Scalable Gaussian Process Classification via Expectation Propagation

Daniel Hernández-Lobato and José Miguel Hernández-Lobato

Universidad Autónoma de Madrid, Harvard University

daniel.hernandez@uam.es, jmh@seas.harvard.edu

Motivation and Main Result of the Paper

Expectation Propagation **does not scale** to large datasets. Computing the gradient of the estimate of the marginal likelihood is expensive and the algorithm **does not allow to use minibatches of data** for training. We describe here a scalable version of Expectation Propagation that does not have these limitations and that **can be applied in datasets with millions** of instances.

Expectation Propagation for Approximate Inference

$$p(\mathbf{z}) \propto p_0(\mathbf{z}|\boldsymbol{\xi})\ \phi_1(\mathbf{z}|\boldsymbol{\xi})\ \phi_2(\mathbf{z}|\boldsymbol{\xi})\ \phi_3(\mathbf{z}|\boldsymbol{\xi}) \approx q(\mathbf{z}) \propto p_0(\mathbf{z}|\boldsymbol{\xi})\ \tilde{\phi}_1(\mathbf{z})\ \tilde{\phi}_2(\mathbf{z})\ \tilde{\phi}_3(\mathbf{z})$$

The update of each factor minimizes the **Kullback-Leibler divergence** between:

$$\hat{p}(\mathbf{z}) \propto \overbrace{p_0(\mathbf{z}|\boldsymbol{\xi})\ \tilde{\phi}_1(\mathbf{z})\ \tilde{\phi}_2(\mathbf{z})}^{q^{\setminus 3}(\mathbf{z})}\ \phi_3(\mathbf{z}|\boldsymbol{\xi}) \approx q(\mathbf{z}) \propto \overbrace{p_0(\mathbf{z}|\boldsymbol{\xi})\ \tilde{\phi}_1(\mathbf{z})\ \tilde{\phi}_2(\mathbf{z})}^{q^{\setminus 3}(\mathbf{z})}\ \tilde{\phi}_3(\mathbf{z})$$

Matches Moments Between the Two Distributions

Marginal likelihood estimate: $\log Z_q = g(\boldsymbol{\theta}_{\text{post}}) - g(\boldsymbol{\theta}_{\text{prior}}) + \sum_{i=1}^{N} \log C_i$
where $\log C_i = \log Z_i + g(\boldsymbol{\theta}^{\setminus i}) - g(\boldsymbol{\theta}_{\text{post}}) = \log \int \phi_i(\mathbf{z}|\boldsymbol{\xi}) q^{\setminus i}(\mathbf{z}) d\mathbf{z} + g(\boldsymbol{\theta}^{\setminus i}) - g(\boldsymbol{\theta}_{\text{post}})$.

$$\frac{\partial \log Z_q}{\partial \xi_j} = \boldsymbol{\eta}_{\text{post}}^{\text{T}} \frac{\partial \boldsymbol{\theta}_{\text{prior}}}{\partial \xi_j} - \boldsymbol{\eta}_{\text{prior}}^{\text{T}} \frac{\partial \boldsymbol{\theta}_{\text{prior}}}{\partial \xi_j} + \sum_{i=1}^{N} \frac{\partial \log Z_i}{\partial \xi_j}$$

Hyper-Parameter Learning

Repeat:

1. Run EP until the approximate factors do not change any more.
2. Compute the estimate of the marginal likelihood $\log Z_q$.
3. Compute the gradient of the marginal likelihood w.r.t. $\boldsymbol{\xi}$.
4. Update $\boldsymbol{\xi}$ by taking a step in the direction of the gradient.

Expensive and inefficient: runs EP until convergence at each iteration and at the beginning the hyper-parameters are likely to be very bad.

Scalable Expectation Propagation (SEP): Allows to Run EP in Very Large Datasets

Solves the scaling problem of Expectation Propagation:

- **Batch Training**: At each iteration update simultaneously all $\tilde{\phi}_i$ and the hyper-parameters using the gradient employed in standard Expectation Propagation at convergence.
- **Stochastic training**: Given a minibatch of data $\mathcal{M}$, update $\tilde{\phi}_i$ with $i \in \mathcal{M}$ and update the hyper-parameters with:

$$\frac{\partial \log Z_q}{\partial \xi_j} \approx \boldsymbol{\eta}_{\text{post}}^{\text{T}} \frac{\partial \boldsymbol{\theta}_{\text{prior}}}{\partial \xi_j} - \boldsymbol{\eta}_{\text{prior}}^{\text{T}} \frac{\partial \boldsymbol{\theta}_{\text{prior}}}{\partial \xi_j} + \frac{N}{|\mathcal{M}|} \sum_{i \in \mathcal{M}} \frac{\partial \log Z_i}{\partial \xi_j}$$

- **Distributed Training:** Send $1/K$ of the data to K computational nodes to compute in each node $1/K$ of the $\tilde{\phi}_i$ and the corresponding contribution to $\partial \log Z_q / \partial \xi_j$.

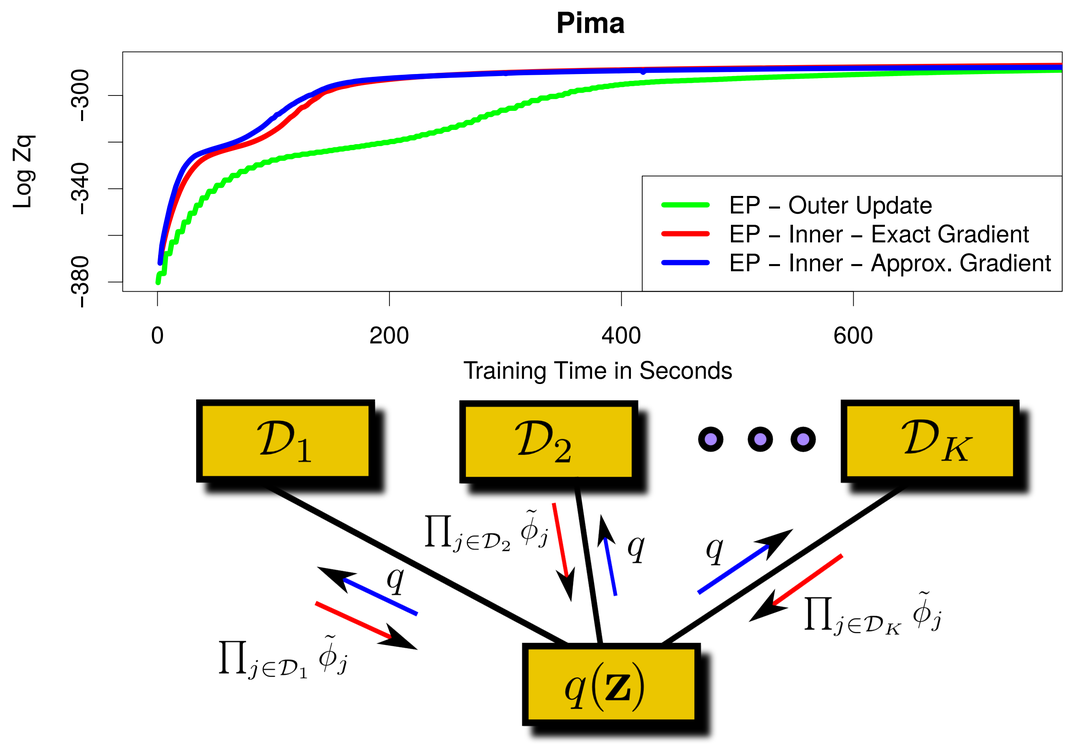
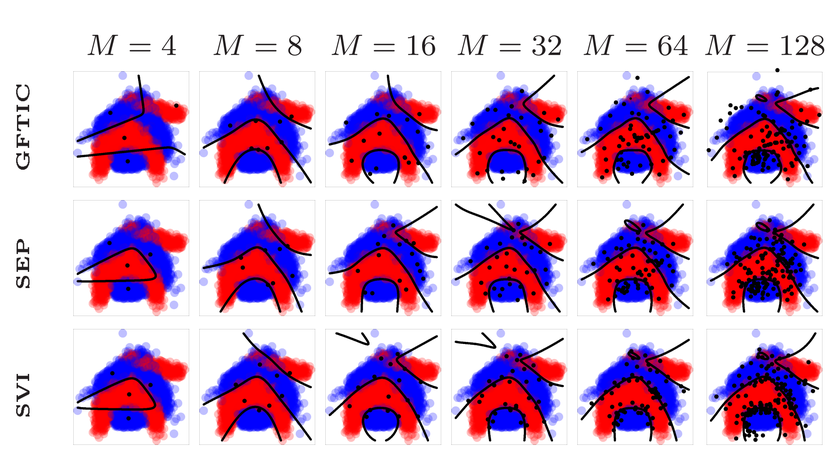
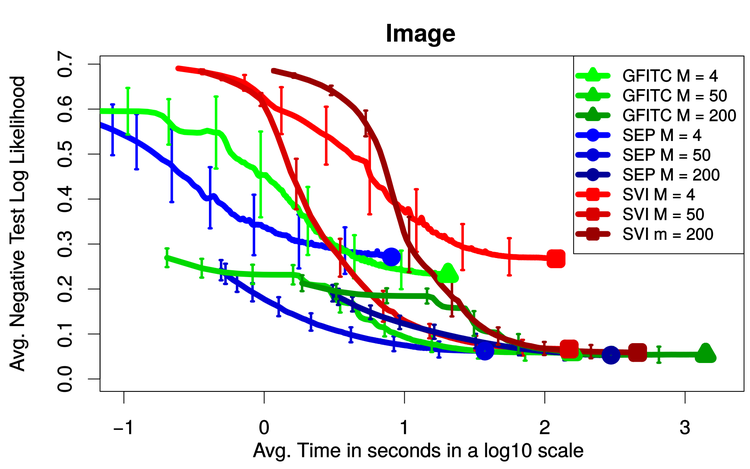


Illustration: Scalable Gaussian Process Classification

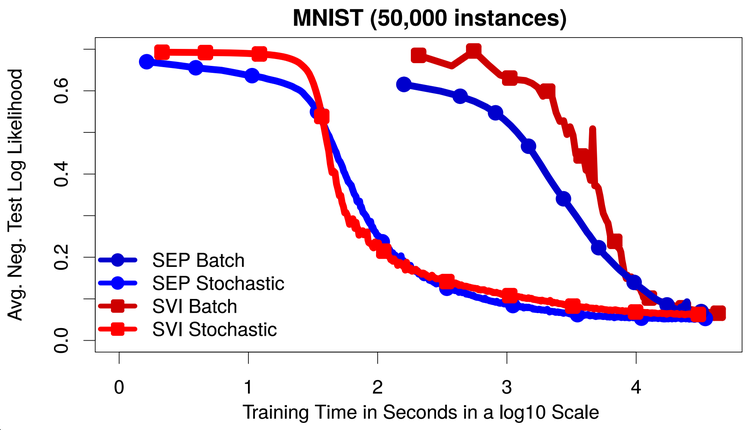
Inducing point representation: $\overline{\mathbf{X}} = (\overline{\mathbf{x}}_1, \ldots, \overline{\mathbf{x}}_M)^{\text{T}}$ and $\overline{\mathbf{f}} = (f(\overline{\mathbf{x}}_1), \ldots, f(\overline{\mathbf{x}}_M))^{\text{T}}$. We compute $q(\overline{\mathbf{f}}) \approx p(\overline{\mathbf{f}}|\mathbf{y})$.

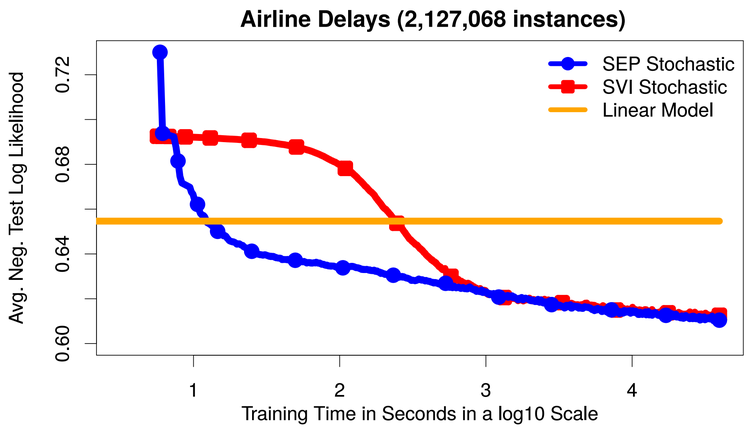




	$M = 25\%$		
Problem	**GFITC**	**SEP**	**SVI**
Australian	.68 ± .08	.67 ± .07	**.63 ± .05**
Breast	.11 ± .06	.11 ± .05	**.10 ± .05**
Crabs	**.06 ± .07**	.06 ± .06	.07 ± .07
Heart	.42 ± .12	.41 ± .12	**.40 ± .11**
Ionosphere	.29 ± .23	**.27 ± .20**	.27 ± .18
Pima	.53 ± .07	.51 ± .06	**.50 ± .05**
Sonar	.35 ± .12	**.32 ± .10**	.40 ± .19
Avg. Time	133 ± 6	37 ± 2	65 ± 3

Training in Large Datasets with Stochastic Gradients





Conclusions

- SEP updates the model hyper-parameters and the approximate factors at the same time.
- SEP can be used to perform approximate inference in large datasets very efficiently.
- SEP allows for stochastic and distributed computations. Training cost is independent of N.
- The memory costs scale linearly with N.