PURDUE UNIVERSITY

Online Spike-and-slab Inference with Stochastic Expectation Propagation

Shandian Zhe[1], Kuang-chih Lee[2], Kai Zhang[3], and Jennifer Neville[1]
[1]Purdue University [2]Yahoo! Research [3]NEC Laboratories America

Motivation

- Sparse learning is important to real applications with high dimensional data.
  - Too many features → complicated model → huge training data → expensive computational cost
  - Small models are critical to real-time applications, such as online bidding for Ads. displaying.
- Spike-and-slab prior is the golden standard for Bayesian sparse learning; compared with popular L_1 regularization approaches, it has an appealing *selective shrinkage* effect. Suppose for each feature j, we have a weight w_j and the spike-and-slab prior over w_j is

$$p(s_j) = \mathrm{Bern}(s_j|\rho_0) = \rho_0^{s_j}(1-\rho_0)^{1-s_j}, \quad p(w_j|s_j) = s_j\mathcal{N}(w_j|0,\tau_0) + (1-s_j)\delta(w_j)$$

  where $\delta(\cdot)$ is a Dirac-delta function.
- Spike-and-slab prior is less popular, mainly due to the computational hurdle for posterior inference, especially for large data—massive samples, very high dimensions.

OLSS: Online Spike-and-slab Inference

- We focus on linear classification model:

$$p(\mathcal{D},\mathbf{w},\mathbf{s}|\rho_0,\tau_0) = \prod_{j=1}^{d}\mathrm{Bern}(s_j|\rho_0)\big(s_j\mathcal{N}(w_j|0,\tau_0)+(1-s_j)\delta(w_j)\big)\prod_{n=1}^{N}\Phi(y_n\mathbf{w}_{I_n}^\top\hat{\mathbf{x}}_n)$$

  where $\mathcal{D} = \{(\mathbf{x}_1,y_1),\ldots,(\mathbf{x}_N,y_n)\}$ are data, $\mathbf{w}$ are classification weights, $\mathbf{s}$ are selection indicators, and $\Phi(\cdot)$ is the CDF of standard Gaussian distribution.
- We use the stochastic expectation propagation framework.
  - Expectation propagation (EP). The general form of a joint distribution is

$$p(\boldsymbol{\theta},\mathcal{D}) = p_0(\boldsymbol{\theta})\prod_n p(\mathbf{z}_n|\boldsymbol{\theta}).$$

    EP approximates $p(\boldsymbol{\theta},\mathcal{D})$ with

$$q(\boldsymbol{\theta}) \propto f_0(\boldsymbol{\theta})\prod_n f_n(\boldsymbol{\theta}).$$

    EP maintains and iteratively refines each approximate terms f_i with four steps: (1) calculating the calibrating distribution, $q_{-i}(\boldsymbol{\theta}) \propto q(\boldsymbol{\theta})/f_i(\boldsymbol{\theta})$; (2) constructing a tilted distribution $t_i(\boldsymbol{\theta}) \propto q_{-i}(\boldsymbol{\theta})p(\mathbf{z}_i|\boldsymbol{\theta})$; (3) projecting t_i back into the exponential family, $q^*(\boldsymbol{\theta}) \propto \mathrm{proj}(t_i(\boldsymbol{\theta}))$, via moment matching; (4) updating the f_i: $f_i^{\mathrm{new}}(\boldsymbol{\theta}) \propto q^*(\boldsymbol{\theta})/q_{-i}(\boldsymbol{\theta})$.
  - Stochastic expectation propagation (SEP): using **one average likelihood** to summarize the data.

$$q(\boldsymbol{\theta}) \propto f_0(\boldsymbol{\theta})f_a(\boldsymbol{\theta})^N$$

    SEP sequentially process data samples and update the average likelihood f_a in an online fashion:

$$f_a(\boldsymbol{\theta})^{\mathrm{new}} = \big(f_n(\boldsymbol{\theta})f_a(\boldsymbol{\theta})^{N-1}\big)^{\frac{1}{N}}.$$

    The corresponding updates in terms of the natural parameters are

$$\boldsymbol{\lambda}_a^{\mathrm{new}} = \frac{1}{N}\boldsymbol{\lambda}_n + (1-\frac{1}{N})\boldsymbol{\lambda}_a$$

- Our approximation for spike-and-slab models:
  - We approximate the prior term, $s_j\mathcal{N}(w_j|0,\tau_0)+(1-s_j)\delta(w_j)$, with $\mathrm{Bern}(s_j|\alpha_j)\mathcal{N}(w_j|\mu_{1j},v_{1j})$.
  - We use **two average-likelihood terms**, $f_a^+(\mathbf{w}_I)$ and $f_a^-(\mathbf{w}_I)$, defined by $f_a^+(\mathbf{w}_I) = \prod_{j\in I}\mathcal{N}(w_j|\mu_{2j}^+,v_{2j}^+)$ and $f_a^-(\mathbf{w}_I) = \prod_{j\in I}\mathcal{N}(w_j|\mu_{2j}^-,v_{2j}^-)$, for the **positive** and **negative** samples, respectively.
  - Fully factorization form:

$$q(\mathbf{w},\mathbf{s}) \propto \prod_{j=1}^{d}\mathrm{Bern}(s_j|\rho_0)\mathrm{Bern}(s_j|\rho_j)\mathcal{N}(w_j|\mu_{1j},v_{1j})\mathcal{N}(w_j|\mu_{2j}^+,v_{2j}^+)^{n_j^+}\mathcal{N}(w_j|\mu_{2j}^-,v_{2j}^-)^{n_j^-}$$

    where n_j^+ and n_j^- are the appearance counts of feature j in positive and negative samples.
- The advantages:
  - Multiple average likelihoods can summarize the data distributions more accurately.
  - Easy to deal with categorical features with high cardinality.
  - Can adjust sample weights, e.g., for positive and negative samples, by setting n_j^+ and n_j^-.

Algorithm 1 OLSS($\mathcal{D},\rho_0,\tau_0,M,T,\{n_j^+,n_j^-\}_j$)

Random shuffle samples in $\mathcal{D}$.
Initialize for each feature j: $\rho_j = 0.5, \mu_{1j} = \mu_{2j}^+ = \mu_{2j}^- = 0, v_{1j} = v_{2j}^+ = v_{2j}^- = 10^6$.
repeat
  Collect a mini-batch of samples B_i with size M, where B_i^+ are B_i^- denote the positive and negative samples, and b_{ij}^+ and b_{ij}^- denote the appearance counts of feature j in B_i^+ and B_i^-.
  Calculate the approximate likelihood for each sample in B_i to obtain $\{\mathcal{N}(w_j|\mu_{jt},v_{jt})\}_{j,t\in B_i}$
  Update the Gaussian terms for the average-likelihoods:
  $v_{2j}^{+\,-1} \leftarrow \frac{b_{ji}^+}{n_j^+}\sum_{t\in B_i^+}v_{jt}^{-1} + \frac{n_j^+-b_{ji}^+}{n_j^+}v_{2j}^{+\,-1}$, $\frac{\mu_{2j}^+}{v_{2j}^+} \leftarrow \frac{b_{ji}^+}{n_j^+}\sum_{t\in B_i^+}\frac{\mu_{jt}}{v_{jt}} + \frac{n_j^+-b_{ji}^+}{n_j^+}\frac{\mu_{2j}^+}{v_{2j}^+}$,
  $v_{2j}^{-\,-1} \leftarrow \frac{b_{ji}^-}{n_j^-}\sum_{t\in B_i^-}v_{jt}^{-1} + \frac{n_j^--b_{ji}^-}{n_j^-}v_{2j}^{-\,-1}$, $\frac{\mu_{2j}^-}{v_{2j}^-} \leftarrow \frac{b_{ji}^-}{n_j^-}\sum_{t\in B_i^-}\frac{\mu_{jt}}{v_{jt}} + \frac{n_j^--b_{ji}^-}{n_j^-}\frac{\mu_{2j}^-}{v_{2j}^-}$.
  If T mini-batches have been processed, update $\{\rho_j,\mu_{1j},v_{1j}\}_j$ for the approximate prior terms.
until all samples in $\mathcal{D}$ is passed.
return $q(\mathbf{w},\mathbf{s}) = \prod_j\mathcal{N}(w_j|\mu_j,v_j)\mathrm{Bern}(s_j|\alpha_j)$, where $v_j = \big(v_{1j}^{-1} + n_j^+v_{2j}^{+\,-1} + n_j^-v_{2j}^{-\,-1}\big)^{-1}$, $\mu_j = v_j\big(\frac{\mu_{1j}}{v_{1j}} + n_j^+\frac{\mu_{2j}^+}{v_{2j}^+} + n_j^-\frac{\mu_{2j}^-}{v_{2j}^-}\big)$, $\alpha_j = \sigma\big(\sigma^{-1}(\rho_0)+\sigma^{-1}(\rho_j)\big)$ ($\sigma(\cdot)$ is the logitic function).

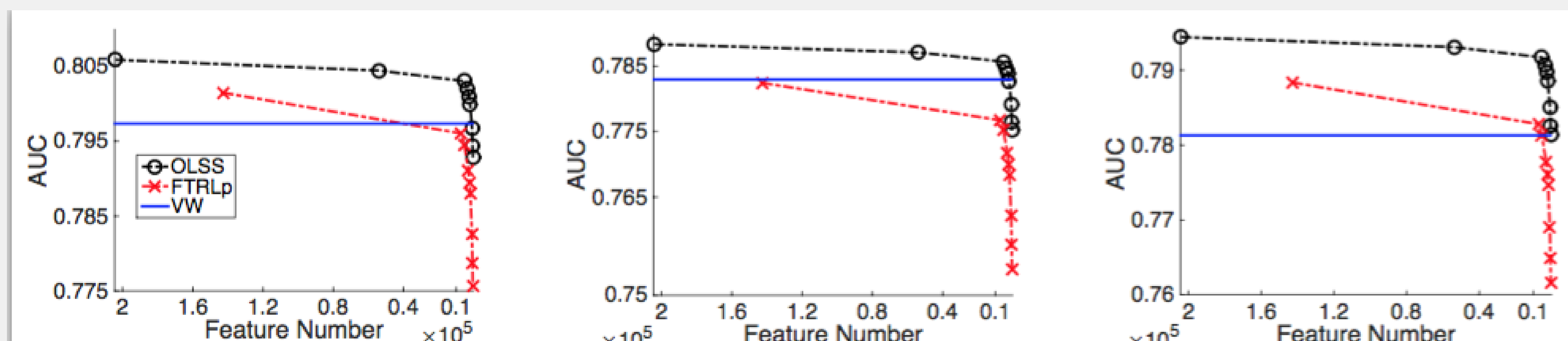
Experiments

- Real CTR prediction task on Yahoo! Display ads platform.
- Training data: click logs between 07/15/2016 and 07/21/2016
- Testing data: click logs in 07/22/2016, 07/23/2016 and 07/24/2016.
- Feature number: 204,327.
- Training and testing sizes: 1.8M, 133.7M, 116.0M, and 110.2M.
- Competing methods: online logistic regression in Vowpal Wabbit (VW), FTRL-proximal (FTRLp).
- Sparsity achievement.

Table: The number of selected features *v.s.* the setting of ρ_0.

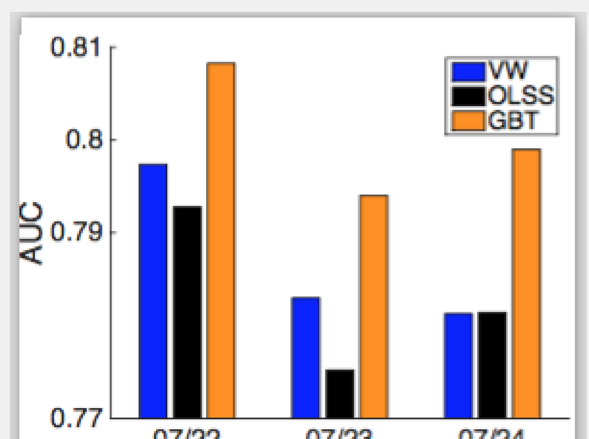
ρ_0	0.8	0.5	0.4	0.3	0.1	10^{-3}	10^{-5}	10^{-7}
feature number	204,080	53,827	5,591	3,810	2,174	1,004	663	504
ratio (%)	99.9%	26.3%	2.7%	1.9%	1.1%	0.5%	0.3%	0.2%

- Predictive performance with different sparsity levels.



(a) 07/22 (b) 07/23 (c) 07/24

- Usage of the selected features. We used 504 features selected by OLSS and trained a nonlinear classification model, Gradient Boosting Tree (GBT).



(d)

GBT outperformed OLSS on 504 features and VW on the entire 204,327 features, among all the three test datasets.

Future Work

- Examination on millions of features, which are more often used in industry.
- Online A/B test on various sample weights settings.
- Distributed, asynchronous stochastic spike-and-slab inference.

szhe@purdue.edu, kclee@yahoo-inc.com, kzhang@nec-labs.com, neville@cs.purdue.edu