alpha, g)$ cannot, in general, be determined in closed form. This is the case for tied binary features, and it applies to the application we describe in Section 5. For these cases it is convenient to rewrite (3) slightly. Let $\beta = e^{\alpha}$ so that $\partial/\partial\alpha = \beta\partial/\partial\beta$. Let

$$g_k = \sum_{\omega} q(\omega)\,\delta(k, g(\omega))$$

be the total probability assigned to the event that the feature g takes the value k. Then (3) becomes

$$\beta \frac{\partial}{\partial \beta} G_q(\log \beta, g) = \tilde{p}[g] - \frac{\sum_{k=0}^{N} k\, g_k \beta^k}{\sum_{k=0}^{N} g_k \beta^k} = 0 \qquad (4)$$

This equation lends itself well to numerical solution. The general shape of the curve $\beta \mapsto \beta\partial/\partial\beta\;G_q(\log\beta, g)$ is shown in Figure 1.

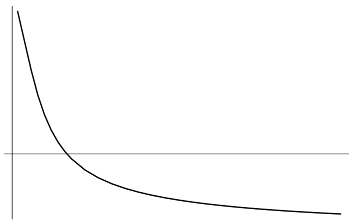

Fig. 1. Derivative of the gain

The limiting value of $\beta\partial G_q(\log\beta, g)/\partial\beta$ as $\beta \to \infty$ is $\tilde{p}[g] - N$. The solution to equation (4) can be found using Newton's method, which in practice converges rapidly for such functions.

When the configuration space Ω is large, so that the coefficients g_k cannot be calculated by summing over all configurations, Monte Carlo techniques may be used to estimate them. It is important to emphasize that the *same* set of random configurations can be used to estimate the coefficients g_k for each candidate g simultaneously. Rather than discuss the details of Monte Carlo techniques for this problem we refer to the extensive literature on this topic. We have obtained good results using the standard technique of Gibbs sampling [17] for the problem we describe in Section 5.

IV. Parameter Estimation

In this section we present an algorithm for selecting the parameters associated with the features of a random field. The algorithm is a generalization of the Generalized Iterative Scaling algorithm of Darroch and Ratcliff [12]. It reduces to the Darroch-Ratcliff algorithm when the features sum to a constant; however, the new algorithm does not make this restriction.

Throughout this section we hold the set of features $f = (f_0, f_1, \ldots, f_n)$, the initial model q_0 and the reference distribution $\tilde{p}$ fixed, and we simplify the notation accordingly. In particular, we write $\gamma \circ q$ instead of $(\gamma \cdot f) \circ q$ for $\gamma \in \mathbf{R}^n$. We assume that $\tilde{p}(\omega) = 0$ whenever $q_0(\omega) = 0$. This condition is commonly written $\tilde{p} \ll q_0$, and it is equivalent to $D(\tilde{p} \,\|\, q_0) < \infty$.

A description of the algorithm requires an additional piece of notation. Let

$$f_{\#}(\omega) = \sum_{i=0}^{n} f_i(\omega)\,.$$

If the features are binary, then $f_{\#}(\omega)$ is the total number of features that are "on" for the configuration ω.

Improved Iterative Scaling.

Initial Data:

A reference distribution $\tilde{p}$ and an initial model q_0, with $\tilde{p} \ll q_0$, and non-negative features $f_0, f_1, \ldots, f_n$.

Output:

The distribution $q_\star = \underset{q \in \bar{\mathcal{Q}}(f, q_0)}{\arg\min}\; D(\tilde{p} \,\|\, q)$

Algorithm:

(0) Set $q^{(0)} = q_0$.

(1) For each i let $\gamma_i^{(k)} \in [-\infty, \infty)$ be the unique solution of

$$q^{(k)}[\, f_i\, e^{\gamma_i^{(k)} f_{\#}}\,] = \tilde{p}[\, f_i\,]\,. \qquad (5)$$

(2) Set $q^{(k+1)} = \gamma^{(k)} \circ q^{(k)}$ and $k \leftarrow k+1$.

(3) If $q^{(k)}$ has converged, set $q_\star = q^{(k)}$ and terminate. Otherwise go to step (1).

In other words, this algorithm constructs a distribution $q_\star = \lim_{m\to\infty} \gamma_m \circ q_0$ where $\gamma_m = \sum_{k=0}^{m} \gamma^{(k)}$ and $\gamma_i^{(k)}$ is determined as the solution to the equation

$$\sum_{\omega} q^{(k)}(\omega)\, f_i(\omega)\, e^{\gamma_i^{(k)} f_{\#}(\omega)} = \sum_{\omega} \tilde{p}(\omega)\, f_i(\omega)\,.$$

When used in the n-th iteration of the field induction algorithm, where a candidate feature $g = f_n$ is added to the field $q = q_n$, we choose the initial distribution q_0 to be $q_0 = q_{\hat{\alpha} g}$, where $\hat{\alpha}$ is the parameter that maximizes the gain of g. In practice, this provides a good starting point from which to begin iterative scaling. In fact, we can view this distribution as the result of applying one iteration of an Iterative Proportional Fitting Procedure [5], [9] to project $q_{\alpha g}$ onto the linear family of distributions with g-marginals constrained to $\tilde{p}[g]$.

Our main result in this section is

Proposition 3: Suppose $q^{(k)}$ is the sequence in Δ determined by the Improved Iterative Scaling algorithm. Then $D(\tilde{p} \,\|\, q^{(k)})$ decreases monotonically to $D(\tilde{p} \,\|\, q_\star)$ and $q^{(k)}$ converges to $q_\star = \underset{q \in \bar{\mathcal{Q}}}{\arg\min}\; D(\tilde{p} \,\|\, q) = \underset{p \in \mathcal{P}}{\arg\min}\; D(p \,\|\, q_0)$.

In the remainder of this section we present a self-contained proof of the convergence of the algorithm. The key idea of the proof is to express the incremental step of the algorithm in terms of an auxiliary function which bounds from below the log-likelihood objective function. This technique is the standard means of analyzing the EM algorithm [13], but it has not previously been applied to iterative scaling. Our analysis of iterative scaling is different and simpler than previous treatments. In particular, in contrast to Csiszár's proof of the Darroch-Ratcliff procedure [10], our proof does not rely upon the convergence of alternating I-projection [9].

We begin by formulating the basic duality theorem which states that the maximum likelihood problem for a Gibbs distribution and the maximum entropy problem subject to linear constraints have the same solution. We then turn to the task of computing this solution. After introducing auxiliary functions in a general setting, we apply this method to prove convergence of the Improved Iterative Scaling algorithm. We finish the section by discussing Monte Carlo methods for estimating the equations when the size of the configuration space prevents the explicit calculation of feature expectations.

A. Duality

The duality between the maximum likelihood and maximum entropy problems is expressed in the following Proposition.

Proposition 4: Suppose that $\tilde{p} \ll q_0$. Then there exists a unique $q_\star \in \Delta$ satisfying

(1) $q_\star \in \mathcal{P} \cap \bar{\mathcal{Q}}$
(2) $D(p \,||\, q) = D(p \,||\, q_\star) + D(q_\star \,||\, q)$ for any $p \in \mathcal{P}$ and $q \in \bar{\mathcal{Q}}$
(3) $q_\star = \arg\min_{q \in \bar{\mathcal{Q}}} D(\tilde{p} \,||\, q)$
(4) $q_\star = \arg\min_{p \in \mathcal{P}} D(p \,||\, q_0)$.

Moreover, any of these four properties determines $q_\star$ uniquely.

This result is well known, although perhaps not quite in this packaging. In the language of constrained optimization, it expresses the fact that the maximum likelihood problem for Gibbs distributions is the convex dual to the maximum entropy problem for linear constraints. Property (2) is called the *Pythagorean property* since it resembles the Pythagorean theorem if we imagine that $D(p \,||\, q)$ is the square of Euclidean distance and $(p, q_\star, q)$ are the vertices of a right triangle.

We include a proof of this result in Appendix A to make this paper self-contained and also to carefully address the technical issues arising from the fact that $\mathcal{Q}$ is not closed. The proposition would not be true if we replaced $\bar{\mathcal{Q}}$ with $\mathcal{Q}$; in fact, $\mathcal{P} \cap \mathcal{Q}$ might be empty. Our proof is elementary and does not rely on the Kuhn-Tucker theorem or other machinery of constrained optimization.

B. Auxiliary functions

We now turn to the task of computing $q_\star$. Fix $\tilde{p}$ and let $L : \Delta \to \mathbf{R}$ be the log-likelihood objective function

$$L(q) = -D(\tilde{p} \,||\, q) .$$

Definition 2: A function $A : \mathbf{R}^n \times \Delta \to R$ is an *auxiliary function for* L if

(1) For all $q \in \Delta$ and $\gamma \in \mathbf{R}^n$

$$L(\gamma \circ q) \geq L(q) + A(\gamma, q)$$

(2) $A(\gamma, q)$ is continuous in $q \in \Delta$ and C^1 in $\gamma \in \mathbf{R}^n$ with $A(0, q) = 0$ and

$$\frac{d}{dt}\Big|_{t=0} A(t\gamma, q) = \frac{d}{dt}\Big|_{t=0} L((t\gamma) \circ q) .$$

We can use an auxiliary function A to construct an iterative algorithm for maximizing L. We start with $q^{(k)} = q_0$ and recursively define $q^{(k+1)}$ by

$$q^{(k+1)} = \gamma^{(k)} \circ q^{(k)} \quad \text{with} \quad \gamma^{(k)} = \arg\max_{\gamma} A(\gamma, q^{(k)}) .$$

It is clear from property (1) of the definition that each step of this procedure increases L. The following proposition implies that in fact the sequence $q^{(k)}$ will reach the maximum of L.

Proposition 5: Suppose $q^{(k)}$ is any sequence in Δ with

$$q^{(0)} = q_0 \quad \text{and} \quad q^{(k+1)} = \gamma^{(k)} \circ q^{(k)}$$

where $\gamma^{(k)} \in \mathbf{R}^n$ satisfies

$$A(\gamma^{(k)}, q^{(k)}) = \sup_{\gamma} A(\gamma, q^{(k)}) . \tag{6}$$

Then $L(q^{(k)})$ increases monotonically to $\max_{q \in \bar{\mathcal{Q}}} L(q)$ and $q^{(k)}$ converges to $q_\star = \arg\max_{q \in \bar{\mathcal{Q}}} L(q)$.

Equation (6) assumes that the supremum $\sup_\gamma A(\gamma, q^{(k)})$ is achieved at finite γ. In Appendix B, under slightly stronger assumptions, we present an extension that allows some components of $\gamma^{(k)}$ to take the value $-\infty$.

To use the proposition to construct a practical algorithm we must determine an auxiliary function $A(\gamma, q)$ for which $\gamma^{(k)}$ satisfying the required condition can be determined efficiently. In Section 4.3 we present a choice of auxiliary function which yields the Improved Iterative Scaling updates.

To prove Proposition 5 we first prove three lemmas.

Lemma 1: If $m \in \Delta$ is a cluster point of $q^{(k)}$, then $A(\gamma, m) \leq 0$ for all $\gamma \in \mathbf{R}^n$.

Proof: Let $q^{(k_l)}$ be a sub-sequence converging to m. Then for any γ

$$\begin{aligned} A(\gamma, q^{(k_l)}) &\leq A(\gamma^{(k_l)}, q^{(k_l)}) \leq L(q^{(k_l+1)}) - L(q^{(k_l)}) \\ &\leq L(q^{(k_{l+1})}) - L(q^{(k_l)}) . \end{aligned}$$

The first inequality follows from property (6) of $\gamma^{(n_k)}$. The second and third inequalities are a consequence of the monotonicity of $L(q^{(k)})$. The lemma follows by taking limits and using the fact that L and A are continuous. □

Lemma 2: If $m \in \Delta$ is a cluster point of $q^{(k)}$, then $\frac{d}{dt}|_{t=0} L((t\gamma) \circ m) = 0$ for any $\gamma \in \mathbf{R}^n$.

Proof: By the previous lemma, $A(\gamma, m) \leq 0$ for all γ. Since $A(0, m) = 0$, this means that $\gamma = 0$ is a maximum of $A(\gamma, m)$ so that

$$0 = \frac{d}{dt}\Big|_{t=0} A(t\gamma, m) = \frac{d}{dt}\Big|_{t=0} L((t\gamma) \circ m) .$$

□

Lemma 3: Suppose $\{q^{(k)}\}$ is any sequence with only one cluster point q_*. Then $q^{(k)}$ converges to q_*.

Proof: Suppose not. Then there exists an open set B containing q_* and a subsequence $q^{(n_k)} \notin B$. Since Δ is compact, $q^{(n_k)}$ has a cluster point $q'_* \notin B$. This contradicts the assumption that $\{q^{(k)}\}$ has a unique cluster point. □

Proof of Proposition 5: Suppose that m is a cluster point of $q^{(k)}$. Then it follows from Lemma 2 that $\frac{d}{dt}|_{t=0}\, L((t\gamma)\circ m) = 0$, and so $m \in \mathcal{P} \cap \bar{\mathcal{Q}}$ by Lemma 2 of Appendix A. But $q_\star$ is the only point in $\mathcal{P} \cap \bar{\mathcal{Q}}$ by Proposition 4. It follows from Lemma 3 that $q^{(k)}$ converges to $q_\star$. □

In Appendix B we prove an extension of Proposition 5 that allows the components of γ to equal $-\infty$. For this extension, we assume that all the components of the feature function f are non-negative:

$$f_i(\omega) \geq 0 \quad \text{for all } i \text{ and all } \omega. \tag{7}$$

This is not a practical restriction since we can replace f_i by $f_i - \min_\omega f_i(\omega)$.

C. Improved Iterative Scaling

We now prove the monotonicity and convergence of the Improved Iterative Scaling algorithm by applying Proposition 5 to a particular choice of auxiliary function. We now assume that each component of the feature function f is non-negative.

For $q \in \Delta$ and $\gamma \in \mathbf{R}^n$, define

$$A(\gamma, q) = 1 + \gamma \cdot \tilde{p}[f] - \sum_\omega q(\omega) \sum_i f(i\,|\,\omega)\, e^{\gamma_i f_\#(\omega)}$$

where $f(i\,|\,\omega) = \frac{f_i(\omega)}{f_\#(\omega)}$. It is easy to check that A extends to a continuous function on $(R \cup -\infty)^n \times \Delta$.

Lemma 4: $A(\gamma, q)$ is an extended auxiliary function for $L(q)$.

The key ingredient in the proof of the lemma is the $\cap$-convexity of the logarithm and the $\cup$-convexity of the exponential, as expressed in the inequalities

$$e^{\sum_i t_i \alpha_i} \leq \sum_i t_i\, e^{\alpha_i} \quad \text{if } t_i \geq 0, \textstyle\sum_i t_i = 1 \tag{8}$$

$$\log x \leq x - 1 \quad \text{for all } x > 0. \tag{9}$$

Proof of Lemma 4: Because A extends to a continuous function on $(R \cup -\infty)^n \times \Delta$, it suffices to prove that it satisfies properties (1) and (2) of Definition 2. To prove property (1) note that

$$\begin{aligned} L(\gamma \circ q) - L(q) &= \gamma \cdot \tilde{p}[f] - \log \sum_\omega q(\omega)\, e^{\gamma \cdot f(\omega)} &(10)\\ &\geq \gamma \cdot \tilde{p}[f] + 1 - \sum_\omega q(\omega)\, e^{\gamma \cdot f(\omega)} &(11)\\ &\geq \gamma \cdot \tilde{p}[f] + 1 - \sum_\omega q(\omega) \sum_i f(i\,|\,\omega)\, e^{\gamma_i f_\#(\omega)} &(12)\\ &= A(\gamma, q)\,. &(13) \end{aligned}$$

Equality (10) is a simple calculation. Inequality (11) follows from inequality (9). Inequality (12) follows from the definition of $f_\#$ and Jensen's inequality (8). Property (2) of Definition 2 is straightforward to verify. □

Proposition 3 follows immediately from the above lemma and the extended Proposition 5. Indeed, it is easy to check that $\gamma^{(k)}$ defined in Proposition 3 achieves the maximum of $A(\gamma, q^{(k)})$, so that it satisfies the condition of Proposition 5 in Appendix B.

D. Monte Carlo methods

The Improved Iterative Scaling algorithm described in the previous section is well-suited to numerical techniques since all of the features take non-negative values. In each iteration of this algorithm it is necessary to solve a polynomial equation for each feature f_i. That is, we can express equation 5 in the form

$$\sum_{m=0}^{M} a_{m,i}^{(k)}\, \beta_i^m = 0$$

where M is the largest value of $f_\#(\omega) = \sum_i f_i(\omega)$ and

$$a_{m,i}^{(k)} = \begin{cases} \sum_\omega q^{(k)}(\omega)\, f_i(\omega)\, \delta(m, f_\#(\omega)) & m > 0 \\ -\tilde{p}[\,f_i\,] & m = 0 \end{cases} \tag{14}$$

where $q^{(k)}$ is the field for the k-th iteration and $\beta_i = e^{\gamma_i^{(k)}}$. This equation has no solution precisely when $a_{m,i}^{(k)} = 0$ for $m > 0$. Otherwise, it can be efficiently solved using Newton's method since all of the coefficients $a_{m,i}^{(k)}$, $m > 0$, are non-negative. When Monte Carlo methods are to be used because the configuration space Ω is large, the coefficients $a_{m,i}^{(k)}$ can be simultaneously estimated for all i and m by generating a single set of samples from the distribution $q^{(k)}$.

V. Application: Word Morphology

Word clustering algorithms are useful for many natural language processing tasks. One such algorithm [6], called mutual information clustering, is based upon the construction of simple bigram language models using the maximum likelihood criterion. The algorithm gives a hierarchical binary classification of words that has been used for a variety of purposes, including the construction of decision tree language and parsing models, and sense disambiguation for machine translation [7].

A fundamental shortcoming of the mutual information word clustering algorithm given in [6] is that it takes as fundamental the word spellings themselves. This increases the severity of the problem of small counts that is present in virtually every statistical learning algorithm. For example, the word "Hamiltonianism" appears only once in the 365,893,263-word corpus used to collect bigrams for the clustering experiments described in [6]. Clearly this is insufficient evidence on which to base a statistical clustering decision. The basic motivation behind the feature-based approach is that by querying features of spellings, a clustering algorithm could notice that such a word begins with a capital letter, ends in "ism" or contains "ian," and profit from how these features are used for other words in similar contexts.

In this section we describe how we applied the random field induction algorithm to discover morphological features of words, and we present sample results. This application demonstrates how our technique gradually sharpens the probability mass from the enormous set of all possible configurations, in this case ASCII strings, onto a set of configurations that is increasingly similar to those in the training sample. It achieves this by introducing both "positive" features which many of the training samples exhibit, as well as "negative" features which do not appear in the sample, or appear only rarely. A description of how the resulting features were used to improve mutual information clustering is given in [20], and is beyond the scope of the present paper; we refer the reader to [6], [20] for a more detailed treatment of this topic.

In Section 5.1 we formulate the problem in terms of the notation and results of Sections 2, 3, and 4. In Section 5.2 we describe how the field induction algorithm is actually carried out in this application. In Section 5.3 we explain the results of the induction algorithm by presenting a series of examples.

A. *Problem formulation*

To discover features of spellings we take as configuration space the set of all strings $\Omega = \mathcal{A}^*$ in the ASCII alphabet $\mathcal{A}$. We construct a probability distribution $p(\omega)$ on Ω by first predicting the length $|\,\omega\,|$, and then predicting the actual spelling; thus, $p(\omega) = p_l(|\,\omega\,|)p_s(\omega\,|\,|\,\omega\,|)$ where p_l is the length distribution and p_s is the spelling distribution. We take the length distribution as given. We model the spelling distribution $p_s(\cdot\,|\,l)$ over strings of length l as a random field. Let Ω_l be the configuration space of all ASCII strings of length l. Then $|\,\Omega_l\,| = O(10^{2l})$ since each ω_i is an ASCII character.

To reduce the number of parameters, we tie features, as described in Section 2.1, so that a feature has the same weight independent of where it appears in the string. Because of this it is natural to view the graph underlying Ω_l as a regular l-gon. The group of automorphisms of this graph is the set of all rotations, and the resulting field is homogeneous as defined in Section 2.

Not only is each field p_s homogeneous, but in addition, we tie features across fields for different values of l. Thus, the weight λ_f of a feature is independent of l. To introduce a dependence on the length, as well as on whether or not a feature applies at the beginning or end of a string, we adopt the following artificial construction. We take as the graph of Ω_l an $(l+1)$-gon rather than an l-gon, and label a distinguished vertex by the length, keeping this label held fixed.

To complete the description of the fields that are induced, we need to specify the set of atomic features. The atomic features that we allow fall into three types. The first type is the class of features of the form

$$f_{v,c}(\omega) = \begin{cases} 1 & \text{if } \omega_v = c \\ 0 & \text{otherwise.} \end{cases}$$

where c is any ASCII character, and v denotes an arbitrary character position in the string. The second type of atomic features involve the special vertex <l> that carries the length of the string. These are the features

$$\begin{aligned} f_{v,l}(\omega) &= \begin{cases} 1 & \text{if } \omega_v = \text{<}l\text{>} \\ 0 & \text{otherwise} \end{cases} \\ f_{v,\text{<>}}(\omega) &= \begin{cases} 1 & \text{if } \omega_v = \text{<}l\text{>} \text{ for some } l \\ 0 & \text{otherwise} \end{cases} \end{aligned}$$

The atomic feature $f_{v,\text{<>}}$ introduces a dependence on whether a string of characters lies at the beginning or end of the string, and the atomic features $f_{v,l}$ introduce a dependence on the length of the string. To tie together the length dependence for long strings, we also introduce an atomic feature $f_{v,7+}$ for strings of length 7 or greater.

The final type of atomic feature asks whether a character lies in one of four sets, `[a-z]`, `[A-Z]`, `[0-9]`, `[@-&]`, denoting arbitrary lowercase letters, uppercase letters, digits, and punctuation. For example, the atomic feature

$$f_{v,\texttt{[a-z]}}(\omega) = \begin{cases} 1 & \text{if } \omega_v \in \texttt{[a-z]} \\ 0 & \text{otherwise} \end{cases}$$

tests whether or not a character is lowercase.

To illustrate the notation that we use, let us suppose that the the following features are active for a field: "ends in `ism`," "a string of at least 7 characters beginning with a capital letter" and "contains `ian`." Then the probability of the word "Hamiltonianism" would be given as

$$\begin{aligned} & p_l(14)\, p_s(\texttt{Hamiltonianism}\,|\,|\,\omega\,| = 14) = \\ & \quad p_l(14)\,\frac{1}{Z_{14}}\, e^{\lambda_{\texttt{7+<[A-Z]}} + \lambda_{\texttt{ian}} + \lambda_{\texttt{ism>}}}\,. \end{aligned}$$

Here the λ's are the parameters of the appropriate features, and we use the characters < and > to denote the beginning and ending of a string (more common regular expression notation would be ^ and $). The notation `7+<[A-Z]` thus means "a string of at least 7 characters that begins with a capital letter," corresponding to the feature

$$f_{u,\texttt{7+}}\, f_{v,\texttt{[A-Z]}}\,,$$

where u and v are *adjacent* positions in the string, recalling from Definition 2.1 that we require the support of a feature to be a connected subgraph. Similarly, `ism>` means "ends in -ism" and corresponds to the feature

$$f_{u,\texttt{i}}\, f_{v,\texttt{s}}\, f_{w,\texttt{m}}\, f_{x,\texttt{<>}}$$

where u, v, w, x are adjacent positions in the string and `ian` means "contains ian," corresponding to the feature

$$f_{u,\texttt{i}}\, f_{v,\texttt{a}}\, f_{w,\texttt{n}}\,.$$

B. *Description of the algorithm*

We begin the random field induction algorithm with a model that assigns uniform probability to all strings. We then incrementally add features to a random field model in order to minimize the Kullback-Leibler divergence between the field and the unigram distribution of the vocabulary obtained from a training corpus. The length distribution is taken according to the lengths of words in the empirical distribution of the training data. The improvement to the model made by a candidate feature is evaluated by the reduction in relative entropy, with respect to the unigram distribution, that adding the new feature yields, keeping the other parameters of the model fixed. Our learning algorithm incrementally constructs a random field to describe those features of spellings that are most informative.

At each stage in the induction algorithm, a set of candidate features is constructed. Because the fields are homogeneous, the set of candidate features can be viewed as follows. Each active feature can be expressed in the form

$$f_s(\omega) = \begin{cases} 1 & \text{substring } s \text{ appears in } \omega \\ 0 & \text{otherwise} \end{cases}$$

where s is a string in the extended alphabet $\mathcal{A}$ of ASCII characters together with the macros `[a-z]`, `[A-Z]`, `[0-9]`, `[@-&]`, and the length labels `<l>` and `<>`. If $\{f_s\}_{s\in\mathcal{S}}$ is the set of active features, (including $s = \epsilon$, the empty string) using this representation, then the set of candidate features is precisely the set $\{f_{a\cdot s}, f_{s\cdot a}\}_{a\in\mathcal{A}, s\in\mathcal{S}}$, where $a \cdot s$ denotes concatenation of strings. As required by Definition 2, each such candidate increases the support of an active feature by a single adjacent vertex.

Since the model assigns probability to arbitrary word strings, the partition function Z_l can be computed exactly for only the smallest string lengths l. We therefore compute feature expectations using a random sampling algorithm. Specifically, we use the Gibbs sampler to generate 10,000 spellings of random lengths. When computing the gain $G_q(g)$ of a candidate feature, we use these spellings to estimate the probability g_k that the candidate feature g occurs k times in a spelling (see equation (4)–for example, the feature $f_{v,\texttt{[a-z]}}$ occurs two times in the string `The`), and then solve for the corresponding β using Newton's method for *each* candidate feature. It should be emphasized that only a single set of random spellings needs to be generated; the same set can be used to estimate g_k for each candidate g. After adding the best candidate to the field, all of the feature weights are readjusted using the Improved Iterative Scaling algorithm. To carry out this algorithm, random spellings are again generated, this time incorporating the new feature, yielding Monte Carlo estimates of the coefficients $a^{(k)}_{m,i}$. Recall that $a^{(k)}_{m,i}$ is the expected number of times that feature i appears (under the substring representation for homogeneous features) in a string for which there is a total of m active features (see equation 14)). Given estimates for these coefficients, Newton's method is again used to solve equation (14), to complete a single iteration of the iterative scaling algorithm. After convergence of the Kullback-Leibler divergence, the inductive step is complete, and a new set of candidate features is considered.

C. Sample results

We began with a uniform field, that is, a field with no features at all. For this field, all ASCII strings of a given length are equally likely, and the lengths are drawn from a fixed distribution. Here is a sample of strings drawn from this distribution:

```
~, mo, _!ZP*@, m/TLL, ks;cm_3, *LQdR, D, aW{,
5&TL|4, tc, ?!@, sNeiO+, wHo8zBr", pQlV, m, H!&,
h9, #Os, :, Ky}FM?, LW, ",8}, 89Lj, -P, A, !, H, `,
Y^:Du:, 1xCl, 1!'J#F*u., w=idHnM), ~, 2, 2leW2,
I,bw~tk1, 3", ], ], b, +JEmj6, +E*, \qjqe"-7f, |al2,
T, ~(sOc1+2ADe, &, \p9oH, i;, $6, q}O+[, xEv, #U,
O)[83COF, =|B|7%cR, Mqq, ?!mv, n=7G, $i9GAJ D, 5,
,=, +u6@I9:, +, =D, 2E#vz@3-, ~nu;.+s, 3xJ, GDWeqL,
R,3R, !7v, FX,@y, 4p_cY2hU, ~
```

It comes as no surprise that the first feature the induction algorithm chooses is `[a-z]`; it simply observes that characters should be lowercase. The maximum likelihood (maximum entropy) weight for this feature is $\beta = e^{\lambda} \approx 6.99$. This means that a string with a lowercase letter in some position is about 7 times more likely than the same string without a lowercase letter in that position.

When we now draw strings from the new distribution (using annealing to concentrate the distribution on the more probable strings), we obtain spellings that are primarily made up of lowercase letters, but that certainly do not resemble English words:

```
m, r, xevo, ijjiir, b, to, jz, gsr, wq, vf, x, ga,
msmGh, pcp, d, oziVlal, hzagh, yzop, io, advzmxnv,
ijv_bolft, x, emx, kayerf, mlj, rawzyb, jp, ag,
ctdnnnbg, wgdw, t, kguv, cy, spxcq, uzflbbf,
dxtkkn, cxwx, jpd, ztzh, lv, zhpkvnu, l^, r, qee,
nynrx, atze4n, ik, se, w, lrh, hp+, yrqyka'h,
zcngotcnx, igcump, zjcjs, lqpWiqu, cefmfhc, o, lb,
fdcY, tzby, yopxmvk, by, fz,, t, govyccm,
ijyiduwfzo, 6xr, duh, ejv, pk, pjw, l, fl, w
```

In the following table we show the first 10 features that the algorithm induced, together with their associated parameters. Several things are worth noticing. The second feature chosen was `[a-z][a-z]`, which denotes adjacent lowercase characters. The third feature added was the letter `e`, which is the most common letter. The weight for this feature is $\beta = e^{\lambda} = 3.47$. The next feature introduces the first dependence on the length of the string: `[a-z]>1` denotes the feature "a one character word ending with a lowercase letter." Notice that this feature has a small weight of 0.04, corresponding to our intuition that such words are uncommon. Similarly, the features `z`, `q`, `j`, and `x` are uncommon, and thus receive small weights. The appearance of the feature `*` is explained by the fact that the vocabulary for our corpus is restricted to the most frequent 100,000 spellings, and all other words receive the "unknown word" spelling `***`, which is rather frequent. (The "end-of-sentence" marker, which makes its appearance later, is given the spelling `|`.)

feature	[a-z]	[a-z][a-z]	e	[a-z]>1	t
β	6.64	6.07	3.47	0.04	2.75
feature	*	z	q	j	x
β	17.25	0.02	0.03	0.02	0.06

Shown below are spellings obtained by Gibbs sampling from the resulting collection of fields.

```
frk, et, egeit, edet, eutdmeeet, ppge, A, dtgd,
falawe, etci, eese, ye, epemtbn, tegoeed, ee, *mp,
temou, enrteunt, ore, erveelew, heyu, rht, *,
lkaeu, lutoee, tee, mmo, eobwtit, weethtw, 7, ee,
teet, gre, /, *, eeeteetue, hgtte, om, he, *,
stmenu, ec, ter, eedgtue, iu, ec, reett, *,
ivtcmeee, vt, eets, tidpt, lttv, *, etttvti, ecte,
X, see, *, pi, rlet, tt, *, eot, leef, ke, *, *,
tet, iwteeiwbeie, yeee, et, etf, *, ov
```

After inducing 100 features, the model finally begins to be concentrated on spellings that resemble actual words to some extent, particularly for short strings. At this point the algorithm has discovered, for example, that `the` is a very common 3-letter word, that many words end in `ed`, and that long words often end in `ion`. A sample of 10 of the first 100 features induced, with their appropriate weights is shown in the table below.

.	,>1	3<the	tion	4<th	y>	ed>	ion>7+	ent	7+<c
22.36	31.30	11.05	5.89	4.78	5.35	4.20	4.83	5.17	5.37

```
thed, and, thed, toftion, |, ieention, cention, |,
ceetion, ant, is, seieeet, cinention, and, .,
tloned, uointe, feredten, iined, sonention,
inathed, other, the, id, and, ,, of, is, of, of, ,,
lcers, ,, ceeecion, ,, roferented, |, ioner, ,, |,
the, the, the, centention, ionent, asers, ,,
ctention, |, of, thed, of, uentie, of, and, ttentt,
in, rerey, and, |, sotth, cheent, is, and, of,
thed, rontion, that, seoftr
```

A sample of the first 1000 features induced is shown in the table below, together with randomly generated spellings. Notice, for example, that the feature `[0-9][0-9]` appears with a surprisingly high weight of 9382.93. This is due to the fact that if a string contains one digit, then it's very likely to contain two digits. But since digits are relatively rare in general, the feature `[0-9]` is assigned a small weight of 0.038. Also, according to the model, a lowercase letter followed by an uppercase letter is rare.

s>	<re	ght>	3<[A-Z]	ly>	al>7+	ing>
7.25	4.05	3.71	2.27	5.30	94.19	16.18
[a-z][A-Z]	't>	ed>7+	er>7+	ity	ent>7+	[0-9][0-9]
0.003	138.56	12.58	8.11	4.34	6.12	9382.93
qu	ex	ae	ment	ies	<wh	ate
526.42	5.265	0.001	10.83	4.37	5.26	9.79

```
was, reaser, in, there, to, will, ,, was, by,
homes, thing, be, reloverated, ther, which,
conists, at, fores, anditing, with, Mr., proveral,
the, ,, ***, on't, prolling, prothere, ,, mento,
at, yaou, 1, chestraing, for, have, to, intrally,
of, qut, ., best, compers, ***, cluseliment, uster,
of, is, deveral, this, thise, of, offect, inatever,
thifer, constranded, stater, vill, in, thase, in,
youse, menttering, and, ., of, in, verate, of, to
```

Finally, we visit the state of the model after inducing 1500 features to describe words. At this point the model is making more refined judgements regarding what is to be considered a word and what is not. The appearance of the features `{}>` and `\[@-&]{`, is explained by the fact that in preparing our corpus, certain characters were assigned special "macro" strings. For example, the punctuation characters `$`, `_`, `%`, and `&` are represented in our corpus as `\${}`, `\_{}`, `\%{}`, and `\&{}`. As the following sampled spellings demonstrate, the model has at this point recognized the existence of macros, but has not yet discerned their proper use.

7+<inte	prov	<der	<wh	19	ons>7+	ugh	ic>
4.23	5.08	0.03	2.05	2.59	4.49	5.84	7.76
sys	ally	7+<con	ide	nal	{}>	qui	\[@-&]{
4.78	6.10	5.25	4.39	2.91	120.56	18.18	913.22
iz	IB	<inc	<im	iong	$	ive>7+	<un
10.73	10.85	4.91	5.01	0.001	16.49	2.83	9.08

```
the, you, to, by, conthing, the, ., not, have,
devened, been, of, |, F., ., in, have, -, ,,
intering, ***, ation, said, prouned, ***,
suparthere, in, mentter, prement, intever, you, .,
and, B., gover, producits, alase, not, conting,
comment, but, |, that, of, is, are, by, from, here,
incements, contive, ., evined, agents, and, be, ',
thent, distements, all, --, has, will, said,
resting, had, this, was, intevent, IBM, whree,
acalinate, herned, are, ***, O., |, 1980, but,
will, ***, is, ., to, becoment, ., with, recall,
has, |, nother, ments, was, the, to, of,
stounicallity, with, camanfined, in, this,
intations, it, conanament, out, they, you
```

While clearly the model still has much to learn, it has at this point compiled a significant collection of morphological observations, and has traveled a long way toward its goal of statistically characterizing English spellings.

VI. Extensions and Relations to Other Approaches

In this section we briefly discuss some relations between our incremental feature induction algorithm for random fields and other statistical learning paradigms. We also present some possible extensions and improvements of our method.

A. *Conditional exponential models*

Almost all of what we have presented here carries over to the more general setting of conditional exponential models, including the Improved Iterative Scaling algorithm. For general conditional distributions $p(y \mid x)$ there may be no underlying random field, but with features defined as binary functions $f(x,y)$, the same general approach is applicable. The feature induction method for conditional exponential models is demonstrated for several problems in statistical machine translation in [3], where it is presented in terms of the principle of maximum entropy.

B. *Decision trees*

Our feature induction paradigm also bears some resemblence to various methods for growing classification and regression trees. Like decision trees, our method builds a top-down classification that refines features. However, decision trees correspond to constructing features that have disjoint support.

To explain, recall that a decision tree determines a partition π of a context random variable $X \in \mathcal{X}$ in order to predict the actual class of the context, represented by a random variable $Y \in \mathcal{Y}$. Each leaf in the tree corresponds to a sequence of binary features

$$f_l\,,\; f_{l\uparrow}\,,\; f_{l\uparrow\uparrow}\,,\; \ldots\,,\; f_{\text{root}}$$

where $n\uparrow$ denotes the parent of node n, each feature f_n is a question which splits $\mathcal{X}$, and where each f_n is the negation $\neg f_n$ of the question asked at its sibling node. The distribution assigned to a leaf l is simply the empirical distribution on $\mathcal{Y}$ determined by the training samples $(x,y) \in \mathcal{X} \times \mathcal{Y}$ for which $\pi(x) = l$. Each leaf l is characterized by the conjunction of these features, and different leaves correspond to conjunctions with disjoint support. In contrast, our feature induction algorithm generally results in features that have overlapping support. The criterion of evaluating questions in terms of the amount by which they reduce the conditional entropy of Y corresponds to our criterion of maximizing the reduction in Kullback-Leibler divergence, $G_q(g)$, over all candidate features g for a field q.

By modifying our induction algorithm in the following way, we obtain an algorithm closely related to standard methods for growing binary decision trees. Instead of considering the 1-parameter family of fields $q_{\lambda,g}$ to determine the best candidate $g = a \wedge f$, we consider the 2-parameter family of fields given by

$$q_{\lambda,\lambda',f}(y \mid x) = \frac{1}{Z_{\lambda,\lambda',f}(x)} e^{\lambda a(x,y) \wedge f(x,y) + \lambda'(\neg a)(x,y) \wedge f(x,y)} \,.$$

Since the features $a \wedge f$ and $(\neg a) \wedge f$ have disjoint support, the improvement obtained by adding both of them is given by $G_q(a \wedge f) + G_q((\neg a) \wedge f)$. In general, the resulting distribution is not absolutely continuous with respect to the empirical distribution. If the random variable Y can take on M values $y_1, \ldots y_M$, then the standard decision tree algorithm is obtained if at the n-th stage we add the $2M$ (disjoint) features $f_n(x) \wedge \delta(y_i, y)$, $\neg f_n(x) \wedge \delta(y_i, y)$, for $i = 1, \ldots, M$. Maximum likelihood training of the parameters of these features recovers the empirical distribution of the data at node n.

C. Extensions

As mentioned in Section 1, our approach differs from the most common applications of statistical techniques in computer vision, since a typical application of our method involves the estimation of thousands of free parameters. Yet the induction technique may not scale well to large 2-dimensional image problems. One potential difficulty is that the degree of the polynomials in the Improved Iterative Scaling algorithm could be quite large, and it could be difficult to obtain reliable estimates of the coefficients since Monte Carlo sampling might not exhibit sufficiently many instances of the desired features. The extent to which this is a significant problem is primarily an empirical issue, dependent on the particular domain to which the method is applied.

The random field induction method presented in this paper is not definitive; there are many possible variations on the basic theme, which is to incrementally construct an increasingly detailed exponential model to approximate the reference distribution $\tilde{p}$. Because the basic technique is based on a greedy algorithm, there are of course many ways for improving the search for a good set of features. The algorithm presented in Section 2 is in some respects the most simple possible within the general framework. But it is also computationally intensive. A natural modification would be to add several of the top candidates at each stage. While this should increase the overall speed of the induction algorithm, it would also potentially result in more redundancy among the features, since the top candidates could be correlated. Another modification of the algorithm would be to add only the best candidate at each step, but then to carry out parameter estimation only after several new features had been added to the field. It would also be natural to establish a more Bayesian framework in which a prior distribution on features and parameters is incorporated. This could enable a principled approach for deciding when the feature induction is complete. While there is a natural class of conjugate priors for the class of exponential models that we use [14], the problem of incorporating prior knowledge about the set of candiate features is more challenging.

APPENDIX

I. DUALITY

In this Appendix we prove Proposition 4 restated here.

Proposition 4: Suppose that $\tilde{p} \ll q_0$. Then there exists a unique $q_\star \in \Delta$ satisfying

(1) $q_\star \in \mathcal{P} \cap \bar{\mathcal{Q}}$
(2) $D(p \,\|\, q) = D(p \,\|\, q_\star) + D(q_\star \,\|\, q)$ for any $p \in \mathcal{P}$ and $q \in \bar{\mathcal{Q}}$
(3) $q_\star = \arg\min_{q \in \bar{\mathcal{Q}}} D(\tilde{p} \,\|\, q)$
(4) $q_\star = \arg\min_{p \in \mathcal{P}} D(p \,\|\, q_0)$.

Moreover, any of these four properties determines $q_\star$ uniquely.

Our proof of the proposition will use a few lemmas. The first two lemmas we state without proof.

Lemma 1:
(1) $D(p \,\|\, q)$ is a non-negative, extended real-valued function on $\Delta \times \Delta$.
(2) $D(p \,\|\, q) = 0$ if and only if $p = q$.
(3) $D(p \,\|\, q)$ is strictly convex in p and q separately.
(4) $D(p \,\|\, q)$ is C^1 in q.

Lemma 2:
(1) The map $(\gamma, p) \mapsto \gamma \circ p$ is smooth in $(\gamma, p) \in \mathbf{R}^n \times \Delta$.
(2) The derivative of $D(p \,\|\, \lambda \circ q)$ with respect to λ is

$$\frac{d}{dt}|_{t=0}\, D(p \,\|\, (t\lambda) \circ q) = \lambda \cdot (p[f] - q[f]) \,.$$

Lemma 3: If $\tilde{p} \ll q_0$ then $\mathcal{P} \cap \bar{\mathcal{Q}}$ is nonempty.

Proof: Define $q_\star$ by property (3) of Proposition 4; that is, $q_\star = \arg\min_{q \in \bar{\mathcal{Q}}} D(\tilde{p} \,\|\, q)$. To see that this makes sense, note that since $\tilde{p} \ll q_0$, $D(\tilde{p} \,\|\, q)$ is not identically ∞ on $\bar{\mathcal{Q}}$. Also, $D(p \,\|\, q)$ is continuous and strictly convex as a function of q. Thus, since $\bar{\mathcal{Q}}$ is closed, $D(\tilde{p} \,\|\, q)$ attains its minimum at a unique point $q_\star \in \bar{\mathcal{Q}}$. We will show that $q_\star$ is also in $\mathcal{P}$. Since $\bar{\mathcal{Q}}$ is closed under the action of $\mathbf{R}^n$, $\lambda \circ q_\star$ is in $\bar{\mathcal{Q}}$ for any λ. Thus by the definition of $q_\star$, $\lambda = 0$ is a minimum of the function $\lambda \to D(\tilde{p} \,\|\, \lambda \circ q_\star)$. Taking derivatives with respect to λ and using Lemma A.2 we conclude $q_\star[f] = \tilde{p}[f]$. Thus $q_\star \in \mathcal{P}$. □

Lemma 4: If $q_\star \in \mathcal{P} \cap \bar{\mathcal{Q}}$ then for any $p \in \mathcal{P}$ and $q \in \bar{\mathcal{Q}}$

$$D(p \,\|\, q) = D(p \,\|\, q_\star) + D(q_\star \,\|\, q) \,.$$

Proof: A straightforward calculation shows that

$$\begin{aligned} &D(p_1 \,\|\, q_1) - D(p_1 \,\|\, q_2) - D(p_2 \,\|\, q_1) + D(p_2 \,\|\, q_2) \\ &\quad = \lambda \cdot (p_1[f] - p_2[f]) \end{aligned}$$

for any $p_1, p_2, q_1, q_2 \in \Delta$ with $q_2 = \lambda \circ q_1$. It follows from this identity and the continuity of D that

$$D(p_1 \,\|\, q_1) - D(p_1 \,\|\, q_2) - D(p_2 \,\|\, q_1) + D(p_2 \,\|\, q_2) = 0$$

if $p_1, p_2 \in \mathcal{P}$ and $q_1, q_2 \in \bar{\mathcal{Q}}$. The lemma follows by taking $p_1 = q_1 = q_\star$. □

Proof of Proposition 4: Choose $q_\star$ to be any point in $\mathcal{P} \cap \bar{\mathcal{Q}}$. Such a $q_\star$ exists by Lemma A.3. It satisfies property (1) by definition, and it satisfies property (2) by Lemma A.4. As a consequence of property (2), it also satisfies properties (3) and (4). To check property (3), for instance, note that if q is any point in $\bar{\mathcal{Q}}$, then $D(\tilde{p} \,\|\, q) = D(\tilde{p} \,\|\, q_\star) + D(q_\star \,\|\, q) \geq D(\tilde{p} \,\|\, q_\star)$.

It remains to prove that each of the four properties (1)–(4) determines $q_\star$ uniquely. In other words, we need to show that if m is any point in Δ satisfying any of the four properties (1)–(4), then $m = q_\star$. Suppose that m satisfies property (1). Then by property (2) for $q_\star$ with $p = q = m$, $D(m \,\|\, m) = D(m \,\|\, q_\star) + D(q_\star \,\|\, m)$. Since $D(m \,\|\, m) = 0$, it follows that $D(m, q_\star) = 0$ so $m = q_\star$. If m satisfies property (2), then the same argument with $q_\star$ and m reversed again proves that $m = q_\star$. Suppose that m satisfies property (3). Then

$$D(\tilde{p} \,\|\, q_\star) \geq D(\tilde{p} \,\|\, m) = D(\tilde{p} \,\|\, q_\star) + D(q_\star \,\|\, m)$$

where the second equality follows from property (2) for $q_\star$. Thus $D(q_\star \,\|\, m) \le 0$ so $m = q_\star$. If m satisfies property (4), then a similar proof shows that once again $m = q_\star$. □

II. Dealing with ∞

In this Appendix we prove an extension of Proposition 5 that allows the components of γ to equal $-\infty$. For this extension, we assume that all of the components of the feature function f are non-negative: $f_i(\omega) \ge 0$ for all i and all ω. This can be assumed with no loss of generality since we can replace f_i by $f_i - \min_\omega f_i(\omega)$ if necessary.

Let $R \cup -\infty$ denote the partially extended real numbers with the usual topology. The operations of addition and exponentiation extend continuously to $R \cup -\infty$. Let $\mathcal{S}$ be the open subset of $(R \cup -\infty)^n \times \Delta$ defined by

$$\mathcal{S} = \{\, (\gamma, q) \,:\, q(\omega) e^{\gamma \cdot f(\omega)} > 0 \text{ for some } \omega \,\}$$

Observe that $R^n \times \Delta$ is a dense subset of $\mathcal{S}$. The map $(\gamma, q) \mapsto \gamma \circ p$, which up to this point we defined only for finite γ, extends uniquely to a continuous map from all of $\mathcal{S}$ to Δ. (The condition on $(\gamma, q) \in \mathcal{S}$ ensures that the normalization in the definition of $\gamma \circ p$ is well defined, even if γ is not finite.)

Definition 3: We call a function $A : \mathcal{S} \to R \cup -\infty$ an *extended auxiliary function* for L if when restricted to $R^n \times \Delta$ it is an ordinary auxiliary function in the sense of Definition 2, and if, in addition, it satisfies property (1) of Definition 2 for any $(q, \gamma) \in \mathcal{S}$, even if γ is not finite.

Note that if an ordinary auxiliary function extends to a continuous function on $\mathcal{S}$, then the extension is an extended auxiliary function.

We have the following extension of Proposition 5:

Proposition 5: Suppose the feature function f satisfies the non-negativity condition 7 and suppose A is an extended auxiliary function for L. Then the conclusion of Proposition 5 continues to hold if the condition on $\gamma^{(k)}$ is replaced by: $(\gamma^{(k)}, q^{(k)}) \in \mathcal{S}$ and $A(\gamma^{(k)}, q^{(k)}) \ge A(\gamma, q^{(k)})$ for any $(\gamma, q^{(k)}) \in \mathcal{S}$.

Proof: Lemma 1 is valid under the altered condition on $\gamma^{(k)}$ since $A(\gamma, q)$ satisfies property (1) of Definition 2 for all $(\gamma, q) \in \mathcal{S}$. As a consequence, Lemma 2 also is valid, and the proof of Proposition 5 goes through without change. □

III. Acknowledgements

Part of the research presented in this paper was carried out while the authors were with the IBM Thomas J. Watson Research Center in Yorktown Heights, New York. Stephen Della Pietra and Vincent Della Pietra's work was partially supported by ARPA under grant N00014-91-C-0135. John Lafferty's work was partially supported by NSF and ARPA under grants IRI-9314969 and N00014-92-C-0189.

References

[1] M. Almeida and B. Gidas, "A variational method for estimating the parameters of MRF from complete or incomplete data," The Annals of Applied Probability, **3**, No. 1, 103–136, 1993.

[2] N. Balram and J. Moura, "Noncausal Gauss Markov random fields: Parameter structure and estimation," IEEE Transactions on Information Theory **39**, No. 4, 1333–1343, July, 1993.

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

[4] L. Breiman, J. Friedman, R. Olshen, and C. Stone, *Classification and Regression Trees*, Wadsworth, Belmont, 1984.

[5] D. Brown, "A note on approximations to discrete probability distributions," Information and Control **2**, 386–392 (1959).

[6] P. Brown, V. Della Pietra, P. de Souza, J. Lai, and R. Mercer, "Class-based n-gram models of natural language," Computational Linguistics **18**, No. 4, 467–479, 1992.

[7] P. F. Brown, J. Cocke, V. Della-Pietra, S. Della-Pietra, J. D. Lafferty, R. L. Mercer, and P. S. Roossin. "A statistical approach to machine translation," *Computational Linguistics*, **16**(2):79–85, 1990.

[8] B. Chalmond, "An iterative Gibbsian technique for reconstruction of m-ary images," Pattern Recognition, **22** No. 6, 747–761, 1989.

[9] I. Csiszár, "I-Divergence geometry of probability distributions and minimization problems," The Annals of Probability, **3**, No. 1, 146–158, 1975.

[10] I. Csiszár, "A geometric interpretation of Darroch and Ratcliff's generalized iterative scaling," The Annals of Statistics, **17**, No. 3, 1409–1413, 1989.

[11] I. Csiszár and G. Tusnády, "Information geometry and alternating minimization procedures," Statistics & Decisions, Supplement Issue, **1**, 205–237, 1984.

[12] J. Darroch and D. Ratcliff, "Generalized iterative scaling for log-linear models," Ann. Math. Statist. **43**, 1470–1480, 1972.

[13] A.P. Dempster, N.M. Laird, and D.B. Rubin, "Maximum likelihood from incomplete data via the EM algorithm," Journal of the Royal Statistical Society **39**, No. B, 1–38, 1977.

[14] P. Diaconis and D. Ylvisaker, "Conjugate priors for exponential families," Ann. Statist. **7**, 269–281, 1979.

[15] P. Ferrari, A. Frigessi and R. Schonmann, "Convergence of some partially parallel Gibbs samplers with annealing," The Annals of Applied Probability, **3** No. 1, 137–152, 1993.

[16] A. Frigessi, C. Hwang, and L. Younes, "Optimal spectral structure of reversible stochastic matrices, Monte Carlo methods and the simulation of Markov random fields," The Annals of Applied Probability, **2**, No. 3, 610–628, 1992.

[17] S. Geman and D. Geman, "Stochastic relaxation, Gibbs distributions, and the Bayesian restoration of images," IEEE Trans. Pattern Anal. Machine Intell. **6**, 721–741, 1984.

[18] C. Geyer and E. Thomson, "Constrained Monte Carlo maximum likelihood for dependent data (with discussion)", J. Royal Stat. Soc. B-54, 657–699, 1992.

[19] E. T. Jaynes, *Papers on Probability, Statistics, and Statistical Physics*, R. Rosenkrantz, ed., D. Reidel Publishing Co., Dordrecht–Holland, 1983.

[20] J. Lafferty and R. Mercer, "Automatic word classification using features of spellings," Proceedings of the 9th Annual Conference of the University of Waterloo Centre for the New OED and Text Research, Oxford University Press, Oxford, England, 1993.

[21] G. Potamianos and J. Goutsias, "Partition function estimation of Gibbs random field images using Monte Carlo simulations," IEEE Transactions on Information Theory **39**, No. 4, 1322–1331, July, 1993.

[22] L. Younes, "Estimation and annealing for Gibbsian fields," Ann. Inst. H. Poincaré Probab. Statist. **24** No. 2, 269–294, 1988.

Stephen A. Della Pietra received his A.B. in Physics from Princeton University in 1981, and Ph.D. in Physics from Harvard University in 1987. From 1987 until 1988 he was a Post-Doctoral Fellow at the University of Texas at Austin. From 1988 until 1989, he was a Member of the Mathematics Department of the Institute for Advanced Study in Princeton, NJ. From 1989 until 1995 he was a Research Staff Member at the IBM Thomas J. Watson Research Center in Yorktown Heights and Hawthorne, NY, where he was project leader of the natural language understanding group. His primary research at IBM was in machine translation and natural language understanding. Since 1995 he has been working on statistical methods for modeling the stock market at Renaissance Technologies in Stony Brook, NY.

Vincent J. Della Pietra received his A.B. in Mathematics from Princeton University in 1982, and Ph.D. in Mathematical Physics from Harvard University in 1988. From 1987 until 1988, he was a Member of the Mathematics Department of the Institute for Advanced Study in Princeton, NJ. From 1988 until 1995 he was a Research Staff Member at the IBM Thomas J. Watson Research Center in Yorktown Heights and Hawthorne, NY, where he was project leader of the machine translation group. His primary research at IBM was in machine translation and natural language understanding. Since 1995 he has been working on statistical methods for modeling the stock market at Renaissance Technologies in Stony Brook, NY.

John D. Lafferty studied mathematics as an undergraduate at Middlebury College and the Massachusetts Institute of Technology, and received the Ph.D. degree in Mathematics from Princeton University in 1986, where he was a member of the Program in Applied and Computational Mathematics. From 1986 until 1987 he was an Assistant Professor and Benjamin Pierce Lecturer on Mathematics at Harvard University, and from 1987 until 1988 he was Visiting Assistant Professor at the Nankai Institute of Mathematics in Tianjin, China. In 1989 he joined the Computer Sciences Department of the IBM Thomas J. Watson Research Center in Yorktown Heights, NY, as a Research Staff Member. Since 1994 he has been a member of the faculty of the Computer Science Department at Carnegie Mellon University, where his primary research interests include speech, language, information theory and statistical learning algorithms.