Supplementary Material of Robust Bayesian Kernel Machine via Stein Variational Gradient Descent for Big Data

January 26, 2018

1 Loss functions for multiclass problem

Given a data example $(\mathbf{x}, y)$ and M hyperplanes $\mathbf{w}_1, \ldots, \mathbf{w}_M$, each corresponds to a class label. For notation convenience, we stack these vectors into a matrix $\mathbf{W} = [\mathbf{w}_1, \ldots, \mathbf{w}_M]^\mathsf{T}$. The discriminative score for the data example $\mathbf{x}$ given by each class is denoted as $g_m(\mathbf{x})$. In the linear classification, $g_m(\mathbf{x})$ is defined as $\mathbf{w}^\mathsf{T}\mathbf{x}$. To deal with nonlinear nature of data, the input data $\mathbf{x}$ is mapped onto a feature space via a transformation $\boldsymbol{\phi}(\mathbf{x})$, thus the score $g_m(\mathbf{x})$ is of the form $\mathbf{w}^\mathsf{T}\boldsymbol{\phi}(\mathbf{x})$. There are two popular loss functions for multiclass problem: the multiclass Hinge loss and the cross-entropy loss with softmax function.

1.1 Multiclass Hinge loss

Inspired from the binary Hinge loss, the multiclass Hinge loss is defined over the gap between the scores given by the true class and the runner-up as follows:

$$l(\mathbf{W}; \mathbf{x}, y) = \max\left\{0, 1 + \max_{j \neq y} g_j(\mathbf{x}) - g_y(\mathbf{x})\right\}$$

This is a convex and non-smooth loss function, hence we have a set of sub-gradients at each point and sub-gradients at different points are typically not smoothly varied.

1.2 Cross-entropy loss with softmax function

Let us consider the one-hot vector $\mathbf{y}$ of M elements where the y-th element is one. The one-hot vector $\mathbf{y}$ describes the true distribution p of labels for the data example $\mathbf{x}$. The distribution q of labels for $\mathbf{x}$ given by the current model $\mathbf{W}$ is defined over the softmax function as follows:

$$q(Y = j \mid \mathbf{W}, \mathbf{x}) = \frac{\exp(g_j(\mathbf{x}))}{\sum_{m=1}^{M} \exp(g_m(\mathbf{x}))}$$

The cross-entropy between two discrete distribution p and q is defined as follows:

$$H(p, q) = -\sum_{Y} p(Y) \log q(Y \mid \mathbf{W}, \mathbf{x})$$

It measures the distance between p and q, thus it can be used as a loss function. If p and q is exactly the same, the loss function is equal to zero. Because $p(Y)$ equals 1 only when $Y = y$ and otherwise is zero, we can derive as follows:

$$l(\mathbf{W}; \mathbf{x}, y) = H(p, q) = -\log q(Y = y \mid \mathbf{W}, \mathbf{x}) = -\log \frac{\exp(g_y(\mathbf{x}))}{\sum_{m=1}^{M} \exp(g_m(\mathbf{x}))}$$

2 Gradients of the log of the posterior distribution

Given the training set $\mathcal{D} = \{(\mathbf{x}_n, y_n)\}_{n=1}^{N}$ where $\mathbf{x} \in \mathcal{X} \subset \mathbb{R}^d$ and $y \in \mathcal{Y}$. In this section, we consider the nonlinear case where the data are mapped onto a random feature space via a transformation $\boldsymbol{\phi}$ defined as:

$$\boldsymbol{\phi}(\mathbf{x}) = \frac{1}{\sqrt{D}} \left[\cos\left(\boldsymbol{\omega}_i^\mathsf{T}\mathbf{x}\right), \sin\left(\boldsymbol{\omega}_i^\mathsf{T}\mathbf{x}\right)\right]_{i=1}^{D}$$

in which the random frequencies $\boldsymbol{\omega}_i, i = 1, \ldots, D$ is computed as follows:

$$\boldsymbol{\omega}_i = \boldsymbol{\gamma} \odot \mathbf{e}_i$$

where $\mathbf{e}_i$ is drawn from $\mathcal{N}(\mathbf{0}, \mathbf{I}_d)$. We note that it is straightforward to derive the formula for the linear case from the nonlinear case where we simply set $\boldsymbol{\phi}(\mathbf{x}) = \mathbf{x}$.
The posterior distribution of the model $\mathbf{W}$ and kernel parameter $\boldsymbol{\gamma}$ defined as follows:

$$p(\mathbf{W}, \boldsymbol{\gamma} \mid \mathcal{D}) \propto p(\mathcal{D} \mid \mathbf{W}, \boldsymbol{\gamma}) p(\mathbf{W}) p(\boldsymbol{\gamma})$$

where we have defined

$$p(\mathcal{D} \mid \mathbf{W}, \boldsymbol{\gamma}) \propto \prod_{(\mathbf{x},y)\in\mathcal{D}} \exp\left(-p(\mathbf{x}, y)\, l(\mathbf{W}; \mathbf{x}, y)\right)$$

and $p(\mathbf{W})$, $p(\boldsymbol{\gamma})$ is prior distributions. The log of the posterior distribution is of the form:

$$\log p(\mathbf{W}, \boldsymbol{\gamma} \mid \mathcal{D}) = \log p(\mathbf{W}) + \log p(\boldsymbol{\gamma}) - \mathbb{E}_{P_{\mathcal{X}\times\mathcal{Y}}}\left[l(\mathbf{W}; \mathbf{x}, y)\right] + const$$

We can approximate $\mathbb{E}_{P_{\mathcal{X}\times\mathcal{Y}}}\left[l(\mathbf{W}; \mathbf{x}, y)\right]$ by sampling a mini-batch $\mathcal{B}$ from the training set $\mathcal{D}$ as follows:

$$\mathbb{E}_{P_{\mathcal{X}\times\mathcal{Y}}}\left[l(\mathbf{W}; \mathbf{x}, y)\right] \approx \frac{1}{|\mathcal{B}|} \sum_{(\mathbf{x},y)\in\mathcal{B}} l(\mathbf{W}; \mathbf{x}, y)$$

In summary, the log of posterior distribution of model and kernel parameters is of the form:

$$\log p(\mathbf{W}, \boldsymbol{\gamma} \mid \mathcal{D}) \approx \log p(\mathbf{W}, \boldsymbol{\gamma} \mid \mathcal{B}) = \log p(\mathbf{W}) + \log p(\boldsymbol{\gamma}) - \frac{1}{|\mathcal{B}|} \sum_{(\mathbf{x},y)\in\mathcal{B}} l(\mathbf{W}; \mathbf{x}, y) + const$$

In this section, our aim is to compute the gradients of $\log p(\mathbf{W}, \boldsymbol{\gamma} \mid \mathcal{D})$ w.r.t to the model $\mathbf{W}$ and kernel parameter $\boldsymbol{\gamma}$ respectively. In our current framework, the prior of $\mathbf{W}$ is a Multivariate Gaussian distribution $\mathcal{N}\left(\mathbf{0}, \lambda^{-1}\mathbf{I}\right)$ and the prior of $\boldsymbol{\gamma}$ is Truncate Multivariate Gaussian distribution $\mathcal{N}(\mathbf{0}, \mathbf{I}_d, \mathbf{0}, +\infty)$. Obviously, the gradients is of the form:

$$\nabla_{\mathbf{W}} \log p(\mathbf{W}, \boldsymbol{\gamma} \mid \mathcal{B}) = -\lambda \|\mathbf{W}\|_{2,2}^2 - \frac{1}{|\mathcal{B}|} \sum_{(\mathbf{x},y)\in\mathcal{B}} \nabla_{\mathbf{W}} l(\mathbf{W}; \mathbf{x}, y)$$

$$\nabla_{\boldsymbol{\gamma}} \log p(\mathbf{W}, \boldsymbol{\gamma} \mid \mathcal{B}) = -\frac{1}{|\mathcal{B}|} \sum_{(\mathbf{x},y)\in\mathcal{B}} \nabla_{\boldsymbol{\gamma}} l(\mathbf{W}; \mathbf{x}, y)$$

Next, we present the gradients of the loss function w.r.t the model $\mathbf{W}$ and kernel parameter $\boldsymbol{\gamma}$ respectively.

• Compute $\nabla_{\mathbf{W}} l(\mathbf{W}; \mathbf{x}, y)$

For the Hinge loss function. We have $\nabla_{\mathbf{W}} l(\mathbf{W}; \mathbf{x}, y) = 0$ if $l(\mathbf{W}; \mathbf{x}, y) = 0$. Otherwise, we have

$$\nabla_{\mathbf{w}_m} l(\mathbf{W}; \mathbf{x}, y) = \begin{cases} -\boldsymbol{\phi}(\mathbf{x}) & \text{if } m = y \\ \boldsymbol{\phi}(\mathbf{x}) & \text{if } m = m^* \\ \mathbf{0} & \text{otherwise} \end{cases}$$

where $m^* = \underset{m\in\mathcal{Y}\setminus y}{\operatorname{argmax}}\, g_m(\mathbf{x}) = \underset{m\in\mathcal{Y}\setminus y}{\operatorname{argmax}}\, \mathbf{w}_m^\mathsf{T}\mathbf{x}$.
For cross-entropy with softmax function. We have

$$l(\mathbf{W}; \mathbf{x}, y) = -\log \frac{\exp(g_y(\mathbf{x}))}{\sum_{m=1}^M \exp(g_m(\mathbf{x}))} = -\log \frac{\exp\left(\mathbf{w}_y^\mathsf{T}\boldsymbol{\phi}(\mathbf{x})\right)}{\sum_{m=1}^M \exp\left(\mathbf{w}_m^\mathsf{T}\boldsymbol{\phi}(\mathbf{x})\right)} = -\mathbf{w}_y^\mathsf{T}\boldsymbol{\phi}(\mathbf{x}) + \log\left(\sum_{m=1}^M \exp\left(\mathbf{w}_m^\mathsf{T}\boldsymbol{\phi}(\mathbf{x})\right)\right)$$

The gradient of the loss function w.r.t $\mathbf{W}$ is formulated as follows:

$$\nabla_{\mathbf{w}_y} l(\mathbf{W}; \mathbf{x}, y) = -\boldsymbol{\phi}(\mathbf{x}) + \frac{\boldsymbol{\phi}(\mathbf{x}) \exp\left(\mathbf{w}_y^\mathsf{T}\boldsymbol{\phi}(\mathbf{x})\right)}{\sum_{m=1}^M \exp\left(\mathbf{w}_m^\mathsf{T}\boldsymbol{\phi}(\mathbf{x})\right)} = \boldsymbol{\phi}(\mathbf{x})\left(s\left(\mathbf{W}^\mathsf{T}\boldsymbol{\phi}(\mathbf{x})\right)_y - 1\right)$$

$$\nabla_{\mathbf{w}_{i\neq y}} l(\mathbf{W}; \mathbf{x}, y) = \frac{\boldsymbol{\phi}(\mathbf{x}) \exp\left(\mathbf{w}_i^\mathsf{T}\boldsymbol{\phi}(\mathbf{x})\right)}{\sum_{m=1}^M \exp\left(\mathbf{w}_m^\mathsf{T}\boldsymbol{\phi}(\mathbf{x})\right)} = \boldsymbol{\phi}(\mathbf{x})\, s\left(\mathbf{W}^\mathsf{T}\boldsymbol{\phi}(\mathbf{x})\right)_i$$

where $s(\mathbf{g})_i = \frac{\exp(g_i)}{\sum_{m=1}^M \exp(g_m)}$ and $\mathbf{g} = [g_1, \ldots, g_M]$.

- **Compute $\nabla_{\boldsymbol{\gamma}} l\left(\mathbf{W}; \mathbf{x}, y\right)$**

The gradient of the loss function w.r.t the kernel parameter $\boldsymbol{\gamma}$ is formulated as follows:

$$\nabla_{\boldsymbol{\gamma}} l\left(\mathbf{W}; \mathbf{x}, y\right) = \nabla_{\boldsymbol{\gamma}} \boldsymbol{\omega} \times \nabla_{\boldsymbol{\omega}} \boldsymbol{\phi}\left(\mathbf{x}\right) \times \nabla_{\boldsymbol{\phi}} l\left(\mathbf{W}; \mathbf{x}, y\right)$$

where we have

$$\begin{aligned}
\nabla_{\boldsymbol{\gamma}} \boldsymbol{\omega} &= \left[\nabla_{\boldsymbol{\gamma}} \boldsymbol{\omega}_1, \ldots \nabla_{\boldsymbol{\gamma}} \boldsymbol{\omega}_D\right] \\
\nabla_{\boldsymbol{\gamma}} \boldsymbol{\omega}_i &= \mathbf{e}_i \\
\nabla_{\boldsymbol{\omega}} \boldsymbol{\phi}\left(\mathbf{x}\right) &= \left[\nabla_{\boldsymbol{\omega}_1} \boldsymbol{\phi}\left(\mathbf{x}\right), \ldots, \nabla_{\boldsymbol{\omega}_D} \boldsymbol{\phi}\left(\mathbf{x}\right)\right]^{\mathsf{T}} \\
\nabla_{\boldsymbol{\omega}_i} \boldsymbol{\phi}\left(\mathbf{x}\right) &= \left[\mathbf{0}, -\mathbf{x} \sin\left(\boldsymbol{\omega}_i^{\mathsf{T}} \mathbf{x}\right), \mathbf{x} \cos\left(\boldsymbol{\omega}_i^{\mathsf{T}} \mathbf{x}\right), \mathbf{0}\right]
\end{aligned}$$

Thus, we have

$$\nabla_{\boldsymbol{\gamma}} l\left(\mathbf{W}; \mathbf{x}, y\right) = \left[-\mathbf{e}_i \odot \mathbf{x} \sin\left(\boldsymbol{\omega}_i^{\mathsf{T}} \mathbf{x}\right), \mathbf{e}_i \odot \mathbf{x} \cos\left(\boldsymbol{\omega}_i^{\mathsf{T}} \mathbf{x}\right)\right]_{i=1}^{D} \times \nabla_{\boldsymbol{\phi}} l\left(\mathbf{W}; \mathbf{x}, y\right)$$

For the Hinge loss function. We have

$$\begin{aligned}
\nabla_{\boldsymbol{\phi}} l\left(\mathbf{W}; \mathbf{x}, y\right) &= \nabla_{\boldsymbol{\phi}} \left(\max\left\{0, 1 + \max_{m \neq y} \mathbf{w}_m^{\mathsf{T}} \boldsymbol{\phi}\left(\mathbf{x}\right) - \mathbf{w}_y^{\mathsf{T}} \boldsymbol{\phi}\left(\mathbf{x}\right)\right\}\right) \\
&= \mathbb{I}_{l(\mathbf{W}; \mathbf{x}, y) > 0} \left(\mathbf{w}_{m^*} - \mathbf{w}_y\right)
\end{aligned}$$

For cross-entropy with softmax function. We have

$$\begin{aligned}
\nabla_{\boldsymbol{\phi}} l\left(\mathbf{W}; \mathbf{x}, y\right) &= \nabla_{\boldsymbol{\phi}} \left(-\mathbf{w}_y^{\mathsf{T}} \boldsymbol{\phi}\left(\mathbf{x}\right) + \log\left(\sum_{m=1}^{M} \exp\left(\mathbf{w}_m^{\mathsf{T}} \boldsymbol{\phi}\left(\mathbf{x}\right)\right)\right)\right) \\
&= -\mathbf{w}_y + \frac{\sum_{m=1}^{M} \mathbf{w}_m \exp\left(\mathbf{w}_m^{\mathsf{T}} \boldsymbol{\phi}\left(\mathbf{x}\right)\right)}{\sum_{m=1}^{M} \exp\left(\mathbf{w}_m^{\mathsf{T}} \boldsymbol{\phi}\left(\mathbf{x}\right)\right)} \\
&= -\mathbf{w}_y + \sum_{m=1}^{M} \mathbf{w}_m s\left(\mathbf{W}^{\mathsf{T}} \boldsymbol{\phi}\left(\mathbf{x}\right)\right)_m
\end{aligned}$$