The Improved Iterative Scaling Algorithm: A Gentle Introduction

Adam Berger
School of Computer Science
Carnegie Mellon University
December, 1997

This note concerns the *improved iterative scaling* algorithm for computing maximum-likelihood estimates of the parameters of exponential models. The algorithm was invented by members of the machine translation group at IBM's T.J. Watson Research Center in the early 1990s. The goal here is to motivate the improved iterative scaling algorithm for conditional models in a way that is as complete and self-contained as possible, yet minimizes the mathematical burden on the reader[1].

Parametric form

The task is to come up with an accurate encapsulation of a random process. This random process produces, at each time step, some output value y, a member of a (necessarily finite) set of possible output values. The value of the random variable y is influenced by some conditioning information (or "context") x. The *language modelling* problem, for example, is to assign a probability $p(y \mid x)$ to the event that the next word in a sequence of text will be y, given x, the values of the previous words.

We adopt a conditional exponential model

$$p_\Lambda(y \mid x) \equiv \frac{1}{Z_\Lambda(x)} \exp\left(\sum_{i=1}^{n} \lambda_i f_i(x,y)\right) \tag{1}$$

where

- $f_i(x,y)$ is a binary-valued function—called a "feature"—of (x,y). Associated with the model p_Λ is some finite collection of n such functions. How one might select these features is not the topic of this note. There exist methods for automatically discovering "good" features from within a large collection of candidates; see [De97] or [Be96] for details.

- λ_i is a real-valued weight associated with f_i. Technically, λ_i is the Lagrange multiplier corresponding to the function f_i in a certain constrained optimization problem. In this sense, the absolute value of λ_i is a measure of the "importance" of the feature f_i. We denote by Λ the vector of weights: $\Lambda \equiv \{\lambda_1, \lambda_2, \ldots \lambda_n\}$.

[1] This and other material related to exponential models—mostly of a survey nature—are available online at `http://www.cs.cmu.edu/~aberger/maxent.html`. Comments on and suggestions for this document should be sent to `aberger@cs.cmu.edu`.

- $Z_\Lambda(x)$ is a normalizing factor, required to make p_Λ a probability distribution:

$$Z_\Lambda(x) = \sum_y \exp\left(\sum_{i=1}^{n} \lambda_i f_i(x,y)\right) \tag{2}$$

Maximum likelihood

Given a joint empirical distribution $\tilde{p}(x,y)$, the *log-likelihood* of $\tilde{p}(x,y)$ according to a conditional model $p_\Lambda(y \mid x)$, is defined as

$$L_{\tilde{p}}(\Lambda) \equiv \sum_{x,y} \tilde{p}(x,y) \log p_\Lambda(y \mid x) \tag{3}$$

We employ the log-likelihood as a measure of the quality of the model p_Λ. From (3) we can immediately see that

- $L_{\tilde{p}}(\Lambda) \leq 0$ always
- $L_{\tilde{p}}(\Lambda) = 0$ is optimal, attained by a model p_Λ which is "perfect" with respect to $\tilde{p}$; that is, $p_\Lambda(y \mid x) = 1$ if and only if $\tilde{p}(x,y) > 0$.

Given the set of features $\{f_1, f_2, \ldots f_n\}$, the exponential form (1), and an empirical distribution $\tilde{p}(x,y)$, the maximum likelihood problem is to discover $\Lambda^\star \equiv \text{argmax}_\Lambda L_{\tilde{p}}(\Lambda)$, which is a search in R^n. In the following section we will describe how to perform this search efficiently.

The log likelihood of the exponential model (1) is

$$L_{\tilde{p}}(\Lambda) = \sum_{x,y} \tilde{p}(x,y) \sum_i \lambda_i f_i(x,y) - \sum_x \tilde{p}(x) \log \sum_y \exp \sum_i \lambda_i f_i(x,y)$$

Differentiating with respect to an individual parameter λ_i, we get

$$\begin{aligned} \frac{\partial L_{\tilde{p}}(\Lambda)}{\partial \lambda_i} &= \sum_{x,y} \tilde{p}(x,y) f_i(x,y) - \sum_{x,y} \tilde{p}(x) p_\Lambda(y \mid x) f_i(x,y) \\ &= < \tilde{f_i} > - < f_i > \end{aligned} \tag{4}$$

Here $< \tilde{f_i} >$ denotes the expectation of $f_i(x,y)$ with respect to the empirical distribution $\tilde{p}$, and $< f_i >$ the expectation of $f_i(x,y)$ with respect to the distribution $\tilde{p}(x) p_\Lambda(y \mid x)$.

Setting (4) to zero yields the condition for an extremum of the log-likelihood with respect to the single parameter λ_i. And the resulting condition—that the empirical expected probability of the feature f_i be equal to the model expected probability—is a very natural condition.

When $f_i(x,y)$ is binary-valued, as is the case here, this condition has an especially intuitive interpretation: the expected fraction of events (x,y) for which f_i is "on" (non-zero) should be the same according to the empirical distribution $\tilde{p}(x,y)$ and the model distribution $\tilde{p}(x) p_\Lambda(y \mid x)$.

Finding $\Lambda^\star$

Say we have a model of the form (1) with some arbitrary set of parameters $\Lambda \equiv \{\lambda_1, \lambda_2, \ldots \lambda_n\}$. We'd like to find a new set of parameters $\Lambda + \Delta \equiv \{\lambda_1 + \delta_1, \lambda_2 + \delta_2, \ldots \lambda_n + \delta_n\}$ which yield a model of higher log-likelihood. If we can find a procedure (a growth transformation) $\tau : \Lambda \to \Lambda + \Delta$ which takes one set of parameters as input and produces a new set as output which is not inferior, we can apply the transformation τ until we reach its fixed point[2] a stationary point for Λ.

[2]For those familiar with the *EM algorithm*, this is reminiscient of how one iterates until reaching a stationary point of the auxiliary function $Q(\Lambda' \mid \Lambda)$. Unlike in the EM algorithm, which guarantees only a locally optimal solution, the IIS algorithm converges to the unique maximum.

With respect to a given empirical distribution $\tilde{p}(x, y)$, the change in log-likelihood from the model Λ to the model $\Lambda + \Delta$ is

$$\begin{aligned} L_{\tilde{p}}(\Lambda + \Delta) - L_{\tilde{p}}(\Lambda) &= \sum_{x,y} \tilde{p}(x, y) \log p_{\Lambda'}(y \mid x) - \sum_{x,y} \tilde{p}(x, y) \log p_\Lambda(y \mid x) \\ &= \sum_{x,y} \tilde{p}(x, y) \sum_i \delta_i f_i(x, y) - \sum_x \tilde{p}(x) \log \frac{Z_{\Lambda'}(x)}{Z_\Lambda(x)} \end{aligned}$$

We now make use of the inequality $-\log \alpha \geq 1 - \alpha$ (true for all $\alpha > 0$), to establish a lower bound on the change in likelihood:

$$\begin{aligned} & L_{\tilde{p}}(\Lambda + \Delta) - L_{\tilde{p}}(\Lambda) \geq \\ & \quad \sum_{x,y} \tilde{p}(x, y) \sum_i \delta_i f_i(x, y) + 1 - \sum_x \tilde{p}(x) \frac{Z_{\Lambda'}(x)}{Z_\Lambda(x)} \\ &= \sum_{x,y} \tilde{p}(x, y) \sum_i \delta_i f_i(x, y) + 1 - \sum_x \tilde{p}(x) \frac{\sum_y \exp \sum_i (\Lambda_i + \delta_i) f_i(x, y)}{\sum_y \exp \sum_i \Lambda_i f_i(x, y)} \\ &= \underbrace{\sum_{x,y} \tilde{p}(x, y) \sum_i \delta_i f_i(x, y) + 1 - \sum_x \tilde{p}(x) \sum_y p_\Lambda(y \mid x) \exp \sum_i \delta_i f_i(x, y)}_{\text{Call this } \mathcal{A}(\Delta \mid \Lambda)} \end{aligned} \tag{5}$$

Since $L_{\tilde{p}}(\Lambda + \Delta) - L_{\tilde{p}}(\Lambda) \geq \mathcal{A}(\Delta \mid \Lambda)$, we know that if we can find a Δ for which $\mathcal{A}(\Delta \mid \Lambda) > 0$, then the model $\Lambda + \Delta$ is an improvement (in terms of log-likelihood) over the model Λ. A greedy strategy for optimizing the parameters of a log-linear model of the form (1), then, is to find the Δ which maximizes $\mathcal{A}(\Delta \mid \Lambda)$, set $\Lambda \leftarrow \Lambda + \Delta$, and repeat. So long as $\mathcal{A}(\Delta \mid \Lambda) > 0$, we're guaranteed an improvement in likelihood by this technique.

The straightforward approach, then, would be to maximize $\mathcal{A}(\Delta \mid \Lambda)$ with respect to each δ_i. Unfortunately, this doesn't quite work: differentiating $\mathcal{A}(\Delta \mid \Lambda)$ with respect to δ_i yields an equation containing $\{\Delta_1, \Delta_2, \ldots \Delta_n\}$; in other words, the contraint equations for the δ_i will be coupled. To get around this, we'll need the quantity

$$f^\#(x, y) \equiv \sum_i f_i(x, y)$$

If the f_i are binary-valued, $f^\#(x, y)$ has the simple interpretation of the number of features which "apply" (are non-zero) at x, y. We can rewrite $\mathcal{A}(\Delta \mid \Lambda)$ as

$$\mathcal{A}(\Delta \mid \Lambda) = \sum_{x,y} \tilde{p}(x, y) \sum_i \delta_i f_i(x, y) + 1 - \sum_x \tilde{p}(x) \sum_y p_\Lambda(y \mid x) \exp \left(f^\#(x, y) \sum_i \frac{\delta_i f_i(x, y)}{f^\#(x, y)} \right) \tag{6}$$

Notice that $\frac{f_i(x,y)}{f^\#(x,y)}$ is a p.d.f. over i, since it's always non-negative and sums to one over the natural numbers. This means we can apply Jensen's inequality—namely, for a p.d.f. $p(x)$,

$$\exp \sum_x p(x) q(x) \leq \sum_x p(x) \exp q(x)$$

to rewrite (6) as

$$\mathcal{A}(\Delta \mid \Lambda) \geq \underbrace{\sum_{x,y} \tilde{p}(x, y) \sum_i \delta_i f_i(x, y) + 1 - \sum_x \tilde{p}(x) \sum_y p_\Lambda(y \mid x) \sum_i \left(\frac{f_i(x, y)}{f^\#(x, y)} \right) e^{\delta_i f^\#(x,y)}}_{\text{Call this } \mathcal{B}(\Delta \mid \Lambda)} \tag{7}$$

$\mathcal{B}(\Delta \mid \Lambda)$ is a new, not as tight, lower bound on the change in log-likelihood. That is,

$$L_{\tilde{p}}(\Lambda + \Delta) - L_{\tilde{p}}(\Lambda) \geq \mathcal{B}(\Delta \mid \Lambda)$$

Differentiating $\mathcal{B}(\Delta \mid \Lambda)$ with respect to δ_i gives

$$\frac{\partial \mathcal{B}(\Delta)}{\partial \delta_i} = \sum_{x,y} \tilde{p}(x,y) f_i(x,y) - \sum_x \tilde{p}(x) \sum_y p_\Lambda(y \mid x) f_i(x,y) e^{\delta_i f^{\#}(x,y)} \quad (8)$$

What's nice about (8) is that δ_i appears alone, without any other free parameters. Thus we can solve for each of the n free parameters $\{\Delta_1, \Delta_2, \ldots \Delta_n\}$ individually by differentiating $\mathcal{B}(\Delta \mid \Lambda)$ with respect to each δ_i in turn. This suggests an iterative algorithm for finding the optimal values of $\lambda_1, \lambda_2 \ldots \lambda_n$:

IIS Algorithm

- Start with some (arbitrary) value for each λ_i
- Repeat until convergence:
  - Solve $\frac{\partial \mathcal{B}(\Delta)}{\partial \delta_i} = 0$ in (8) for δ_i
  - Set $\lambda_i \leftarrow \lambda_i + \delta_i$

References

[Be96] A. Berger, S. Della Pietra, V. Della Pietra (1996). A maximum entropy approach to natural language processing. *Computational Linguistics*, **22(1)**, 39—71.

[De97] S. Della Pietra, V. Della Pietra and J. Lafferty (1997). Inducing features of random fields. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, **19(4)**, 380—393.