Log-Linear Models

Noah A. Smith*
Department of Computer Science /
Center for Language and Speech Processing
Johns Hopkins University
nasmith@cs.jhu.edu

December 2004

Abstract

This is yet another introduction to log-linear ("maximum entropy") models for NLP practitioners, in the spirit of Berger (1996) and Ratnaparkhi (1997b). The derivations here are similar to Berger's, but more details are filled in and some errors are corrected. I do not address iterative scaling (Darroch and Ratcliff, 1972), but rather give derivations of the gradient and Hessian of the dual objective function (conditional likelihood).

Note: This is a draft; please contact the author if you have comments, and do not cite or circulate this document.

1 Log-linear Models

Log-linear models[1] have become a widely-used tool in NLP classification tasks (Berger et al., 1996; Ratnaparkhi, 1998). Log-linear models assign joint probabilities to observation/label pairs $(x, y) \in \mathcal{X} \times \mathcal{Y}$ as follows:

$$\Pr_{\vec{\theta}}(x, y) = \frac{\exp\left(\vec{\theta} \cdot \vec{f}(x, y)\right)}{\sum_{x', y'} \exp\left(\vec{\theta} \cdot \vec{f}(x', y')\right)} \tag{1}$$

where $\vec{\theta}$ is a $\mathbb{R}$-valued vector of feature weights and $\vec{f}$ is a function that maps pairs (x, y) to a nonnegative $\mathbb{R}$-valued feature vector. These features can take on any form; in particular, unlike directed, generative models (like HMMs and PCFGs), the features may overlap, predicting parts of the data more than once.[2] Each feature has an associated θ_i, which is called its *weight*.

Maximum likelihood parameter estimation (training) for such a model, with a set of labeled examples, amounts to solving the following optimization problem. Let $\{(x_1, y_1^*), (x_2, y_2^*), ..., (x_m, y_m^*)\}$

*This document is a revised version of portions of the author's 2004 thesis research proposal, "Discovering grammatical structure in unannotated text: implicit negative evidence and dynamic feature selection."

[1]Such models have many names, including maximum-entropy models, exponential models, and Gibbs models; Markov random fields are structured log-linear models, conditional random fields (Lafferty et al., 2001) are Markov random fields with a specific training criterion.

[2]The ability to handle arbitrary, overlapping features is an important advantage that log-linear models have over directed generative models (like HMMs and PCFGs). Importantly, for the latter type of models, maximum likelihood estimation is not straightforward when complicated feature dependencies are introduced that cannot be described as a series of generative steps (Abney, 1997). This comparison will be more fully explored in my thesis.

be the labeled training set.

$$\vec{\theta}^* = \max_{\vec{\theta}} \prod_{j=1}^{m} \Pr_{\vec{\theta}}(x_j, y_j^*) \tag{2}$$

$$= \max_{\vec{\theta}} \sum_{j=1}^{m} \log \Pr \vec{\theta}(x_j, y_j^*) \tag{3}$$

$$\overset{\text{(Eq. 1)}}{=} \max_{\vec{\theta}} \sum_{j=1}^{m} \log \frac{\exp\left(\vec{\theta} \cdot \vec{f}(x_j, y_j^*)\right)}{\left\{\sum_{x',y'} \exp\left(\vec{\theta} \cdot \vec{f}(x', y')\right)\right\}_{Z(\vec{\theta})}} \tag{4}$$

$$= \left(\sum_{j=1}^{m} \vec{\theta} \cdot \vec{f}(x_j, y_j^*)\right) - m \log Z(\vec{\theta}) \tag{5}$$

This function can be shown to be concave, with a single global maximum.[3]

Commonly the *conditional probability* of the label given the observation is computed; for one example (x, y) it is given by:

$$\Pr_{\vec{\theta}}(y \mid x) = \frac{\exp\left(\vec{\theta} \cdot \vec{f}(x, y)\right)}{\sum_{y'} \exp\left(\vec{\theta} \cdot \vec{f}(x, y')\right)} \tag{6}$$

Why is this quantity important? A second feature of log-linear models is that they admit relatively efficient *conditional* parameter estimation. Conditional training is a technique that seeks to give maximum probability (or weight, or score) to the correct label y^* for a given observation x (known from an annotated corpus), *at the expense* of competing possible labels. This kind of training can be contrasted with joint maximum likelihood training (Equation 4), which seeks to maximize the total probability of training pairs (x_j, y_j^*).

Conditional training for log-linear models amounts to maximizing the product of conditional probabilities (as in Equation 6):[4]

$$\vec{\theta}^* = \max_{\vec{\theta}} \prod_{j=1}^{m} \Pr_{\vec{\theta}}(y_j^* \mid x_j) \tag{7}$$

$$= \max_{\vec{\theta}} \sum_{j=1}^{m} \log \Pr_{\vec{\theta}}(y_j^* \mid x_j) \tag{8}$$

$$= \left(\sum_{j=1}^{m} \vec{\theta} \cdot \vec{f}(x_j, y_j^*)\right) - \sum_{j=1}^{m} \log \underbrace{\sum_{y'} \exp\left(\vec{\theta} \cdot \vec{f}(x_j, y')\right)}_{Z(\vec{\theta}, x_j)} \tag{9}$$

The argument for conditional training is that it is unnecessary to model x, the observed part of the data, since (even at test time) it is given; we need only be able to determine which y fits x

[3] In Section 3 I derive the Hessian and demonstrate that the function is everywhere concave.

[4] A discussion of conditional likelihood as a statistical estimator is given by Johnson et al. (1999). An earlier discussion is Nadas (1983).

best, and predicting x is extraneous effort imposed on the model.[5]

There is also a practical reason for using conditional likelihood $\Pr_{\vec{\theta}}(y^* \mid x)$ rather than $\Pr_{\vec{\theta}}(x, y^*)$. Consider the terms marked in Equations 4 and 9 as $Z(\vec{\theta})$ and $Z(\vec{\theta}, x_j)$, respectively. This term is known as the *partition function*. In ML training (Equation 4), computing it involves summing over the entire space of observation/label pairs predicted by the model, while in conditional training (Equation 9) one must sum only over the labels for each example x. The former is in practice much more computationally expensive, even with approximations.

Both optimization problems are *unconstrained* and convex, making them relatively straightforward to solve (modulo the partition function) using iterative hill-climbing methods. Iterative scaling (Darroch and Ratcliff, 1972) is one approach; Malouf (2002) describes how Newtonian numerical optimization algorithms using the gradient of the function can be used (see Section 3).[6]

With log-linear models of labeled data, the numerical part of learning is straightforward; the difficult part is feature selection. Using too many features is likely to result in over-fitting of the training data. One approach is to use features that have frequency in training data above some threshold; Della Pietra et al. (1997) use a more sophisticated feature selection algorithm. Another way of avoiding over-fitting is to smooth the model. This is often done using a prior (Khudanpur, 1995; Chen and Rosenfeld, 2000). One recent approach to smoothing by Kazama and Tsujii (2003) (this work extends that of Khudanpur (1995)) uses soft constraints and a penalty function (which can be interpreted as a prior on parameter values); this has the side-effect of many parameters going to zero,[7] resulting in automatic feature selection.[8] Of course, introducing priors requires introducing more parameters (and perhaps auxiliary learning algorithms to estimate them). This is an active area of research.

Log-linear models have been applied to many supervised problems in NLP:

- Sentence boundary detection (Reynar and Ratnaparkhi, 1997)
- Parsing (Ratnaparkhi et al., 1994b; Ratnaparkhi, 1997a; Abney, 1997; Riezler et al., 2000; Johnson and Riezler, 2000; Johnson, 2001)[9]

[5]Conditional training encompasses a number of approaches to parameter estimation and originated in the automatic speech recognition community. In particular, maximum mutual information (MMI) estimation (Bahl et al., 1986) can be shown to be equivalent to conditional maximum likelihood estimation. The other approach is to directly minimize classification error (Juang and Katagiri, 1992).

[6]The learning task for log-linear models is usually described as follows. Given a vector of observed feature values $\tilde{\vec{f}}$, we seek model parameters $\vec{\theta}$ that yield these expectations. That is, we wish the following constraint to be met for all feature indices i:

$$\sum_{x,y} \Pr_{\vec{\theta}}(y \mid x) f_i(x, y) = \tilde{f}_i. \tag{10}$$

Now, there may be many solutions to this system of equations. The criterion used to decide among them is *maximum entropy* — we seek the parameters that are closest to uninformative (uniform), subject to the expectation constraints. This is motivated by an Occam's razor argument that advocates choosing the simplest possible model that explains the data; entropy can be roughly regarded as a measure of simplicity of the model.

It turns out that the dual of this constrained optimization problem has the form of the maximum likelihood problems given in Equations 4 and 9, depending on whether the entropy to be maximized is joint entropy $H(\Pr_{\vec{\theta}}(X, Y))$ or the total entropy of the conditional distributions for each x_j, $\sum_{j=1}^{m} H(\Pr_{\vec{\theta}}(Y \mid x_j))$ (Berger et al., 1996). See Section 2 for a derivation of the dual in the latter case.

[7]A zero-weighted feature will have no effect on the probability of examples where the feature is present; see Equation 1.

[8]To my knowledge, this technique has not been carefully tested as a feature selection technique *per se*, though M. Osborne (personal communication) reports that its performance as such is reasonable in comparison with other feature selection algorithms.

[9]Charniak (2000) used log-linear-like features to build a state-of-the-art parser, but without training via the

- Priors over grammar rules (Eisner, 2001)
- Language modeling for automatic speech recognition (Rosenfeld, 1994; Khudanpur and Wu, 2000)
- Prepositional phrase attachment (Ratnaparkhi et al., 1994a)
- Part-of-speech tagging (Ratnaparkhi, 1996)
- Named entity recognition (Borthwick et al., 1998)
- Machine translation (Berger et al., 1996; Och and Ney, 2002)

2 Maximum Conditional Likelihood as the Dual of Maximum Entropy

Berger et al. (1996) did not explicitly derive the relationship between the maximum entropy estimation problem and the maximum likelihood problem; they gave only a sketch. Here I give a derivation that shows how maximum conditional likelihood estimation is the dual of the maximum entropy problem.

Notation is shown in Table 1.

The maximum entropy problem we seek to solve is:

$$\underset{p}{\text{maximize}} \quad \sum_x \tilde{p}(x) H(p(Y \mid x)) \tag{11}$$

$$\text{subject to} \quad \sum_{(x,y)} \tilde{p}(x,y) f_j(x,y) = \sum_x \tilde{p}(x) \sum_y p(y \mid x) f_j(x,y), \forall j \tag{12}$$

$$\sum_y p(y \mid x) = 1, \forall x \tag{13}$$

$$p(y \mid x) \geq 0, \forall x, y \tag{14}$$

In short, we seek to maximize the empirically-averaged entropy over labels given each example x, subject to the constraint that, on average the feature expectations within each neighborhood match the feature expectations of the corresponding observations.

maximum-entropy criterion.

$\mathcal{X}$	the space of unlabeled examples
$\mathcal{Y}$	the space of labels
Y	a random variable over labels
$\tilde{p}$	the observed (empirical) distribution over $\mathcal{X} \times \mathcal{Y}$ (we also use $\tilde{p}$ to denote marginals over $\mathcal{X}$)
p	the desired distribution over $\mathcal{X} \times \mathcal{Y}$ that we would like to estimate; here, p takes the form of a conditional distribution, $p(y \mid x)$
f_j	the jth feature function, $\mathcal{X} \times \mathcal{Y} \to \mathbb{R}^+$

Table 1: Notation.

We define the conditional entropy as

$$H(Y \mid x) = -\sum_y p(y \mid x) \log p(y \mid x) = \mathbf{E}_{p(Y|x)}[-\log p(Y \mid x)] \tag{15}$$

Let θ_j be a Lagrangian multiplier for constraint j. We can write the above as an unconstrained problem, where θ_j is the cost of violating the jth constraint in line 12. Let μ_x be the constraint corresponding to the constraint for x in line 13. Note that we have ignored the nonnegativity constraint in line 14; that will ultimately fall out of the derivation.

$$\begin{aligned}\min_{\vec{\mu},\vec{\theta}} \max_p \quad & \sum_x \tilde{p}(x) H(p(Y \mid x)) - \sum_j \theta_j \left(\sum_{(x,y)} \tilde{p}(x,y) f_j(x,y) - \sum_x \tilde{p}(x) \sum_y p(y \mid x) f_j(x,y) \right) \\ & - \sum_x \mu_x \left(\sum_y p(y \mid x) - 1 \right)\end{aligned} \tag{16}$$

Next we hold the Lagrangian multipliers θ_j and μ_x constant and maximize Equation 16 with respect to $p(y \mid x)$. This will give a closed form solution for $p(y \mid x)$ in terms of the Lagrangian multipliers.[10]

To solve Equation 16 with respect to $p(y \mid x)$, we take the derivative with respect to $p(y \mid x)$. (Note that there are many x and many y; the form of the derivative with respect to each will be the same, with different bindings for x, y.):

$$\frac{\partial}{\partial p(y \mid x)} = -\tilde{p}(x) \left(1 + \log p(y \mid x)\right) + \sum_j \theta_j \tilde{p}(x) f_j(x,y) - \mu_x \tag{17}$$

Setting this equal to 0 and solving for $p(y \mid x)$, we have:

$$\begin{aligned}-\tilde{p}(x) \left(1 + \log p(y \mid x)\right) + \sum_j \theta_j \tilde{p}(x) f_j(x,y) - \mu_x &= 0 \\ \tilde{p}(x) \left(1 + \log p(y \mid x)\right) &= \sum_j \theta_j \tilde{p}(x) f_j(x,y) - \mu_x \\ 1 + \log p(y \mid x) &= \frac{\sum_j \theta_j \tilde{p}(x) f_j(x,y) - \mu_x}{\tilde{p}(x)} \\ \log p(y \mid x) &= \vec{\theta} \cdot \vec{f}(x,y) + \frac{\mu_x}{\tilde{p}(x)} - 1 \\ p(y \mid x) &= \frac{\exp\left(\vec{\theta} \cdot \vec{f}(x,y)\right)}{\exp\left(1 - \frac{\mu_x}{\tilde{p}(x)}\right)}\end{aligned}$$

The remarkable thing at this point is to notice that the optimal $p(y \mid x)$ will take on an exponential form. Note that the numerator is exactly the numerator in Equation 6. The denominator is

[10] The theory underlying these moves is hard to give at the right level of detail. Like most other NLP-oriented expositions of log-linear models, this one will avoid that theory. The most rigorous treatment of this area is Bertsekas (1999). The key point from the theory is that a maximum in Equation 11 will always correspond to a maximum in Equation 16 (an unconstrained problem with more variables). For this problem, we can move the optimization burden onto the Lagrange multipliers and forget about $p(y \mid x)$, because the latter have a closed form in terms of the former.

a function only of μ_x; to enforce that the sum-to-one constraints (Equation 13) are met, we simply let, for all x,

$$\sum_{y'} \exp\left(\vec{\theta} \cdot \vec{f}(x, y')\right) = \exp\left(1 - \frac{\mu_x}{\tilde{p}(x)}\right) \tag{18}$$

(Note that the constraints in line 14 must be satisfied, because the range of the exponential function is nonnegative.) We see now that the solution to the maximum entropy problem will always be a log-linear model. This is the source of the name, "maximum entropy model," though I maintain that the name is not always appropriate because there are alternative training objectives for these models.

Now we substitute this form back into Equation 16. We remove the μ_x terms, since we have guaranteed by the form of the model that the sum-to-one constraints (Equation 13) are satisfied:

$$\begin{aligned}
&\sum_x \tilde{p}(x) H(p(Y \mid x)) - \sum_j \theta_j \left(\sum_{(x,y)} \tilde{p}(x,y) f_j(z) - \sum_x \tilde{p}(x) \sum_y p(y \mid x) f_j(x,y) \right) \\
&= -\sum_x \tilde{p}(x) \sum_y p(y \mid x) \log p(y \mid x) - \sum_j \theta_j \left(\sum_{(x,y)} \tilde{p}(x,y) f_j(x,y) - \sum_x \tilde{p}(x) \sum_y p(y \mid x) f_j(x,y) \right) \\
&= -\sum_x \tilde{p}(x) \sum_y p(y \mid x) \log p(y \mid x) - \sum_{(x,y)} \tilde{p}(x,y) \sum_j \theta_j f_j(x,y) + \sum_x \tilde{p}(x) \sum_y p(y \mid x) \sum_j \theta_j f_j(x,y) \\
&= -\sum_x \tilde{p}(x) \sum_y p(y \mid x) \log p(y \mid x) - \sum_{(x,y)} \tilde{p}(x,y) \left(\vec{\theta} \cdot \vec{f}(x,y)\right) + \sum_x \tilde{p}(x) \sum_y p(y \mid x) \left(\vec{\theta} \cdot \vec{f}(x,y)\right) \\
&= -\sum_x \tilde{p}(x) \sum_y p(y \mid x) \left[\left(\vec{\theta} \cdot \vec{f}(x,y)\right) - \log \underbrace{\sum_{y'} \exp\left(\vec{\theta} \cdot \vec{f}(x,y')\right)}_{Z(\vec{\theta},x)} \right] \\
&\qquad - \sum_{(x,y)} \tilde{p}(x,y) \left(\vec{\theta} \cdot \vec{f}(x,y)\right) + \sum_x \tilde{p}(x) \sum_y p(y \mid x) \left(\vec{\theta} \cdot \vec{f}(x,y)\right) \\
&= -\sum_x \tilde{p}(x) \sum_y p(y \mid x) \left(\vec{\theta} \cdot \vec{f}(x,y)\right) + \sum_x \tilde{p}(x) \sum_y p(y \mid x) \log Z\left(\vec{\theta}, x\right) \\
&\qquad - \sum_{(x,y)} \tilde{p}(x,y) \left(\vec{\theta} \cdot \vec{f}(x,y)\right) + \sum_x \tilde{p}(x) \sum_y p(y \mid x) \left(\vec{\theta} \cdot \vec{f}(x,y)\right)
\end{aligned}$$

The first and last terms cancel.

$$\begin{aligned}
&= \sum_x \tilde{p}(x) \sum_y p(y \mid x) \log Z\left(\vec{\theta}, x\right) - \sum_{(x,y)} \tilde{p}(x,y) \left(\vec{\theta} \cdot \vec{f}(x,y)\right) \\
&= \sum_x \tilde{p}(x) \log Z\left(\vec{\theta}, x\right) - \sum_{(x,y)} \tilde{p}(x,y) \left(\vec{\theta} \cdot \vec{f}(x,y)\right) \\
&= -\sum_{(x,y)} \tilde{p}(x,y) \log \frac{\exp\left(\vec{\theta} \cdot \vec{f}(x,y)\right)}{Z\left(\vec{\theta}, x\right)}
\end{aligned}$$

This is the cross-entropy, and we seek to minimize it. Cross-entropy is equivalent to the negative likelihood (Equation 9) times a constant factor (m, the size of the data).

3 A Bit About Training

I noted previously that Newtonian methods can be used to carry out maximum conditional likelihood training of log-linear models. First-order Newtonian methods operate by iteratively improving parameter values. Given parameter values $\vec{\theta}^{(i)}$, the objective $\mathcal{L}\left(\vec{\theta}^{(i)}\right)$ and the gradient $\nabla_{\vec{\theta}^{(i)}}\mathcal{L}$ are computed. The optimization proceeds by letting $\vec{\theta}^{(i+1)} = \vec{\theta}^{(i)} + \alpha\nabla_{\vec{\theta}^{(i)}}\mathcal{L}$, i.e., moving in the direction of the gradient. (α, the step size, is determined using an auxiliary procedure that looks for the step size that will increase $\mathcal{L}$ as much as possible.) Modern Newtonian methods may use an additional term in the update, taking advantage of exact or approximate second-derivative information. Since log-linear models usually have many parameters, it is not typically tractable to compute the second derivative exactly (though we will see that it has a very simple form).

Here I derive the gradient and show that the function is everywhere concave. We start with the objective to be maximized (Equation 9, repeated below):

$$\mathcal{L}\left(\vec{\theta}\right) = \left(\sum_{j=1}^{m} \vec{\theta}\cdot\vec{f}(x_j, y_j^*)\right) - \sum_{j=1}^{m} \log \sum_{y'} \exp\left(\vec{\theta}\cdot\vec{f}(x_j, y')\right) \tag{19}$$

$$= \left(\sum_{j=1}^{m}\sum_{k} \theta_k f_k(x_j, y_j^*)\right) - \sum_{j=1}^{m} \log \sum_{y'} \exp\left(\sum_{k} \theta_k f_k(x_j, y')\right) \tag{20}$$

Taking the derivative with respect to θ_ℓ, we have:

$$\frac{\partial\mathcal{L}}{\partial\theta_\ell} = \left(\sum_{j=1}^{m} f_\ell(x_j, y_j^*)\right) - \sum_{j=1}^{m} \frac{\sum_{y'} \left(\exp\sum_k \theta_k f_k(x_j, y')\right) f_\ell(x_j, y')}{\sum_{y'} \exp\sum_k \theta_k f_k(x_j, y')} \tag{21}$$

$$= \sum_{j=1}^{m}\left(f_\ell(x_j, y_j^*) - \frac{\sum_{y'} \left(\exp\sum_k \theta_k f_k(x_j, y')\right) f_\ell(x_j, y')}{\sum_{y'} \exp\sum_k \theta_k f_k(x_j, y')}\right) \tag{22}$$

$$= \sum_{j=1}^{m}\left(f_\ell(x_j, y_j^*) - \sum_{y'} \Pr_{\vec{\theta}}\left(y' \mid x_j\right) f_\ell(x_j, y')\right) \tag{23}$$

$$= \sum_{j=1}^{m}\left(f_\ell(x_j, y_j^*) - \mathbf{E}_{\Pr_{\vec{\theta}}(Y|x_j)}[f_\ell(x_j, Y)]\right) \tag{24}$$

Therefore,

$$\nabla_{\vec{\theta}}\mathcal{L} = \sum_{j=1}^{m} \vec{f}(x_j, y_j^*) - \mathbf{E}_{\Pr_{\vec{\theta}}(Y|x_j)}[\vec{f}(x_j, Y)] \tag{25}$$

An important note is that, when the constraints in line 12 are met, then this gradient will be all zeroes. The gradient is no more difficult to compute than the feature expectations; the time required to do this is $O(mn|\mathcal{Y}|)$ where n is the length of $\vec{\theta}$. In some cases, such as conditional random fields Lafferty et al. (2001), $\mathcal{Y}$ is a set whose cardinality grows exponentially in the length of the input x. (In Lafferty et al.'s CRFs, x is a sequence.) Often a dynamic programming algorithm can be applied in those cases to efficiently compute the expectations.

Second derivative. We can now take the derivative of an element of the gradient with respect to θ_p. We begin with Equation 22 for convenience:

$$\frac{\partial^2 \mathcal{L}}{\partial \theta_\ell \partial \theta_p} = -\sum_{j=1}^{m} \frac{\left[\left(\sum_{y'} e^{\vec{\theta}\cdot\vec{f}(x_j,y')}\right)\left(\sum_{y'} f_\ell(x_j,y') e^{\vec{\theta}\cdot\vec{f}(x_j,y')} f_p(x_j,y')\right) - \left(\sum_{y'} e^{\vec{\theta}\cdot\vec{f}(x_j,y')} f_\ell(x_j,y')\right)\left(\sum_{y'} e^{\vec{\theta}\cdot\vec{f}(x_j,y')} f_p(x_j,y')\right)\right]}{\left(\sum_{y'} e^{\vec{\theta}\cdot\vec{f}(x_j,y')}\right)^2} \quad (26)$$

$$= -\sum_{j=1}^{m} \left(\mathbf{E}_{\Pr_{\vec{\theta}}(Y|x_j)}[f_\ell(x_j,Y) f_p(x_j,Y)] - \mathbf{E}_{\Pr_{\vec{\theta}}(Y|x_j)}[f_\ell(x_j,Y)]\mathbf{E}_{\Pr_{\vec{\theta}}(Y|x_j)}[f_p(x_j,Y)]\right)$$

$$= -\sum_{j=1}^{m} \mathrm{cov}_{\Pr_{\vec{\theta}(Y|x_j)}}(f_\ell(x_j,Y), f_p(x_j,Y)) \quad (27)$$

where cov is the covariance. We see therefore that the Hessian matrix is the negated covariance matrix with the feature values under the model as the random variables. Covariance matrices are always positive definite or positive semi-definite; this means that the function we are maximizing is concave everywhere.

References

S. P. Abney. Stochastic attribute-value grammars. *Computational Linguistics*, 23(4):597–617, 1997.

L. R. Bahl, P. F. Brown, P. V. de Souza, and R. L. Mercer. Maximum mutual information estimation of hidden Markov model parameters for speech recognition. In *Proc. of ICASSP*, 1986.

A. Berger. A brief maxent tutorial, 1996.

A. Berger, S. Della Pietra, and V. Della Pietra. A maximum entropy approach to natural language processing. *Computational Linguistics*, 22(1), 1996.

D. P. Bertsekas. *Nonlinear Programming*. Athena Scientific, second edition, 1999.

A. Borthwick, J. Sterling, E. Agichtein, and R. Grishman. Exploiting diverse knowledge sources via maximum entropy in named entity recognition. In *Proc. of VLC*, 1998.

E. Charniak. A maximum-entropy-inspired parser. In *Proc. of NAACL*, 2000.

S. Chen and R. Rosenfeld. A survey of smoothing techniques for ME models. *IEEE Transactions on Speech and Audio Processing*, 8(1):37–50, 2000.

J. N. Darroch and D. Ratcliff. Generalized iterative scaling for log-linear models. *Annals of Mathematical Statistics*, 43:1470–80, 1972.

S. Della Pietra, V. J. Della Pietra, and J. D. Lafferty. Inducing features of random fields. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 19(4):380–393, 1997.

J. Eisner. *Smoothing a Probabilistic Lexicon via Syntactic Transformations*. PhD thesis, University of Pennsylvania, 2001.

M. Johnson. Joint and conditional estimation of tagging and parsing models. In *Proc. of ACL*, 2001.

M. Johnson, S. Geman, S. Canon, Z. Chi, and S. Riezler. Estimators for stochastic "unification-based" grammars. In *Proc. of ACL*, 1999.

M. Johnson and S. Riezler. Exploiting auxiliary distributions in stochastic unification-based grammars. In *Proc. of NAACL*, 2000.

B. H. Juang and S. Katagiri. Discriminative learning for minimum error classification. *IEEE Transactions on Acoustics, Speech, and Signal Processing*, 40(12):3043–54, 1992.

J. Kazama and J. Tsujii. Evaluation and extension of maximum entropy models with inequality constraints. In *Proc. of EMNLP*, 2003.

S. Khudanpur. A method of ME estimation with relaxed constraints. In *Proc. of JHU Language Modeling Workshop*, 1995.

S. Khudanpur and J. Wu. Maximum entropy techniques for exploiting syntactic, semantic, and collocational dependencies in language modeling. *Computer Speech and Language*, 14:355–72, 2000.

J. Lafferty, A. McCallum, and F. Pereira. Conditional random fields: Probabilistic models for segmenting and labeling sequence data. In *Proc. of ICML*, pages 282–9, 2001.

R. Malouf. A comparison of algorithms for maximum entropy parameter estimation. In *Proc. of CoNLL*, 2002.

A. Nadas. A decision-theoretic formulation of a trainings problem in speech recognition and a comparison of training by unconditional versus conditional maximum likelihood. *IEEE Transactions on Acoustics, Speech, and Signal Processing*, 4:814–7, 1983.

F. J. Och and H. Ney. Discriminative training and maximum entropy models for statistical machine translation. In *Proc. of ACL*, 2002.

A. Ratnaparkhi. A maximum entropy part-of-speech tagger. In *Proc. of EMNLP*, 1996.

A. Ratnaparkhi. A linear observed time statistical parser based on maximum entropy models. In *Proc. of EMNLP*, 1997a.

A. Ratnaparkhi. A simple introduction to maximum entropy models for natural language processing, 1997b.

A. Ratnaparkhi. *Maximum Entropy Models for Natural Language Ambiguity Resolution*. PhD thesis, University of Pennsylvania, 1998.

A. Ratnaparkhi, J. Reynar, and S. Roukos. A maximum entropy model for prepositional phrase attachment. In *Proc. of ARPA HLT Workshop*, 1994a.

A. Ratnaparkhi, S. Roukos, and R. T. Ward. A maximum entropy model for parsing. In *Proc. of ICSLP*, 1994b.

J. C. Reynar and A. Ratnaparkhi. A maximum entropy approach to identifying sentence boundaries. In *Proc. of ANLP*, 1997.

S. Riezler, D. Prescher, J. Kuhn, and M. Johnson. Lexicalized stochastic modeling of constraint-based grammars using log-linear measures and EM training. In *Proc. of ACL*, 2000.

R. Rosenfeld. *Adaptive Statistical Language Modeling: A Maximum Entropy Approach*. PhD thesis, Carnegie Mellon University, 1994.