

Newton update in L₂-norm random tree approximation

Hongyu Su

Helsinki Institute for Information Technology HIIT Department of Computer Science Aalto University

May 20, 2015

Preliminaries

- $ightharpoonup \mathcal{X}$ is an arbitrary input space, $\mathbf{x} \in \mathcal{X}$.
- $ightharpoonup \mathcal{Y}$ is an output space of a set of ℓ -dimensional *multilabels*

$$\mathbf{y}=(y_1,\cdots,y_\ell)\in \mathbf{\mathcal{Y}}.$$

- y_i is a microlabel and $y_i \in \{1, \dots, r_i\}, r_i \in \mathbb{Z}$.
- ▶ For example, multilabel binary classification $y_i \in \{-1, +1\}$.
- ▶ Training examples are sampled from $(x, y) \in \mathcal{X} \times \mathcal{Y}$.
- **Each** example (x, y) is mapped into a joint feature space $\phi(x, y)$.
- **w** is the weight vector in the joint feature space.
- ▶ Define a linear score function $F(\mathbf{w}, \mathbf{x}, \mathbf{y}) = \langle \mathbf{w}, \phi(\mathbf{x}, \mathbf{y}) \rangle$.
- ▶ The prediction $y_w(x)$ of an input x is the multilabel y that maximizes the score function

$$\mathbf{y}_{\mathbf{w}}(\mathbf{x}) = \underset{\mathbf{y} \in \mathcal{Y}}{\operatorname{argmax}} \langle \mathbf{w}, \boldsymbol{\phi}(\mathbf{x}, \mathbf{y}) \rangle. \tag{1}$$

 (1) is called *inference* problem which is NP-hard for most output feature maps.



Markov network

- We assume that the output feature map ϕ is a potential function on a Markov network G = (E, V).
- ▶ *G* is a complete graph with $|V| = \ell$ nodes and $|E| = \frac{\ell(\ell-1)}{2}$ undirected edges.
- $ightharpoonup \varphi(x)$ is the input feature map, e.g., bag-of-words feature of an example x.
- $lackbox{}\psi(y)$ is the output feature map which is a collection of edges and labels

$$\varphi(\mathbf{y}) = (u_e)_{e \in E}, u_e \in \{-1, +1\}^2.$$

lacktriangle The joint feature is the Kronecker product of $oldsymbol{arphi}({ t x})$ and $oldsymbol{\psi}({ t y})$

$$\phi(\mathsf{x},\mathsf{y}) = (\phi_e(\mathsf{x},\mathsf{y}))_{e \in E} = (\varphi(\mathsf{x}) \otimes \psi_e(\mathsf{y}_e))_{e \in E}.$$

The score function is

$$F(\mathbf{w}, \mathbf{x}, \mathbf{y}) = \langle \mathbf{w}, \phi(\mathbf{x}, \mathbf{y}) \rangle = \sum_{e \in F} \langle \mathbf{w}_e, \phi_e(\mathbf{x}, \mathbf{y}_e) \rangle.$$



Inference in terms of all spanning trees

ightharpoonup Solving the following inference problem on a complete graph is \mathcal{NP} -hard

$$y_{w}(x) = \mathop{argmax}_{y \in \mathcal{Y}} F(w, x, y) = \mathop{argmax}_{y \in \mathcal{Y}} \sum_{e \in E} \langle w_{e}, \phi_{e}(x, y_{e}) \rangle.$$

- ▶ For a complete graph, there are $\ell^{\ell-2}$ unique spanning trees.
- We can write $F(\mathbf{w}, \mathbf{x}, \mathbf{y})$ as a conic combination of all spanning trees

$$\begin{split} F(\mathbf{w}, \mathbf{x}, \mathbf{y}) &= \mathop{\mathbf{E}}_{T \in U(G)} a_T \langle \mathbf{w}_T, \phi_T(\mathbf{x}, \mathbf{y}) \rangle \\ & \mathop{\mathbf{E}}_{T \in U(G)} a_T^2 = 1, \mathop{\mathbf{E}}_{T \in U(G)} a_T < 1. \end{split}$$

- ▶ U(G) is the uniform distribution over $\ell^{\ell-2}$ spanning trees.
- ▶ There is a exponential dependency on the number of spanning trees.

A sample of *n* spanning trees

▶ Instead of using all spanning trees, we can just use *n* spanning trees

$$F_{\mathcal{T}}(\mathbf{w}, \mathbf{x}, \mathbf{y}) = \frac{1}{n} \sum_{i=1}^{n} a_{\mathcal{T}_i} \langle \mathbf{w}_{\mathcal{T}_i}, \boldsymbol{\phi}_{\mathcal{T}_i}(\mathbf{x}, \mathbf{y}) \rangle$$
$$\frac{1}{n} \sum_{i=1}^{n} a_{\mathcal{T}_i}^2 = 1, \frac{1}{n} \sum_{i=1}^{n} a_{\mathcal{T}_i} < 1.$$

When

$$n \geq rac{\ell^2}{\epsilon^2} (rac{1}{16} + rac{1}{2} \ln rac{8\sqrt{n}}{\delta}),$$

with high probability, we have $|F_{\mathcal{T}}(\mathbf{w}, \mathbf{x}, \mathbf{y}) - F(\mathbf{w}, \mathbf{x}, \mathbf{y})| \leq \epsilon$.

- ▶ A sample of $n \in \Theta(\ell^2/\delta^2)$ random spanning tree is sufficient to estimate the score function.
- Margin achieved by $F(\mathbf{w}, \mathbf{x}, \mathbf{y})$ is also preserved by the sample of n random spanning trees $F_{\mathcal{T}}(\mathbf{w}, \mathbf{x}, \mathbf{y})$.

Optimization problem

The primal optimization problem is defined as

$$\begin{aligned} & \min_{\mathbf{w}_{T_i}, \xi_i} & \frac{1}{2} \sum_{i=1}^n ||\mathbf{w}_{T_i}||^2 + C \sum_{k=1}^m \xi_k \\ & \text{s.t.} & \frac{1}{\sqrt{n}} \sum_{i=1}^n \left\langle \mathbf{w}_{T_i}, \boldsymbol{\phi}_{T_t}(\mathbf{x}_k, \mathbf{y}_k) \right\rangle - \max_{\mathbf{y} \neq \mathbf{y}_k} \frac{1}{\sqrt{n}} \sum_{i=1}^n \left\langle \mathbf{w}_{T_t}, \boldsymbol{\phi}_{T_i}(\mathbf{x}_k, \mathbf{y}) \right\rangle \geq 1 - \xi_k, \\ & \xi_k \geq 0, \forall \ k \in \{1, \dots, m\}. \end{aligned}$$

The marginalized dual problem is defined as

$$\begin{aligned} \max_{\mu \in \mathcal{M}} \quad & \sum_{i=1}^{n} \left(\mu_{T_i} \ell_{T_i} - \frac{1}{2} \mu_{T_i} \mathsf{K}_{T_i} \mu_{T_i} \right) \\ \text{s.t.} \quad & \sum_{u_e} \mu_{T_i,e}(u_e) \leq C. \end{aligned}$$

Optimization algorithm for a single spanning tree

- ► We can solve the optimization problem efficiently for each individual spanning tree.
- ► The algorithm iterates over all training example until convergence.
- For the kth iteration:
 - 1. Obtain the solution of the jth example in kth iteration $\mu_{T_i}^k(j)$.
 - 2. Compute the gradient $g_{T_i}^k(j) = \ell_{T_i}(j) K_{T_i}\mu_{T_i}^k(j)$.
 - 3. Compute the update direction

$$\hat{\mu}_{T_i}^{k+1}(j) = \operatorname*{argmax}_{\mu \in \mathcal{M}} \mu^{\mathsf{T}} g_{T_i}^k(j).$$

- 4. Compute the difference $\Delta \mu_{T_i}^{k+1}(j) = \hat{\mu}_{T_i}^{k+1}(j) \hat{\mu}_{T_i}^{k}(j)$.
- 5. Perform the update $\mu_{T_i}^{k+1}(j) = \mu_{T_i}^k(j) + \tau \Delta \mu_{T_i}^{k+1}(j)$
- ▶ The step size along the update direction τ is given by the exact line search.

$$\frac{\partial \left(f(\mu_{T_i}^{k+1}(j)) - f(\mu_{T_i}^{k}(j)) \right)}{\partial \tau} = 0, 0 \le \tau \le 1.$$

κ -best inference for a collection of n spanning trees

- ▶ The algorithm iterates over all training example until convergence.
- For the kth iteration:
 - 1. Obtain the solutions of the jth example over all trees $(\mu_{T_i}^k(j))_{i=1}^n$.
 - 2. Compute the gradients over all trees $(g_{T_i}^k(j))_{i=1}^n$.
 - 3. Compute the update directions

$$\hat{\mu}_{T_i}^{k+1}(j) = \operatorname*{argmax}_{\mu \in \mathcal{M}} \mu^{\mathsf{T}} g_{T_i}^k(j), \ orall i.$$

4. Compute the best direction

$$\tilde{\mu}_{T_i}^{k+1}(j) = \underset{\mu \in (\hat{\mu}_{T_i}^{k+1}(j))_{i=1}^n}{\operatorname{argmax}} \sum_{i=1}^n \mu^{\mathsf{T}} g_{T_i}^k(j)$$

- 5. Compute the difference $\Delta \mu_{T_i}^{k+1}(j) = \hat{\mu}_{T_i}^{k+1}(j) \hat{\mu}_{T_i}^{k}(j), \forall i$.
- 6. Perform the update $\mu_T^{k+1}(j) = \mu_T^k(j) + \tau \Delta \mu_T^{k+1}(j), \forall i$.
- The step size along the update direction τ is given by the exact line search.

$$\frac{\partial \left(\sum_{i=1}^n f(\mu_{T_i}^{k+1}(j)) - \sum_{i=1}^n f(\mu_{T_i}^k(j))\right)}{\partial \tau} = 0, 0 \le \tau \le 1.$$



Update with multiple directions

- ▶ The algorithm iterates over all training example until convergence.
- For the kth iteration:
 - 1. Obtain the solutions of the jth example over all trees $(\mu_{T_i}^k(j))_{i=1}^n$.
 - 2. Compute the gradients over all trees $(g_{T_i}^k(j))_{i=1}^n$.
 - 3. Compute the update directions

$$\hat{\mu}_{T_i}^{k+1}(j) = \operatorname*{argmax}_{\mu \in \mathcal{M}} \mu^{\mathsf{T}} g_{T_i}^k(j), \, orall i.$$

4. Define a conic combination of update directions

$$\tilde{\mu}_G^{k+1}(j) \leftarrow \sum_{i=1}^n \tau_i \hat{\mu}_{T_i}^{k+1}(j)$$

- 5. Compute the difference $\Delta \mu_{T_i}^{k+1}(j) = \hat{\mu}_{T_i}^{k+1}(j) \hat{\mu}_{T_i}^{k}(j), \forall i$.
- 6. Perform the update $\mu_{T_i}^{k+1}(j) = \mu_{T_i}^k(j) + \tau \Delta \mu_{T_i}^{k+1}(j), \forall i$.
- ▶ The step size along the update direction τ is given by the exact line search.

$$\frac{\partial \left(\sum_{i=1}^n f(\mu_{T_i}^{k+1}(j)) - \sum_{i=1}^n f(\mu_{T_i}^k(j))\right)}{\partial \tau} = 0, 0 \le \tau \le 1.$$

Conclusions

