Computing Conditional Feature Covariance in Non-Projective Tree Conditional Random Fields

University of Massachusetts Technical Report # UM-CS-2009-060

Gregory Druck
gdruck@cs.umass.edu

David A. Smith
dasmith@cs.umass.edu

Abstract

We present an $O(N^4)$ time algorithm for computing conditional feature covariance in edge-factored conditional random fields (CRFs) over non-projective dependency trees. Applications of this algorithm include more efficient Generalized Expectation (GE) parameter estimation.

1 Introduction

In this technical report we present an $O(N^4)$ time algorithm for computing conditional feature covariance in edge-factored conditional random fields (CRFs) over non-projective dependency trees. We derive the algorithm by computing the second derivative of the log-likelihood function of the CRF. This algorithm is applicable to Generalized Expectation (GE) parameter estimation, parameter estimation by minimizing the expectation of some edge-factored loss function, and entropy regularization.

1.1 Applications to GE Parameter Estimation

In previous work, Druck et al. use Generalized Expectation Criteria to estimate parameters of conditional random fields (CRFs) over non-projective dependency trees [1]. GE criteria express preferences on the value of a model expectation of a constraint feature function $g(\mathbf{x}, \mathbf{y})$. For example, a GE criterion $G(\theta)$ term may specify that the distance from some target expectation $\hat{g}$ should be minimized $G(\theta) = (\hat{g} - \mathrm{E}_{\tilde{p}(\mathbf{x})}[\mathrm{E}_{p(\mathbf{y}|\mathbf{x};\theta)}[g(\mathbf{x}, \mathbf{y})]])^2$, where $\mathbf{x}$ and $\mathbf{y}$ are input and output variables, θ are model parameters, and $\tilde{p}$ is an empirical distribution. Computing the gradient of a GE term with respect to the model parameters requires computing the model predicted covariance between model and constraint features. For edge-factored constraint features, this requires computing marginal distributions over pairs of edges $p(y_i = j, y_k = \ell|\mathbf{x}; \theta)$. Druck et al. [1] propose an $O(N^5)$ algorithm for computing two edge marginals, where N is the length of the sentence. In words, the algorithm first modifies the model's edge scores to require the edge $j \rightarrow i$ to be in the tree. Then, it uses the $O(N^3)$ algorithm of Smith and Smith [4] to compute the probabilities of all possible second edges $\ell \rightarrow k$ conditioned on the presence of the first edge $j \rightarrow i$, $p(y_k = \ell|y_i = j, \mathbf{x}; \theta)$. Finally, the conditional probabilities of $\ell \rightarrow k$ given $j \rightarrow i$ are multiplied by the marginal probabilities of $j \rightarrow i$, yielding the desired two edge marginal. For more details, readers are referred to [1] and [4].

The covariance computation described in Section 2 only requires one $O(N^3)$ matrix operation per sentence, which reduces the time complexity from $O(N^5)$ to $O(N^4)$.

1.2 Applications to Minimum Expected Loss Parameter Estimation

Suppose we have some loss function L that factors over edges $L(\mathbf{y}) = \sum_{i=1}^{N} l(y_i)$, and we want to minimize its expectation under an empirical distribution $\tilde{p}$ over $\mathbf{x}$ and the model distribution of $\mathbf{y}$ given $\mathbf{x}$, $p(\mathbf{y}|\mathbf{x}; \theta)$. Computing the gradient $\nabla_\theta \, \mathrm{E}_{\tilde{p}(\mathbf{x})}[\mathrm{E}_{p(\mathbf{y}|\mathbf{x};\theta)}[\sum_{i=1}^{N} l(y_i)]]$ requires $\nabla_\theta \, p(y_i|\mathbf{x}; \theta)$, which can computed with algorithm presented in Section 2.

For example, in a supervised setting, L could be the number of incorrect edges, $\sum_{i=1}^{N} 1 - \mathbb{I}(y_i = y_i^*)$, where $\mathbf{y}^*$ is the gold standard parse. To estimate parameters that minimize the *expected number of incorrect edges*, we need to compute $\nabla_\theta\ p(y_i^*|\mathbf{x};\theta)$.

1.3 Applications to Entropy Regularization

Entropy regularization [2] is a semi-supervised learning method that aims to minimize the entropy of model predictions on unlabeled data. If the model is a CRF and the regularizer uses the Shannon entropy[1], $\sum_{\mathbf{y}} p(\mathbf{y}|\mathbf{x};\theta) \log p(\mathbf{y}|\mathbf{x};\theta)$, it can be shown that the gradient with respect to θ_m is

$$\mathrm{E}_{\tilde{p}(\mathbf{x})}\Big[\sum_n \theta_n \Big(\mathrm{E}_{p(\mathbf{y}|\mathbf{x};\theta)}[f_n(\mathbf{x},\mathbf{y}) f_m(\mathbf{x},\mathbf{y})] - \mathrm{E}_{p(\mathbf{y}|\mathbf{x};\theta)}[f_n(\mathbf{x},\mathbf{y})]\mathrm{E}_{p(\mathbf{y}|\mathbf{x};\theta)}[f_m(\mathbf{x},\mathbf{y}]\Big)\Big].$$

Note that this expression contains the covariance between f_n and f_m, which can be computed with the algorithm presented in Section 2.

2 $O(N^4)$ Time Feature Covariance Computation

Because CRFs are exponential family models, the second derivative of the log-likelihood function with respect to the model parameters is equal to the covariance between model feature functions. We will use this fact to derive an $O(N^4)$ algorithm for computing the covariance[2]. We start by reviewing the first partial derivative of the partition function $Z_{\mathbf{x},\theta}$. Smith and Smith [4] (Equation 13) show that

$$\frac{\partial \log Z_{\mathbf{x},\theta}}{\partial \theta_m} = \sum_{i=1}^{N}\sum_{j=0}^{N} s_{\mathbf{x},\theta}(i,j) f_m(\mathbf{x}, x_i, x_j)\big([\mathbf{K}^{-1}_{\mathbf{x},\theta}]_{i,i} - [\mathbf{K}^{-1}_{\mathbf{x},\theta}]_{i,j}\big),$$

where $\mathbf{x}$ is the input sentence, N is the length of the sentence, f_m is a feature function that considers the entire input and the edge $j \rightarrow i$, θ are model parameters, $s_{\mathbf{x},\theta}(i,j) = \exp(\theta \cdot \vec{f}(\mathbf{x}, x_i, x_j))$ is the score of edge $j \rightarrow i$, and $\mathbf{K}_{\mathbf{x},\theta} \in \mathbb{R}^{N \times N}$ is the Kirchoff matrix:

$$[\mathbf{K}_{\mathbf{x},\theta}]_{j,i} = \begin{cases} \sum_{k \in \{0,\ldots,n\}: k \neq i} s_{\mathbf{x},\theta}(i,k) & : i = j \\ -s_{\mathbf{x},\theta}(i,j) & : i \neq j \end{cases}$$

In general we use the notation $[\mathbf{A}]_{i,j}$ to denote the value at row i and column j in matrix $\mathbf{A}$. We assume $s_{\mathbf{x},\theta}(i,i) = 0$. Note that index 0 corresponds to the root.

The second partial derivative of the partition function (other terms in the first partial derivative of the log-likelihood are constant) with respect to θ_m and θ_n (the parameter for feature function f_n) is

$$\begin{aligned}\frac{\partial^2 \log Z_{\mathbf{x},\theta}}{\partial \theta_m \partial \theta_n} &= \sum_{k=1}^{N}\sum_{\ell=0}^{N} \frac{\partial^2 \log Z_{\mathbf{x},\theta}}{\partial \theta_m \partial s_{\mathbf{x},\theta}(k,\ell)} \frac{\partial s_{\mathbf{x},\theta}(k,\ell)}{\partial \theta_n} \\ &= \sum_{k=1}^{N}\sum_{\ell=0}^{N} s_{\mathbf{x},\theta}(k,\ell) f_n(\mathbf{x}, x_k, x_\ell) \frac{\partial^2 \log Z_{\mathbf{x},\theta}}{\partial \theta_m \partial s_{\mathbf{x},\theta}(k,\ell)}. \end{aligned} \quad (1)$$

If $j \rightarrow i$ and $\ell \rightarrow k$ are not the same edge, $i \neq k \ \vee j \neq \ell$, then

$$\frac{\partial^2 \log Z_{\mathbf{x},\theta}}{\partial \theta_m \partial s_{\mathbf{x},\theta}(k,\ell)} = \sum_{i=1}^{N}\sum_{j=0}^{N} s_{\mathbf{x},\theta}(i,j) f_m(\mathbf{x}, x_i, x_j) \frac{\partial}{\partial s_{\mathbf{x},\theta}(k,\ell)}\big([\mathbf{K}^{-1}_{\mathbf{x},\theta}]_{i,i} - [\mathbf{K}^{-1}_{\mathbf{x},\theta}]_{i,j}\big).$$

[1] Smith and Eisner [3] previously applied Renyi entropy regularization to dependency parsing.

[2] Note that although we derive the covariance between features that appear in the model, the resulting expression can also be used to compute the model predicted covariance between two arbitrary edge-factored features (that may or may not be in the model), we only need to compute the inverse of one matrix per sentence to compute the GE gradient. This is important because GE does not require constraint features to be model features.

For an arbitrary matrix $\mathbf{A}$, the derivative with respect to t of its inverse $\mathbf{A}^{-1}$ is

$$\frac{\partial \mathbf{A}^{-1}}{\partial t} = -\mathbf{A}^{-1}\frac{\partial \mathbf{A}}{\partial t}\mathbf{A}^{-1}.$$

The derivative of $\mathbf{A}$ with respect to a single cell in $[\mathbf{A}]_{k,l}$ is a matrix with a 1 at k, ℓ and zeros elsewhere. Therefore, the derivative of a cell in the inverse $[\mathbf{A}^{-1}]_{i,j}$ with respect to cell $[\mathbf{A}]_{k,\ell}$ is the product of two cells in the inverse.

$$\frac{\partial [\mathbf{A}^{-1}]_{i,j}}{\partial [\mathbf{A}]_{k,\ell}} = -[\mathbf{A}^{-1}]_{i,k}[\mathbf{A}^{-1}]_{\ell,j}.$$

Using this fact, we have

$$\frac{[\mathbf{K}^{-1}_{\mathbf{x},\theta}]_{i,i}}{\partial s_{\mathbf{x},\theta}(k,\ell)} = [\mathbf{K}^{-1}_{\mathbf{x},\theta}]_{i,\ell}[\mathbf{K}^{-1}_{\mathbf{x},\theta}]_{k,i} - [\mathbf{K}^{-1}_{\mathbf{x},\theta}]_{i,k}[\mathbf{K}^{-1}_{\mathbf{x},\theta}]_{k,i},$$
$$\frac{[\mathbf{K}^{-1}_{\mathbf{x},\theta}]_{i,j}}{\partial s_{\mathbf{x},\theta}(k,\ell)} = [\mathbf{K}^{-1}_{\mathbf{x},\theta}]_{i,\ell}[\mathbf{K}^{-1}_{\mathbf{x},\theta}]_{k,j} - [\mathbf{K}^{-1}_{\mathbf{x},\theta}]_{i,k}[\mathbf{K}^{-1}_{\mathbf{x},\theta}]_{k,j}.$$

Putting these terms together, we have (when $i \neq k \ \vee j \neq \ell$)

$$\begin{aligned}\frac{\partial}{\partial s_{\mathbf{x},\theta}(k,\ell)}\big([\mathbf{K}^{-1}_{\mathbf{x},\theta}]_{i,i} - [\mathbf{K}^{-1}_{\mathbf{x},\theta}]_{i,j}\big) =& [\mathbf{K}^{-1}_{\mathbf{x},\theta}]_{i,\ell}[\mathbf{K}^{-1}_{\mathbf{x},\theta}]_{k,i} - [\mathbf{K}^{-1}_{\mathbf{x},\theta}]_{i,k}[\mathbf{K}^{-1}_{\mathbf{x},\theta}]_{k,i} \\ & -[\mathbf{K}^{-1}_{\mathbf{x},\theta}]_{i,\ell}[\mathbf{K}^{-1}_{\mathbf{x},\theta}]_{k,j} + [\mathbf{K}^{-1}_{\mathbf{x},\theta}]_{i,k}[\mathbf{K}^{-1}_{\mathbf{x},\theta}]_{k,j}.\end{aligned}$$

$$\begin{aligned}\frac{\partial^2 \log Z_{\mathbf{x},\theta}}{\partial\theta_m \partial s_{\mathbf{x},\theta}(k,\ell)} =& \sum_{i=1}^{N}\sum_{j=0}^{N} s_{\mathbf{x},\theta}(i,j) f_m(\mathbf{x}, x_i, x_j)\Big([\mathbf{K}^{-1}_{\mathbf{x},\theta}]_{i,\ell}[\mathbf{K}^{-1}_{\mathbf{x},\theta}]_{k,i} - [\mathbf{K}^{-1}_{\mathbf{x},\theta}]_{i,k}[\mathbf{K}^{-1}_{\mathbf{x},\theta}]_{k,i} \\ & -[\mathbf{K}^{-1}_{\mathbf{x},\theta}]_{i,\ell}[\mathbf{K}^{-1}_{\mathbf{x},\theta}]_{k,j} + [\mathbf{K}^{-1}_{\mathbf{x},\theta}]_{i,k}[\mathbf{K}^{-1}_{\mathbf{x},\theta}]_{k,j}\Big).\end{aligned}$$

If $j \to i$ and $\ell \to k$ are the same edge, $i = k \ \wedge j = \ell$, then

$$\begin{aligned}\frac{\partial^2 \log Z_{\mathbf{x},\theta}}{\partial\theta_m \partial s_{\mathbf{x},\theta}(i,j)} =& \sum_{i=1}^{N}\sum_{j=0}^{N}\big([\mathbf{K}^{-1}_{\mathbf{x},\theta}]_{i,i} - [\mathbf{K}^{-1}_{\mathbf{x},\theta}]_{i,j}\big)\frac{\partial}{\partial s_{\mathbf{x},\theta}(i,j)} s_{\mathbf{x},\theta}(i,j) f_m(\mathbf{x}, x_i, x_j) \\ & + s_{\mathbf{x},\theta}(i,j) f_m(\mathbf{x}, x_i, x_j)\frac{\partial}{\partial s_{\mathbf{x},\theta}(i,j)}\big([\mathbf{K}^{-1}_{\mathbf{x},\theta}]_{i,i} - [\mathbf{K}^{-1}_{\mathbf{x},\theta}]_{i,j}\big). \\ =& \sum_{i=1}^{N}\sum_{j=0}^{N} f_m(\mathbf{x}, x_i, x_j)\big([\mathbf{K}^{-1}_{\mathbf{x},\theta}]_{i,i} - [\mathbf{K}^{-1}_{\mathbf{x},\theta}]_{i,j}\big) \\ & - s_{\mathbf{x},\theta}(i,j) f_m(\mathbf{x}, x_i, x_j)([\mathbf{K}^{-1}_{\mathbf{x},\theta}]_{i,i} - [\mathbf{K}^{-1}_{\mathbf{x},\theta}]_{i,j})^2\end{aligned}$$

Substituting back into Equation 1 gives us the covariance

$$\begin{aligned}\frac{\partial^2 \log Z_{\mathbf{x},\theta}}{\partial\theta_m \partial\theta_n} =& \sum_{k=1}^{N}\sum_{\ell=0}^{N}\sum_{i=1}^{N}\sum_{j=0}^{N} s_{\mathbf{x},\theta}(k,\ell) f_n(\mathbf{x}, x_k, x_\ell) s_{\mathbf{x},\theta}(i,j) f_m(\mathbf{x}, x_i, x_j) \times \\ & \Big([\mathbf{K}^{-1}_{\mathbf{x},\theta}]_{i,\ell}[\mathbf{K}^{-1}_{\mathbf{x},\theta}]_{k,i} - [\mathbf{K}^{-1}_{\mathbf{x},\theta}]_{i,k}[\mathbf{K}^{-1}_{\mathbf{x},\theta}]_{k,i} - [\mathbf{K}^{-1}_{\mathbf{x},\theta}]_{i,\ell}[\mathbf{K}^{-1}_{\mathbf{x},\theta}]_{k,j} + [\mathbf{K}^{-1}_{\mathbf{x},\theta}]_{i,k}[\mathbf{K}^{-1}_{\mathbf{x},\theta}]_{k,j}\Big) \\ & + \sum_{i=1}^{N}\sum_{j=0}^{N} s_{\mathbf{x},\theta}(i,j) f_n(\mathbf{x}, x_i, x_j) f_m(\mathbf{x}, x_i, x_j)\big([\mathbf{K}^{-1}_{\mathbf{x},\theta}]_{i,i} - [\mathbf{K}^{-1}_{\mathbf{x},\theta}]_{i,j}\big),\end{aligned} \tag{2}$$

where the last line accounts for the extra term introduced when $i = k \ \wedge j = \ell$.

Equation 2 computes the covariance between two edge-factored feature functions f_m and f_n. Notice that this computation only requires cells from the inverse Kirchoff matrix. Computing the inverse Kirchoff

matrix takes $O(N^3)$ time for each sentence. The time required to compute the complete covariance is then $O(N^4)$, the time required to consider all possible pairs of edges.

Finally, we provide an expression for computing the two edge marginal itself. The covariance can also be written as

$$\mathrm{E}_{\tilde{p}(\mathbf{x})}[\mathrm{E}_{p(y_i=j,y_k=\ell|\mathbf{x};\theta)}[f_m(\mathbf{x},x_i,x_j)f_n(\mathbf{x},x_k,x_\ell)] - \mathrm{E}_{p(y_i=j|\mathbf{x};\theta)}[f_m(\mathbf{x},x_i,x_j)]\mathrm{E}_{p(y_k=\ell|\mathbf{x};\theta)}[f_n(\mathbf{x},x_k,x_\ell)]].$$

Therefore, it can be shown that the two edge marginal is

$$\begin{aligned} p(y_i=j,y_k=\ell|\mathbf{x},\theta) =& [\mathbf{K}^{-1}_{\mathbf{x},\theta}]_{i,\ell}[\mathbf{K}^{-1}_{\mathbf{x},\theta}]_{k,i} - [\mathbf{K}^{-1}_{\mathbf{x},\theta}]_{i,k}[\mathbf{K}^{-1}_{\mathbf{x},\theta}]_{k,i} - [\mathbf{K}^{-1}_{\mathbf{x},\theta}]_{i,\ell}[\mathbf{K}^{-1}_{\mathbf{x},\theta}]_{k,j} + [\mathbf{K}^{-1}_{\mathbf{x},\theta}]_{i,k}[\mathbf{K}^{-1}_{\mathbf{x},\theta}]_{k,j} \\ &+[\mathbf{K}^{-1}_{\mathbf{x},\theta}]_{i,i}[\mathbf{K}^{-1}_{\mathbf{x},\theta}]_{k,k} - [\mathbf{K}^{-1}_{\mathbf{x},\theta}]_{i,i}[\mathbf{K}^{-1}_{\mathbf{x},\theta}]_{k,\ell} - [\mathbf{K}^{-1}_{\mathbf{x},\theta}]_{i,j}[\mathbf{K}^{-1}_{\mathbf{x},\theta}]_{k,k} + [\mathbf{K}^{-1}_{\mathbf{x},\theta}]_{i,j}[\mathbf{K}^{-1}_{\mathbf{x},\theta}]_{k,\ell}. \quad (3) \end{aligned}$$

References

[1] G. Druck, G. Mann, and A. McCallum. Semi-supervised learning of dependency parsers using generalized expectation criteria. In *ACL-IJCNLP*, pages 360–368, 2009.

[2] Y. Grandvalet and Y. Bengio. Semi-supervised learning by entropy minimization. In *Advances in Neural Information Processing Systems 17*, pages 529–536. MIT Press, 2005.

[3] D. A. Smith and J. Eisner. Bootstrapping feature-rich dependency parsers with entropic priors. In *EMNLP-CoNLL*, pages 667–677, 2007.

[4] D. A. Smith and N. A. Smith. Probabilistic models of nonprojective dependency trees. In *EMNLP-CoNLL*, pages 132–140, 2007.