

CRF-OPT: An Efficient High-Quality Conditional Random Field Solver*

Minmin Chen, Yixin Chen, and Michael R. Brent

Department of Computer Science and Engineering
 Washington University in St. Louis
 St. Louis, MO 63130, USA
 {mc15,chen,brent}@cse.wustl.edu

Abstract

Conditional random field (CRF) is a popular graphical model for sequence labeling. The flexibility of CRF poses significant computational challenges for training. Using existing optimization packages often leads to long training time and unsatisfactory results. In this paper, we develop CRF-OPT, a general CRF training package, to improve the efficiency and quality for training CRFs.

We propose two improved versions of the forward-backward algorithm that exploit redundancy and reduce the time by several orders of magnitudes. Further, we propose an exponential transformation that enforces sufficient step sizes for quasi-Newton methods. The technique improves the convergence quality, leading to better training results. We evaluate CRF-OPT on a gene prediction task on pathogenic DNA sequences, and show that it is faster and achieves better prediction accuracy than both the HMM models and the original CRF model without exponential transformation.

Introduction

Conditional random field (CRF) (Lafferty, McCallum, & Pereira 2001) is a major model for sequential data labeling. It significantly relaxes the independence assumptions of the hidden Markov model (HMM). However, the added flexibility of CRF greatly increases the optimization difficulties. In most applications, an optimization package is called to learn the CRF weights. We find that for large-scale problems this approach can be slow and result in unsatisfactory solution quality.

In this paper, we propose to enhance generic optimization solvers by a number of techniques that are designed specifically for improving the speed and quality of CRF training. First, we observe that, in gradient-based optimization, most of the time is spent on the evaluations of objective and gradients. For example, TAO (Benson *et al.* 2005), a state-of-the-art quasi-Newton solver, requires up to two days on a PC in a

gene prediction task with only 1615 bases. The real problem may have more than 100K bases. We find that, there are much computational redundancy in the standard implementation of the forward-backward algorithm. We propose two techniques that can improve the speed by eliminating the redundancy.

CRF training is a convex continuous optimization problem for which gradient-based search algorithms have solid theoretical convergence guarantees. However, we observe that, gradient-based search algorithms on CRF models usually terminate prematurely without reaching the actual optimal point. Moreover, different starting points usually lead to various different solutions other than the unique global optimum. We propose an exponential transformation technique that can force sufficient reduction in each step and achieve much better convergence quality.

Parameter Estimation for CRF

As shown in Figure 1(a), a CRF is a graphical model based on a graph $G = (V, E)$, where $Y = (Y_v)_{v \in V}$ is the set of hidden variables, and X is the set of observation variables. The random variables Y_v obey the Markov property: $p(Y_v | X, Y_u, \dots, Y_w) = p(Y_v | X, Y_w, w \sim v)$, where $w \sim v$ indicates that w and v are neighbors in G . Each clique in G defines a *feature* for the CRF.

In this paper, we focus on linear-chain CRF in which each feature only involves two consecutive hidden states as shown in Figure 1(b). A linear-chain CRF defines the conditional distribution of a label sequence \mathbf{y} , given the observation sequence \mathbf{x} , as

$$p(\mathbf{y} | \mathbf{x}) = \frac{1}{Z(\mathbf{x})} \exp \left\{ \sum_{t=1}^T \sum_{k=1}^F \lambda_k f_k(y_t, y_{t-1}, \mathbf{x}_t) \right\}, \quad (1)$$

where $\Lambda = \{\lambda_k\} \in \mathcal{R}^F$ is the weight vector, and $\{f_k(y_t, y_{t-1}, \mathbf{x}_t)\}_{k=1}^F$ is a set of feature functions, and $Z(\mathbf{x})$ is a normalization function

$$Z(\mathbf{x}) = \sum_{\mathbf{y}'} \exp \left\{ \sum_{t=1}^T \sum_{k=1}^F \lambda_k f_k(y'_t, y'_{t-1}, \mathbf{x}_t) \right\}. \quad (2)$$

*This work was supported by Microsoft New Faculty Fellowship and an ECPI grant from the Department of Energy. Copyright © 2008, Association for the Advancement of Artificial Intelligence (www.aaai.org). All rights reserved.

