Newton update in L_2-norm random tree approximation

Hongyu Su

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

May 22, 2015

Preliminaries

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

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

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

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

- (1) is called *inference* problem which is $\mathcal{NP}$-hard for most output feature maps.

Markov network

- We assume that the joint 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.

- G models all pairwise correlations.

- $\varphi(\mathbf{x})$ is the input feature map, e.g., bag-of-words feature of an example $\mathbf{x}$.

- $\psi(\mathbf{y})$ is the output feature map which is a collection of edges and labels

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

- The joint feature is the Kronecker product of $\varphi(\mathbf{x})$ and $\psi(\mathbf{y})$

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

- The score function can be factorized by the complete graph G

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

Inference in terms of all spanning trees

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

$$\mathbf{y_w}(\mathbf{x}) = \underset{\mathbf{y} \in \mathcal{Y}}{\operatorname{argmax}} \, F(\mathbf{w}, \mathbf{x}, \mathbf{y}) = \underset{\mathbf{y} \in \mathcal{Y}}{\operatorname{argmax}} \sum_{e \in E} \langle \mathbf{w}_e, \phi_e(\mathbf{x}, \mathbf{y}_e) \rangle.$$

▶ 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

$$F(\mathbf{w}, \mathbf{x}, \mathbf{y}) = \underset{T \in U(G)}{\mathbf{E}} a_T \langle \mathbf{w}_T, \phi_T(\mathbf{x}, \mathbf{y}) \rangle$$

$$\underset{T \in U(G)}{\mathbf{E}} a_T^2 = 1, \quad \underset{T \in U(G)}{\mathbf{E}} a_T < 1.$$

▶ $U(G)$ is the uniform distribution over $\ell^{\ell-2}$ spanning trees.

▶ The number of spanning trees is exponentially dependent on the number of nodes ℓ.

A sample of n spanning trees

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

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

$$\frac{1}{n} \sum_{i=1}^{n} a_{T_i}^2 = 1, \frac{1}{n} \sum_{i=1}^{n} a_{T_i} < 1.$$

▶ When

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

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

▶ 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

$$\min_{\mathbf{w}_{T_i}, \xi_i} \quad \frac{1}{2} \sum_{i=1}^{n} ||\mathbf{w}_{T_i}||^2 + C \sum_{k=1}^{m} \xi_k$$

$$\text{s.t.} \quad \sum_{i=1}^{n} \langle \mathbf{w}_{T_i}, \Delta \boldsymbol{\phi}_{T_t}(\mathbf{x}_k, \mathbf{y}_k) \rangle \geq \ell_{T_i,k} - \xi_k,$$

$$\xi_k \geq 0, \forall \ k \in \{1, \dots, m\}.$$

- The marginalized dual problem is defined as

$$\max_{\mu \in \mathcal{M}} \quad \sum_{i=1}^{n} \left(\mu_{T_i} \boldsymbol{\ell}_{T_i} - \frac{1}{2} \mu_{T_i} K_{T_i}^{\Delta \boldsymbol{\phi}} \mu_{T_i} \right)$$

$$\text{s.t.} \quad \sum_{u_e} \mu_{T_i,e}(u_e) \leq C.$$

Optimization algorithm for a single spanning tree T_i

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

    $$\mu_{T_i}^{k,*} = \underset{\mu \in \mathcal{M}}{\operatorname{\mathbf{argmax}}} \, \mu^{\intercal} g_{T_i}^k.$$

    4. Compute the difference $\Delta\mu_{T_i}^k = \mu_{T_i}^k - \mu_{T_i}^{k,*}$.
    5. Compute a step size τ.
    6. Perform the update $\mu_{T_i}^{k+1} = \mu_{T_i}^k + \tau\Delta\mu_{T_i}^k$
- The step size along the update direction τ is given by the line search.

$$\underset{\tau}{\mathbf{max}} \quad f(\mu_{T_i}^k + \tau\Delta\mu_{T_i}^k)$$

$$\mathbf{s.t.} \quad 0 \le \tau \le 1.$$

κ-best inference for a collection of n spanning trees

▶ The algorithm iterates over all training example until convergence.

▶ $\mu : \mu(j)$, and $g : g(j)$.

▶ For the kth iteration:

1. Obtain the solutions of the jth example over all trees $(\mu_{T_i}^k)_{i=1}^n$.
2. Compute the gradients over all trees $(g_{T_i}^k)_{i=1}^n$.
3. Compute the update directions

$$\mu_{T_i}^{k,*} = \underset{\mu \in \mathcal{M}}{\operatorname{argmax}} \, \mu^\mathsf{T} g_{T_i}^k, \, \forall i.$$

4. Compute the best direction

$$\mu_T^{k,*} = \underset{\mu \in (\mu_{T_i}^{k,*})_{i=1}^n}{\operatorname{argmax}} \sum_{i=1}^n \mu^\mathsf{T} g_{T_i}^k$$

5. Compute the difference $\Delta\mu_{T_i}^k = \mu_{T_i}^k - \mu_T^{k,*}, \, \forall i.$
6. Compute the step size τ.
7. Perform the update $\mu_{T_i}^{k+1} = \mu_{T_i}^k + \tau\Delta\mu_{T_i}^k, \, \forall i.$

Line search to get the step size

▶ The step size along the update direction τ is given by the exact line search.

$$\max_{\tau} \quad \sum_{i=1}^{n} f(\mu_{T_i}^k + \tau \Delta \mu_{T_i}^k)$$

$$\text{s.t.} \quad 0 \leq \tau \leq 1.$$

Update with multiple directions

- The algorithm iterates over all training example until convergence.
- $\mu : \mu(j)$, and $g : g(j)$.
- For the kth iteration:
  1. Obtain the solutions of the jth example over all trees $(\mu_{T_i}^k)_{i=1}^n$.
  2. Compute the gradients over all trees $(g_{T_i}^k)_{i=1}^n$.
  3. Compute local update direction from each spanning tree

$$\mu_{T_i}^{k,*} = \underset{\mu \in \mathcal{M}}{\textbf{argmax}} \, \mu^{\mathsf{T}} g_{T_i}^k, \, \forall i.$$

  4. Project local directions into global directions

$$\mu_{T_i}^{G,k,*} \leftarrow \mu_{T_i}^{k,*}, \, \forall i.$$

  5. Define a conic combination of update directions

$$\Delta \mu^{G,k} = \sum_{i=1}^n \tau_i \Delta \mu_{T_i}^{G,k,*} = \sum_{i=1}^n \tau_i \left(\mu^{G,k} - \mu_{T_i}^{G,k,*} \right), 0 \le \tau_i \le 1, \forall i.$$

  6. Perform the update $\mu^{G,k+1} = \mu^{G,k} + \Delta \mu^{G,k+1}$.
  7. Project the global solution on spanning trees $(\mu_{T_i}^{k+1})_{i=1}^n \leftarrow \mu^{G,k+1}$.

Newton method to compute τ

- We want to find τ that maximize the objective function given the update

$$\max_{\tau} \quad f(\mu^{G,k} + \Delta\mu^{G,k+1})$$
$$\textbf{s.t.} \quad 0 \leq \tau_i \leq 1.$$

- The objective is quadratic with respect to τ.
- We use Newton method to find τ that maximize the objective.
- τ is projected into the feasible region.

Compute duality gap

- ▶ We use duality gap to measure the progress of the optimization.
- ▶ Primal and dual objective function

$$f(\mathbf{w}) = \frac{1}{2}||\mathbf{w}||^2 + C \sum_{i=1}^{m} \left(\ell_i - \langle \mathbf{w}, \Delta\phi(\mathbf{x}_i, \mathbf{y}_i)\rangle\right)$$

$$g(\boldsymbol{\alpha}) = \sum_{i=1}^{m} \alpha_i \ell_i - \frac{1}{2}\sum_{i=1}^{m}\sum_{j=1}^{m} \alpha_i K^{\Delta\phi}(\mathbf{x}_i, \mathbf{y}_i; \mathbf{x}_j, \mathbf{y}_j)\alpha_j$$

- ▶ $\max_{\boldsymbol{\alpha}} g(\boldsymbol{\alpha}) \leq \min_{\mathbf{w}} f(\mathbf{w})$, minimum gap when optimal.
- ▶ Duality gap at $\boldsymbol{\alpha}^k$

$$f(\mathbf{w}^k) - g(\boldsymbol{\alpha}^k) = C\left(\boldsymbol{\ell} - K^{\Delta\phi}\boldsymbol{\alpha}^k\right) - \boldsymbol{\alpha}^k\left(\boldsymbol{\ell} - K^{\Delta\phi}\boldsymbol{\alpha}^k\right)$$

$$= C^{\mathsf{T}}\nabla g(\boldsymbol{\alpha}^k) - \boldsymbol{\alpha}^{k\mathsf{T}}\nabla g(\boldsymbol{\alpha}^k)$$

1. Estimate the dual objective function using a linear approximation ∇g.
2. Dual objective value at $\boldsymbol{\alpha}^k$ is computed by $\boldsymbol{\alpha}^{k\mathsf{T}}\nabla g(\boldsymbol{\alpha}^k)$.
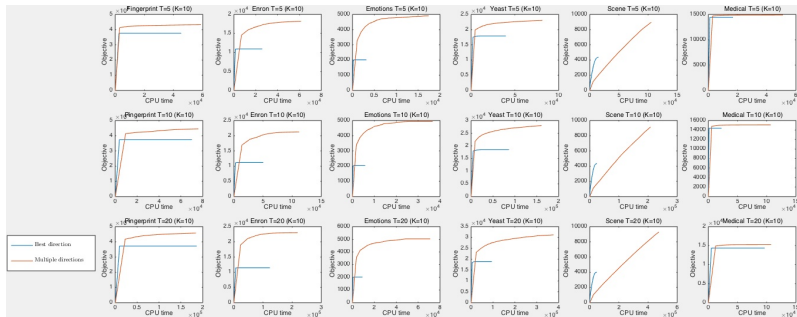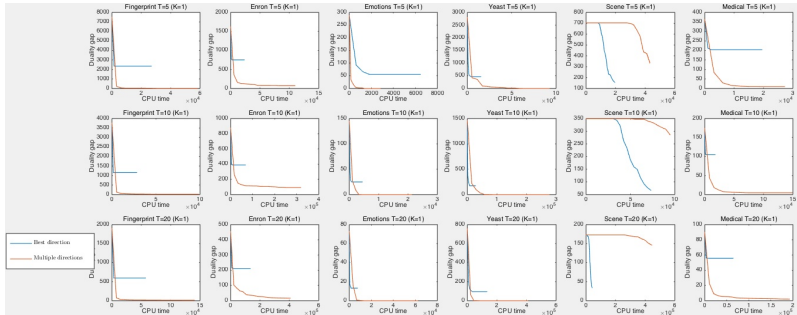3. Primal objective value is estimate by $C^{\mathsf{T}}\nabla g(\boldsymbol{\alpha}^k)$.

Experimental results - Objective value

- Compare update with the best direction v.s. update with multiple directions

- Number of spanning trees $|T| = \{5, 10, 20\}$.

- Each spanning tree outputs top $\kappa = 1$ best direction.

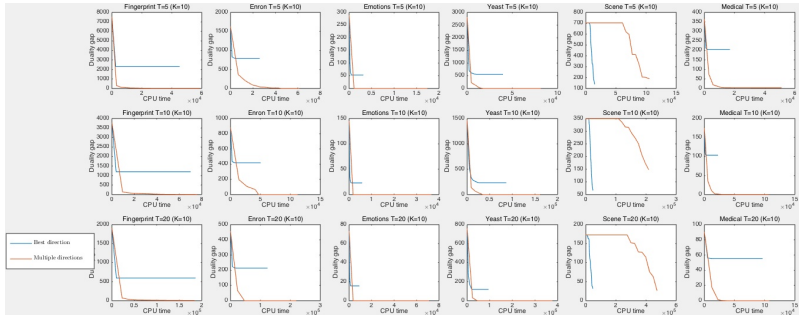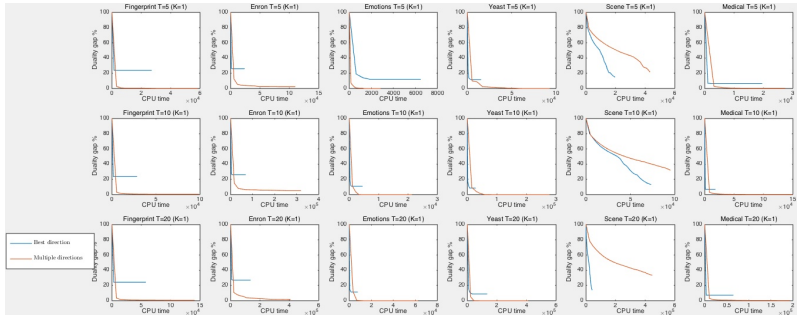# Experimental results - Objective value

- Compare update with the best direction v.s. update with multiple directions
- Number of spanning trees $|T| = \{5, 10, 20\}$.
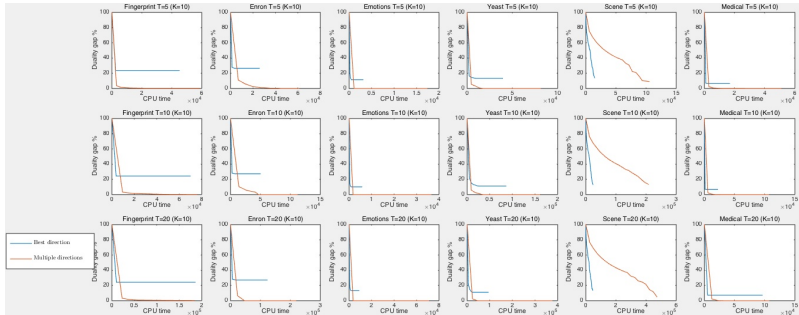- Each spanning tree outputs top $\kappa = 10$ best directions.

Experimental results - Duality gap

- Compare update with the best direction v.s. update with multiple directions
- Number of spanning trees $|T| = \{5, 10, 20\}$.
- Each spanning tree outputs top $\kappa = 1$ best direction.

Experimental results - Duality gap

- Compare update with the best direction v.s. update with multiple directions
- Number of spanning trees $|T| = \{5, 10, 20\}$.
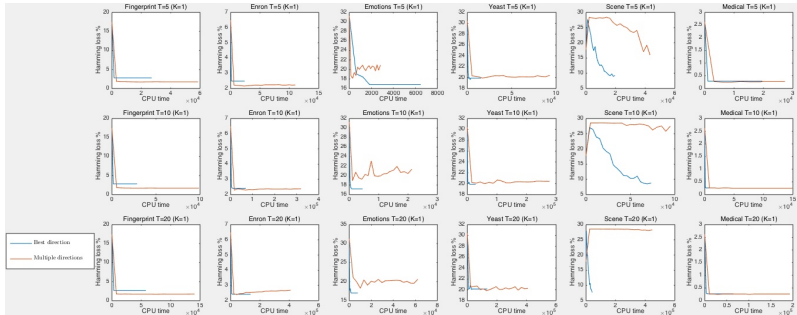- Each spanning tree outputs top $\kappa = 10$ best directions.

Experimental results - Relative duality gap

- ▶ Compare update with the best direction v.s. update with multiple directions
- ▶ Number of spanning trees $|T| = \{5, 10, 20\}$.
- ▶ Each spanning tree outputs top $\kappa = 1$ best direction.

Aalto University
School of Science
and Technology

Experimental results - Relative duality gap

▶ Compare update with the best direction v.s. update with multiple directions

▶ Number of spanning trees $|T| = \{5, 10, 20\}$.

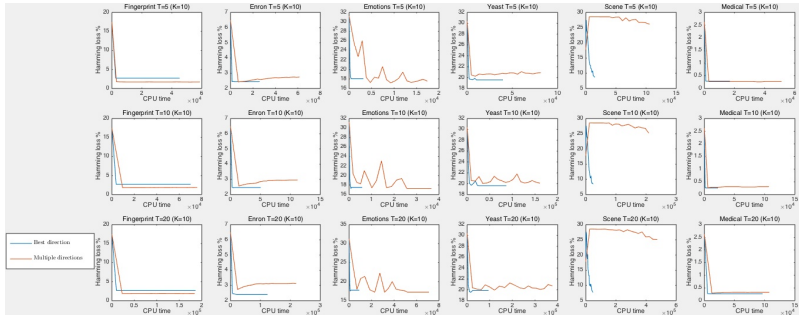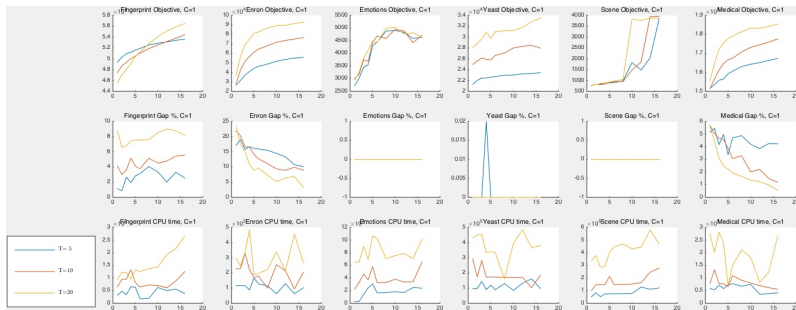▶ Each spanning tree outputs top $\kappa = 10$ best directions.

Experimental results - Hamming loss training

- ▶ Compare update with the best direction v.s. update with multiple directions
- ▶ Number of spanning trees $|T| = \{5, 10, 20\}$.
- ▶ Each spanning tree outputs top $\kappa = 1$ best direction.

Experimental results - Hamming loss training

- Compare update with the best direction v.s. update with multiple directions
- Number of spanning trees $|T| = \{5, 10, 20\}$.
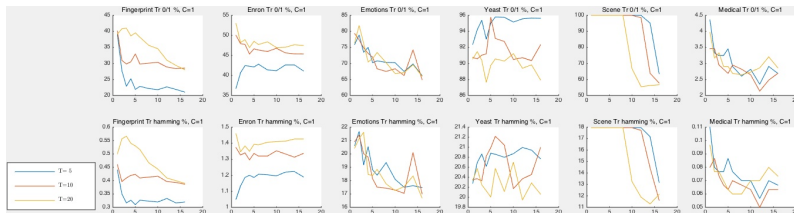- Each spanning tree outputs top $\kappa = 10$ best directions.

Experimental results

- Number of spanning trees $|T| = \{5, 10, 20\}$.
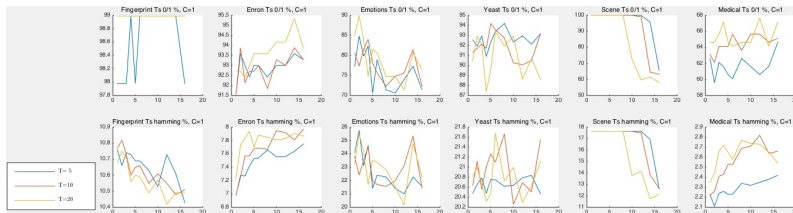- Each spanning tree outputs $\kappa = \{1, \cdots, 16\}$ best directions.

Experimental results

- Number of spanning trees $|T| = \{5, 10, 20\}$.
- Each spanning tree outputs $\kappa = \{1, \cdots, 16\}$ best directions.

Experimental results

▶ Number of spanning trees $|T| = \{5, 10, 20\}$.

▶ Each spanning tree outputs $\kappa = \{1, \cdots, 16\}$ best directions.

Conclusions

▶