Conclusions

GFITC
133
.42
.53
.35
.29
.11
x
x
x
x
x
f
x
x
x

EP − Outer Update
EP − Inner − Exact Gradient
EP − Inner − Approx. Gradient

Illustration: Scalable Gaussian Process Classification

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 \( \phi_i \) and the hyper-parameters using the gradient employed in standard Expectation Propagation at convergence.

- **Stochastic training**: Given a minibatch of data \( M \), update \( \phi_i \) with \( i \in M \) and update the hyper-parameters with:

\[
\frac{\partial \log Z_q}{\partial \xi_j} \approx \left( \eta_{\text{post}} - \eta_{\text{prior}} \right) / \left| M \right| + \sum_{i \in M} \frac{\partial \log Z_i}{\partial \xi_j}
\]

- **Distributed Training**: Send \( 1/K \) of the data to \( K \) computational nodes to compute in each node \( 1/K \) of the \( \phi_i \) and the corresponding contribution to \( \partial \log Z_q/\partial \xi_j \).

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. \( \xi \).
4. Update \( \xi \) 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.

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

\[
p(z) \propto p_0(z) \prod_{i} \phi_i(z) \approx q(z) \propto p_0(z) \prod_{i} \phi_i(z)
\]

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

\[
q^{13}(z) \approx \hat{q}^{13}(z)
\]

Matches Moments Between the Two Distributions

Marginal likelihood estimate: \( \log Z_q = g(\theta_{\text{post}}) - g(\theta_{\text{prior}}) + \sum_{i=1}^{N} \log C_i \), where \( \log C_i = \log Z_i + g(\theta_{\text{prior}}) - g(\theta_{\text{post}}) \).

\[
\frac{\partial \log Z_q}{\partial \xi_j} = \eta_{\text{post}} \frac{\partial \theta_{\text{prior}}}{\partial \xi_j} - \eta_{\text{prior}} \frac{\partial \theta_{\text{prior}}}{\partial \xi_j} + \sum_{i=1}^{N} \frac{\partial \log Z_i}{\partial \xi_j}
\]

Scalable Expectation Propagation (SEP): A ates the model hyper-parameters and the approximate factors at the same time.

- SEP allows for stochastic and distributed computations. Training cost is independent of \( N \).
- The memory costs scale linearly with \( N \).

Illustration: Scalable Gaussian Process Classification

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

Training in Large Datasets with Stochastic Gradients

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