Extending Expectation Propagation on Graphical Models

Yuan (Alan) Qi
MIT Media Lab
Motivation

• Graphical models are widely used in real-world applications, such as human behavior recognition and wireless digital communications.

• Inference on graphical models: infer hidden variables
  – Previous approaches often sacrifice efficiency for accuracy or sacrifice accuracy for efficiency.
  > Need methods that better balance the trade-off between accuracy and efficiency.

• Learning graphical models: learning model parameters
  – Overfitting problem: Maximum likelihood approaches
  > Need efficient Bayesian training methods
Outline

• Background:
  – Graphical models and expectation propagation (EP)

• Inference on graphical models
  – Extending EP on Bayesian dynamic networks
    • Fixed lag smoothing: wireless signal detection
    • Different approximation techniques: Poisson tracking
  – Combining EP with junction tree algorithm on loopy graphs

• Learning conditional graphical models
  – Extending EP classification to perform feature selection
    • Gene expression classification
  – Training Bayesian conditional random fields
    • Handwritten ink analysis

• Conclusions
Outline

• Background on expectation propagation (EP)
  – 4 kinds of graphical models
  – EP in a nutshell
• Inference on graphical models
• Learning conditional graphical models
• Conclusions and future work
Graphical Models

$\text{Bayesian networks}$

$p(x,y) = \prod_i p(x_i \mid x_{pa(i)}) \prod_j p(y_j \mid x_{pa(j)})$

$p(x) = \frac{1}{Z} \prod_a \phi_a(x, y)$

$\text{Markov networks}$

$p(t \mid x, w) = \prod_i p(t_i \mid x_i, w)$

$p(t \mid w, x) = \frac{1}{Z(w)} \prod_a \phi_a(x, y, t)$

$\text{conditional classification}$

$\text{conditional random fields}$
Expectation Propagation in a Nutshell

• Approximate a probability distribution by simpler parametric terms:

\[ p(x) = \prod_a f_a(x) \quad \longrightarrow \quad q(x) = \prod_a \tilde{f}_a(x) \]

  - For Bayesian networks: \( f_a(x) = p(x_{i\in a} \mid x_{j\not\in a}) \)
  - For Markov networks: \( f_a(x) = \phi_a(x, y) \)

\[ p(w \mid t, x) = \prod_a f_a(w) \quad \rightarrow \quad q(w) = \prod_a \tilde{f}_a(w) \]

  - For conditional classification: \( f_a(w) = p(t_a \mid x_a, w) \)
  - For conditional random fields: \( f_a(w) = \phi_a(x, t_a, w) \)

Each approximation term \( \tilde{f}_a(x) \) or \( \tilde{f}_a(w) \) lives in an exponential family (such as Gaussian & Multinomial)
EP in a Nutshell (2)

• The approximate term $\tilde{f}_a(x)$ minimizes the following KL divergence by \textit{moment matching}:

$$\arg\min_{\tilde{f}_a(x)} D(f_a(x)q^a(x) \| \tilde{f}_a(x)q^a(x))$$

Where the leave-one-out approximation is

$$q^a(x) = \prod_{b \neq a} \tilde{f}_b(x) \propto \frac{q(x)}{f_a(x)}$$
EP in a Nutshell (3)

Three key steps:

• Deletion Step: approximate the “leave-one-out” predictive posterior for the $i^{th}$ point:
  \[ q^{\backslash i}(\mathbf{x}) \propto q(\mathbf{x}) / \tilde{f}_i(\mathbf{x}) = \prod_{j \neq i} \tilde{f}_j(\mathbf{x}) \]

• Minimizing the following KL divergence by moment matching (Assumed Density filtering):
  \[ \underset{\tilde{f}_i(\mathbf{x})}{\arg \min} KL(f_i(\mathbf{x})q^{\backslash i}(\mathbf{x}) \parallel \tilde{f}_i(\mathbf{x})q^{\backslash i}(\mathbf{x})) \]

• Inclusion: \[ q(\mathbf{x}) = \tilde{f}_i(\mathbf{x})q^{\backslash i}(\mathbf{x}) \]
Limitations of Plain EP

- Batch processing of terms: not online
- Can be difficult or expensive to analytically compute ADF step
- Can be expensive to compute and maintain a valid approximation distribution \( q(x) \), which is coherent under marginalization
  - Tree-structured \( q(x) \): \( \sum_{x_j} q(x_i, x_j) = q(x_i) \)
- EP classification degenerates in the presence of noisy features.
- Cannot incorporate denominators
Outline

• Background on expectation propagation (EP)
• Inference on graphical models
  – Extending EP on Bayesian dynamic networks
    • Fixed lag smoothing: wireless signal detection
    • Different approximation techniques: poisson tracking
  – Combining EP with junction tree algorithm on loopy graphs
• Learning conditional graphical models
• Conclusions and future work
Object Tracking

Guess the position of an object given noisy observations
Bayesian Network

e.g. \[ x_t = x_{t-1} + v_t \]  (random walk)

\[ y_t = x_t + \text{noise} \]

want distribution of x’s given y’s
Approximation

\[ p(x, y) = p(x_1) p(y_1 | x_1) \prod_{t>1} p(x_t | x_{t-1}) p(y_t | x_t) \]

\[ q(x) = p(x_1) \tilde{o}_1(x_1) \prod_{t>1} \tilde{p}_{t-1 \rightarrow t}(x_t) \tilde{p}_{t \rightarrow t-1}(x_{t-1}) \tilde{o}_t(x_t) \]

Factorized and Gaussian in x
Message Interpretation

\[ q(x_t) = \tilde{p}_{t-1 \rightarrow t}(x_t) \tilde{\sigma}(x_t) \tilde{p}_{t+1 \rightarrow t}(x_t) \]

= (forward msg)(observation msg)(backward msg)
Extensions of EP

• Instead of batch iterations, use fixed-lag smoothing for online processing.

• Instead of assumed density filtering, use any method for approximate filtering.
  – Examples: unscented Kalman filter (UKF), mixture of Kalman filters

*Turn a deterministic filtering method into a smoothing method!*

All methods can be interpreted as finding linear/Gaussian approximations to original terms.

– Use quadrature or Monte Carlo for term approximations
Bayesian network for Wireless Signal Detection

\[ s_1 \rightarrow x_1 \rightarrow y_1 \]
\[ s_2 \rightarrow x_2 \rightarrow y_2 \]
\[ s_T \rightarrow x_T \rightarrow y_T \]

\( s_i \): Transmitted signals
\( x_i \): Channel coefficients for digital wireless communications
\( y_i \): Received noisy observations
Experimental Results

EP outperforms particle smoothers in efficiency with comparable accuracy.

(Chen, Wang, Liu 2000)
Computational Complexity

- Expectation propagation $O(nLd^2)$
- Stochastic mixture of Kalman filters $O(LMd^2)$
- Rao-blackwised particle smoothers $O(LMNd^2)$

$L$: Length of fixed-lag smooth window
$n$: Number of EP iterations (Typically, 4 or 5)
$d$: Dimension of the parameter vector
$M$: Number of samples in filtering (Often larger than 500)
$N$: Number of samples in smoothing (Larger than 50)

EP is about 5,000 times faster than Rao-blackwised particle smoothers.
Example: Poisson Tracking

- $y_t$ is an integer valued Poisson variate with mean $\exp(x_t)$
Accuracy/Efficiency Tradeoff
Outline

• Background on expectation propagation (EP)
• Inference on graphical models
  – Extending EP on Bayesian dynamic networks
    • Fixed lag smoothing: wireless signal detection
    • Different approximation techniques: poisson tracking
  – Combining EP with junction tree algorithm on loopy graphs
• Learning conditional graphical models
• Conclusions and future work
Inference on Loopy Graphs

Problem: estimate marginal distributions of the variables indexed by the nodes in a loopy graph, e.g., \(p(x_i), i = 1, \ldots, 16\).
4-node Loopy Graph

Joint distribution is product of pairwise potentials for all edges:

\[ p(\mathbf{x}) = \prod_a f_a(\mathbf{x}) \]

Want to approximate \( p(\mathbf{x}) \) by a simpler distribution
BP vs. TreeEP

BP

TreeEP
Junction Tree Representation

\[ q(x) = \frac{\prod_{(j,k) \in T} q(x_j, x_k)}{\prod_{s \in S} q(x_s)} \]
Two Kinds of Edges

• On-tree edges, e.g., \((x_1, x_4)\): *exactly* incorporated into the junction tree

\[
\begin{array}{c}
\text{x}_1 \\
\text{x}_4 \\
\text{x}_2 \\
\text{x}_3
\end{array}
\]

• Off-tree edges, e.g., \((x_1, x_2)\): *approximated* by projecting them onto the tree structure
KL Minimization

- KL minimization $\iff$ moment matching
- Match single and pairwise marginals of

$$x_1 \quad x_4 \quad x_2 \quad x_3$$

$$x_1 \quad x_4 \quad x_2 \quad x_3$$

and

- Reduces to exact inference on single loops
  - Use cutset conditioning
Matching Marginals on Graph

(1) Incorporate edge \((x_3, x_4)\)

(2) Incorporate edge \((x_6, x_7)\)
Drawbacks of Global Propagation by Regular EP

- Update all the cliques even when only incorporating one off-tree edge
  - Computationally expensive
- Store each off-tree data message as a whole tree
  - Require large memory size
Solution: Local Propagation

• Allow $q(x)$ be *non-coherent* during the iterations. It only needs to be coherent in the end.

• Exploit the junction tree representation: only *locally* propagate information within the minimal loop (subtree) that is directly connected to the off-tree edge.
  – Reduce computational complexity
  – Save memory
(1) Incorporate edge \((x_3, x_4)\)

(2) Propagate evidence

On this simple graph, local propagation runs roughly 2 times faster and uses 2 times less memory to store messages than plain EP

(3) Incorporate edge \((x_6, x_7)\)
Tree-EP

• Combine EP with junction algorithm
• Can perform efficiently over hypertrees and hypernodes
Fully-connected graphs

Results are averaged over 10 graphs with randomly generated potentials
• TreeEP performs the same or better than all other methods in both accuracy and efficiency!
TreeEP versus BP and GBP

- TreeEP is always more accurate than BP and is often faster
- TreeEP is much more efficient than GBP and more accurate on some problems
- TreeEP converges more often than BP and GBP
Outline

• Background on expectation propagation (EP)
• Inference on graphical models
• Learning conditional graphical models
  – Extending EP classification to perform feature selection
    • Gene expression classification
  – Training Bayesian conditional random fields
    • Handwritten ink analysis
• Conclusions and future work
Conditional Bayesian Classification Model

Labels: $t$  inputs: $X$  parameters: $w$

Likelihood for the data set:

$$p(t|w, X) = \prod_i p(t_i | x_i, w) = \prod_i \Psi(t_i w^T \phi(x_i))$$

Prior of the classifier $w$:

$$p(w|\alpha) = \prod_i \mathcal{N}(w_i | 0, \alpha_i^{-1})$$

Where $t = \{t_i\}_{i=1}^N$, $X = \{x_i\}_{i=1}^N$, $\Psi(\cdot)$ is a cumulative distribution function for a standard Gaussian.
Evidence and Predictive Distribution

The evidence, i.e., the marginal likelihood of the hyperparameters $\alpha$:

$$p(t|\alpha) = \int p(w, t|\alpha)dw$$

The predictive posterior distribution of the label $t_{N+1}$ for a new input $x_{N+1}$:

$$p(t_{N+1}|x_{N+1}, t) = \int p(t_{N+1}|x_{N+1}, w)p(w|t)dw$$
Limitation of EP classifications

• In the presence of noisy features, the performance of classical conditional Bayesian classifiers, e.g., Bayes Point Machines trained by EP, degenerates.
Automatic Relevance Determination (ARD)

- Give the classifier weight independent Gaussian priors whose variance, $\alpha^{-1}$, controls how far away from zero each weight is allowed to go:
  $$p(w|\alpha) = \prod_{i} \mathcal{N}(w_i|0, \alpha_i^{-1}),$$

- Maximize $p(t|\alpha)$, the marginal likelihood of the model, with respect to $\alpha$.

- Outcome: many elements of $\alpha$ go to infinity, which naturally prunes irrelevant features in the data.
Two Types of Overfitting

• Classical Maximum likelihood:
  – Optimizing the classifier weights $w$ can directly fit noise in the data, resulting in a complicated model.

• Type II Maximum likelihood (ARD):
  – Optimizing the hyperparameters $\alpha$ corresponds to choosing which variables are irrelevant. Choosing one out of exponentially many models can also overfit if we maximize the model marginal likelihood.
Risk of Optimizing $\alpha$

\[ X: \text{Class 1} \quad \text{vs} \quad O: \text{Class 2} \]

\[ \alpha \]

\[ \text{Evd-ARD-1} \]

\[ \text{Bayes Point} \]

\[ \text{Evd-ARD-2} \]
Predictive-ARD

• Choosing the model with the best estimated predictive performance instead of the most probable model.

• Expectation propagation (EP) estimates the leave-one-out predictive performance without performing any expensive cross-validation.
Estimate Predictive Performance

- Predictive posterior given a test data point $\mathbf{x}_{N+1}$
  $$p(t_{N+1} \mid \mathbf{x}_{N+1}, \mathbf{t}) = \int p(t_{N+1} \mid \mathbf{x}_{N+1}, \mathbf{w}) p(\mathbf{w} \mid \mathbf{t}) d\mathbf{w}$$

- EP can estimate predictive leave-one-out error probability
  $$\frac{1}{N} \sum_{i=1}^{N} \left(1 - p(t_i \mid \mathbf{x}_i, \mathbf{t}_{\backslash i})\right) \approx \frac{1}{N} \sum_{i=1}^{N} \left(1 - \int p(t_i \mid \mathbf{x}_i, \mathbf{w}) q(\mathbf{w} \mid \mathbf{t}_{\backslash i}) d\mathbf{w}\right)$$

  - where $q(\mathbf{w} \mid \mathbf{t}_{\backslash i})$ is the approximate posterior of leaving out the $i^{th}$ label.

- EP can also estimate predictive leave-one-out error count

  $$\text{LOO} = \frac{1}{N} \sum_{i=1}^{N} \mathbb{I}(p(t_i \mid \mathbf{x}_i, \mathbf{t}_{\backslash i}) < \frac{1}{2})$$
Comparison of different model selection criteria for ARD training

- 1st row: Test error
- 2nd row: Estimated leave-one-out error probability
- 3rd row: Estimated leave-one-out error counts
- 4th row: Evidence (Model marginal likelihood)
- 5th row: Fraction of selected features

The estimated leave-one-out error probabilities and counts are better correlated with the test error than evidence and sparsity level.

(a) Some graphs:
- 1st row: Test error
- 2nd row: Estimated leave-one-out error probability
- 3rd row: Estimated leave-one-out error counts
- 4th row: Evidence (Model marginal likelihood)
- 5th row: Fraction of selected features
Gene Expression Classification

Task: Classify gene expression datasets into different categories, e.g., normal v.s. cancer

Challenge: Thousands of genes measured in the micro-array data. Only a small subset of genes are probably correlated with the classification task.
Classifying Leukemia Data

- The task: distinguish acute myeloid leukemia (AML) from acute lymphoblastic leukemia (ALL).
- The dataset: 47 and 25 samples of type ALL and AML respectively with 7129 features per sample.
- The dataset was randomly split 100 times into 36 training and 36 testing samples.
Classifying Colon Cancer Data

- The task: distinguish normal and cancer samples
- The dataset: 22 normal and 40 cancer samples with 2000 features per sample.
- The dataset was randomly split 100 times into 50 training and 12 testing samples.
- SVM results from Li et al. 2002
Outline

• Background on expectation propagation (EP)
• Inference on graphical models
• Learning conditional graphical models
  – Extending EP classification to perform feature selection
    • Gene expression classification
  – Training Bayesian conditional random fields
    • Handwritten ink analysis
• Conclusions and future work
Conditional random fields (CRFs)

- Generalize traditional classification model by introducing correlation between labels.

Potential functions: $\Phi_{i,j}(t_i, t_j, x; w)$

- Model data with independence and structure, e.g, web pages, natural languages, and multiple visual objects in a picture.

- Information at one location propagates to other locations.
Learning the parameter $w$ by ML/MAP

Maximum likelihood (ML): maximize the data likelihood

$$p(t|x, w) = \frac{1}{Z(w)} \prod_{\{i,j\} \in \mathcal{E}} \Phi_{i,j}(t_i, t_j, x; w)$$

where

$$Z(w) = \sum_t \prod_{\{i,j\} \in \mathcal{E}} \Phi_{i,j}(t_i, t_j, x; w)$$

Maximum a posterior (MAP): Gaussian prior on $w$

$$\mathcal{N}(w; 0, \text{diag}(\alpha))$$

ML/MAP problem: overfitting to the noise in data.
Bayesian Conditional Networks

• Bayesian training to avoid overfitting

• Need efficient training:
  – The exact posterior of \( w \)

\[
p(w | t, x) \propto p(w)p(t | w, x) = \mathcal{N}(w; 0, \text{diag}(\alpha)) \frac{1}{Z(w)} \prod_{\{i,j\} \in \mathcal{E}} \Phi_{i,j}(t_i, t_j, x; w)
\]

  – The Gaussian approximate posterior of \( w \)

\[
q(w) = \mathcal{N}(w; 0, \text{diag}(\alpha)) \frac{1}{\tilde{Z}(w)} \prod_{\{i,j\} \in \mathcal{E}} \tilde{g}_{i,j}(w)
\]
Two Difficulties for Bayesian Training

• the partition function appears in the denominator

\[
\frac{1}{Z(w)} \, q(w)
\]

where \( q(w) \) is the approximation distribution.

• the partition function is a complex function of \( w \)

\[
Z(w) = \sum_t \prod_{\{i,j\} \in \mathcal{E}} \Phi_{i,j}(t_i, t_j, x; w)
\]
The twist of power EP

- Power EP:
  - To incorporate an exact term with power $k$, process the term with power one and then scale $k$ times the obtained approximate term.
  - Avoid to process the same term $k$ times.

- A twist of power EP:
  - Turn denominator into a numerator
  - Simple observation:

$$q(w) = \mathcal{N}(w; 0, \text{diag}(\alpha)) \frac{\tilde{Z}(w)}{(\tilde{Z}(w))^2} \prod_{\{i,j\} \in \mathcal{E}} \tilde{g}_{ij}(w)$$

- Fix the denominator and only update the numerator
Approximating the partition function

- The parameters \( w \) and the labels \( t \) are intertwined in \( Z(w) \):

\[
Z(w) = \sum_{t} \prod_{k \in \mathcal{E}} \Phi_k(t_i, t_j, x; w)
\]

where \( k = \{i, j\} \) is the index of edges.

The joint distribution of \( w \) and \( t \):

\[
p(w, t) = \prod_{k \in \mathcal{E}} \Phi_k(t_i, t_j, x; w) q^\perp_z(w)
\]

- Factorized approximation:

\[
q(w)q(t) = q^\perp_z(w) \prod_{k \in \mathcal{E}} \tilde{f}_k(w) \prod_{i \in \mathcal{V}} q(t_i)
\]

\[= q^\perp_z(w) \prod_{k \in \mathcal{E}} \tilde{f}_k(w) \prod_{k \in \mathcal{E}} \prod_{i \in \mathcal{V}} \tilde{o}_k(t_i)\]
Flatten Approximation Structure

$\tilde{g}_1(w) \quad \tilde{g}_k(w) \quad \tilde{Z}(w)$

$\tilde{f}_1(w) \quad \tilde{f}_k(w)$

$\tilde{q}_k(w)$

Remove the intermediate level

*Increased efficiency, stability, and accuracy!*
Results on Synthetic Data

- Data generation: first, randomly sample input $x$, fix true parameters $w$, and then sample the labels $t$
- Graphical structure: Four nodes in a simple loop
- Comparing maximum likelihood trained CRF with Bayesian conditional networks: 10 Trials. 100 training examples and 1000 test examples.

BCNs significantly outperformed ML CRFs.
Ink Application: analyzing handwritten organization charts

• Parsing a graph into different components: containers vs boxes
Ink Application: compare BCNs with i.i.d conditional Bayesian classifiers

Results: conditional Bayesian classifiers  BCNs
Ink Application: compare ML CRFs with BCNs

- Comparing maximum likelihood trained CRFs with Bayesian conditional networks (BCNs): 15 Trials, 14 graphs for training and 9 graphs for testing in each trials.
- BCNs significantly outperformed ML CRFs.
Outline

- Background on expectation propagation (EP)
- Inference on graphical models
- Learning conditional graphical models
- Conclusions and future work
  - 4 extensions to EP on 4 types of graphical models & 3 real-world applications
  - Inference: better trade-off between accuracy and efficiency
  - Learning: better generalization than the state of the art.
Conclusion: 4 types of graphical models

<table>
<thead>
<tr>
<th>Bayesian networks</th>
<th>Markov networks</th>
</tr>
</thead>
<tbody>
<tr>
<td>$p(x, y) = \prod_i p(x_i</td>
<td>x_{pa(i)}) \prod_j p(y_j</td>
</tr>
<tr>
<td>$p(t</td>
<td>x, w) = \prod_i p(t_i</td>
</tr>
</tbody>
</table>

- Conditional classification
- Conditional random fields
Conclusion: 4 extensions & 3 applications

- Extending EP on dynamic models:
  - Fixed lag smoothing and embedding different approximation techniques: less computation, and comparable or superior accuracy to sequential Monte Carlo
- Combining EP with junction tree algorithm on loopy graphs:
  - Outperformed belief propagation, naïve mean field, and structure variational methods
- Extending EP classification to perform feature selection:
  - Outperformed traditional ARD, SVM with feature selection
- Training Bayesian conditional random fields
  - Ink analysis: beats ML CRFs
Extended EP algorithms for inference and learning

Outperform state-of-art inference techniques in the trade-off between accuracy and efficiency.

Outperform state-of-art techs on sparse learning and CRFs in generalization performance.
End

Questions?

Now or

yuanqi@mit.edu

Thesis will be online at

www.media.mit.edu/~yuanqi
\[ p(t \mid w, x) = \frac{1}{Z(w)} \prod_a \phi_a(x, y, t) \]
Conclusions

• Extend EP on graphical models:
  – Instead of minimizing KL divergence, use other sensible criteria to generate messages. Effectively turn any deterministic filtering method into a smoothing method.
  – Use quadrature to approximate messages.
  – Local propagation to save the computation and memory in tree structured EP.
Conclusions

- Extended EP algorithms outperform state-of-art inference methods on graphical models in the trade-off between accuracy and efficiency.
Future Work

• More extensions of EP:
  – How to choose a sensible approximation family (e.g. which tree structure)
  – More flexible approximation: mixture of EP?
  – Error bound?
  – Bayesian conditional random fields
  – EP for optimization (generalize max-product)

• More real-world applications, e.g., classification of gene expression data.
Motivation

Task 1: Classify high dimensional datasets with many *irrelevant features*, e.g., normal v.s. cancer microarray data.

Task 2: Sparse Bayesian kernel classifiers for fast test performance.
Outline

- Background on expectation propagation (EP)
- Extending EP on Bayesian dynamic networks
  - Fixed lag smoothing: wireless signal detection
  - Different approximation techniques: poisson tracking
- Combining EP with junction tree algorithm on loopy graphs
- Extending EP classification to perform feature selection
  - Gene expression classification
- Training Bayesian conditional random fields
  - Handwritten ink analysis
- Conclusions and future work
Inference on markov networks

Bayesian networks

Markov networks

\[ p(x, y) = \prod_i p(x_i \mid x_{pa(i)}) \prod_j p(y_j \mid x_{pa(j)}) \]

\[ p(x, y) = \frac{1}{Z} \prod_a \phi_a (x, y) \]

Conditional classification

Conditional random fields
Outline

• Background
  – Bayesian classification model
  – Automatic relevance determination (ARD)
• Risk of Overfitting by optimizing hyperparameters
• Predictive ARD by expectation propagation (EP):
  – Approximate prediction error
  – EP approximation
• Experiments
• Conclusions
Outline

• Background
  – Bayesian classification model
  – Automatic relevance determination (ARD)
• Risk of Overfitting by optimizing hyperparameters
• Predictive ARD by expectation propagation (EP):
  – Approximate prediction error
  – EP approximation
  – Sequential update
• Experiments
• Conclusion
Outline

• Background
  – Bayesian classification model
  – Automatic relevance determination (ARD)
• Risk of Overfitting by optimizing hyperparameters
• Predictive ARD by expectation propagation (EP):
  – Approximate prediction error
  – EP approximation
• Experiments
• Conclusions
Outline

• Background
  – Bayesian classification model
  – Automatic relevance determination (ARD)
• Risk of Overfitting by optimizing hyperparameters
• Predictive ARD by expectation propagation (EP):
  – Approximate prediction error
  – EP approximation
  – Sequential update
• Experiments
• Conclusions
Conclusions

• Maximizing marginal likelihood can lead to overfitting in the model space if there are *a lot* of features.

• We propose Predictive-ARD based on EP for
  – feature selection
  – sparse kernel learning

• In practice Predictive-ARD works better than traditional ARD.
Three Extensions

1. Instead of choosing the approximate term $\tilde{f}_a(x)$ to minimize the following KL divergence:

$$\arg \min_{\tilde{f}_a(x)} D(f_a(x)q^\text{\{a\}}(x) \parallel \tilde{f}_a(x)q^\text{\{a\}}(x))$$

use other criteria.

2. Use numerical approximation to compute moments: Quadrature or Monte Carlo.

3. Allow the tree-structured $q(x)$ to be non-coherent during the iterations. It only needs to be coherent in the end.
Motivation

Current Techniques

What we want

Error

Computational Time
Efficiency vs. Accuracy

- Loopy BP (Factorized EP)
- Extended EP?
- Monte Carlo
Inference on dynamic Bayesian networks

Bayesian networks

\[ p(x, y) = \prod_i p(x_i \mid x_{pa(i)}) \prod_j p(y_j \mid x_{pa(j)}) \]

Markov networks

\[ p(x, y) = \frac{1}{Z} \prod_a \phi_a(x, y) \]

conditional classification

\[ p(t \mid x, w) = \prod_i p(t_i \mid x_i, w) \]

conditional random fields

\[ p(t \mid w, x) = \frac{1}{Z(w)} \prod_a \phi_a(x, y, t) \]
Outline

• Background
  – Bayesian classification model
  – Automatic relevance determination (ARD)
• Risk of Overfitting by optimizing hyperparameters
• Predictive ARD by expectation propagation (EP):
  – Approximate prediction error
  – EP approximation
  – Sequential update
• Experiments
• Conclusions
Conclusions

• Maximizing marginal likelihood can lead to overfitting in the model space if there are a lot of features.
• We propose Predictive-ARD based on EP for
  – feature selection
  – sparse kernel learning
• In practice Predictive-ARD works better than traditional ARD.
Outline

• Background
  – Bayesian classification model
  – Automatic relevance determination (ARD)
• Risk of Overfitting by optimizing hyperparameters
• Predictive ARD by expectation propagation (EP):
  – Approximate prediction error
  – EP approximation
  – Sequential update
• Experiments
• Conclusions
Conclusions

• Maximizing marginal likelihood can lead to overfitting in the model space if there are a lot of features.

• We propose Predictive-ARD based on EP for
  – feature selection
  – sparse kernel learning

• In practice Predictive-ARD works better than traditional ARD.
Outline

• Background
  – Bayesian classification model
  – Automatic relevance determination (ARD)
• Risk of Overfitting by optimizing hyperparameters
• Predictive ARD by expectation propagation (EP):
  – Approximate prediction error
  – EP approximation
  – Sequential update
• Experiments
• Conclusions
Conclusions

• Maximizing marginal likelihood can lead to overfitting in the model space if there are a lot of features.

• We propose Predictive-ARD based on EP for
  – feature selection
  – sparse kernel learning

• In practice Predictive-ARD works better than traditional ARD.
Outline

- Background
  - Bayesian classification model
  - Automatic relevance determination (ARD)
- Risk of Overfitting by optimizing hyperparameters
- Predictive ARD by expectation propagation (EP):
  - Approximate prediction error
  - EP approximation
  - Sequential update
- Experiments
- Conclusion
Motivation

What we want

Current Techniques

Error

Computational Time
Inference on Graphical Models

- Bayesian inference techniques:
  - Belief propagation (BP): Kalman filtering/smoothing, forward-backward algorithm
  - Monte Carlo: Particle filter/smoothers, MCMC

- Loopy BP: typically efficient, but not accurate on general loopy graphs

- Monte Carlo: accurate, but often not efficient
Extended EP vs. Monte Carlo: Accuracy

Mean

Variance
Poisson Tracking Model

\[ p(x_1) \sim N(0,100) \]

\[ p(x_t \mid x_{t-1}) \sim N(x_{t-1},0.01) \]

\[ p(y_t \mid x_t) = \exp(y_t x_t - e^{x_t}) / y_t! \]
Extended-EP Joint Signal Detection and Channel Estimation

• *Turn mixture of Kalman filters into a smoothing method*
• Smoothing over the last $\delta$ observations
• Observations before $(t-\delta)$ act as prior for the current estimation
Bayesian Networks for Adaptive Decoding

The information bits $e_i$ are coded by a convolutional error-correcting encoder.
EP Outperforms Viterbi Decoding

![Graph showing Bit Error Rate (BER) vs. Signal-Noise-Ratio (Eb/N0)]

- viterbi decoding
- ADF
- EP smoothing: δ = 40
Combine Tree-structured Approximation with Junction Tree algorithm

• Combine EP with junction algorithm
• Can perform efficiently over hypertrees and hypernodes
8x8 grids, 10 trials

<table>
<thead>
<tr>
<th>Method</th>
<th>FLOPS</th>
<th>Error</th>
</tr>
</thead>
<tbody>
<tr>
<td>Exact</td>
<td>30,000</td>
<td>0</td>
</tr>
<tr>
<td>TreeEP</td>
<td>300,000</td>
<td>0.149</td>
</tr>
<tr>
<td>BP/double-loop</td>
<td>15,500,000</td>
<td>0.358</td>
</tr>
<tr>
<td>GBP</td>
<td>17,500,000</td>
<td>0.003</td>
</tr>
</tbody>
</table>
### 4-node Graph

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Exact</td>
<td>200</td>
<td>0.474</td>
<td>0.468</td>
<td>0.482</td>
<td>0.536</td>
<td>0</td>
</tr>
<tr>
<td>GBP</td>
<td>2200</td>
<td>0.467</td>
<td>0.459</td>
<td>0.477</td>
<td>0.535</td>
<td>0.008</td>
</tr>
<tr>
<td>TreeVB</td>
<td>11700</td>
<td>0.460</td>
<td>0.460</td>
<td>0.476</td>
<td>0.540</td>
<td>0.014</td>
</tr>
<tr>
<td>BP</td>
<td>500</td>
<td>0.499</td>
<td>0.499</td>
<td>0.5</td>
<td>0.501</td>
<td>0.035</td>
</tr>
<tr>
<td>MF</td>
<td>11500</td>
<td>0.000</td>
<td>0.000</td>
<td>0.094</td>
<td>0.946</td>
<td>0.474</td>
</tr>
</tbody>
</table>

TreeEP = the proposed method  
GBP = generalized belief propagation on triangles  
TreeVB = variational tree  
BP = loopy belief propagation = Factorized EP  
MF = mean-field
Efficiency vs. Accuracy

Loopy BP (Factorized EP)

Extended EP?

Computational Time

Error

Monte Carlo
Outline

• Background on expectation propagation (EP)
• Extending EP on Bayesian dynamic networks
  – Fixed lag smoothing: wireless signal detection
  – Different approximation techniques: poisson tracking
• Combining EP with junction tree algorithm on loopy graphs
• Extending EP classification to perform feature selection
  – Gene expression classification
• Training Bayesian conditional random fields
  – Handwritten ink analysis
• Conclusions and future work
Outline

• Background
  – Bayesian classification model
  – Automatic relevance determination (ARD)

• Risk of Overfitting by optimizing hyperparameters

• Predictive ARD by expectation propagation (EP):
  – Approximate prediction error
  – EP approximation

• Experiments

• Conclusions
Outline: Extending EP classification to perform feature selection

• Background
  – Bayesian classification model
  – Automatic relevance determination (ARD)
• Risk of Overfitting by optimizing hyperparameters
• Predictive ARD by expectation propagation (EP):
  – Approximate prediction error
  – EP approximation
• Experiments
Approximate Leave-One-Out Error

Three key steps:

• Deletion Step: approximate the “leave-one-out” predictive posterior for the \( i^{th} \) point:
  \[
  q^{\backslash i}(w) \propto q(w) / \tilde{f}_i(w) = \prod_{j \neq i} \tilde{f}_j(w)
  \]

• Minimizing the following KL divergence by moment matching:
  \[
  \arg \min_{\tilde{f}_i(w)} KL(f_i(w)q^{\backslash i}(w) \parallel \tilde{f}_i(w)q^{\backslash i}(w))
  \]

• Inclusion: \( q(w) = \tilde{f}_i(w)q^{\backslash i}(w) \)

The key observation: we can use the approximate predictive posterior, obtained in the deletion step, for model selection. No extra computation!
Bayesian Sparse Kernel Classifiers

• Using feature/kernel expansions defined on training data points:
  \[ \phi(x_i) = [1, k(x_i, x_1), \ldots, k(x_i, x_N)]^T \]

• Predictive-ARD-EP trains a classifier that depends on a small subset of the training set.

• Fast test performance.
Test error rates and numbers of relevance or support vectors on breast cancer dataset.

50 partitionings of the data were used. All these methods use the same Gaussian kernel with kernel width $= 5$. The trade-off parameter $C$ in SVM is chosen via 10-fold cross-validation for each partition.
Test error rates on diabetes data

100 partitionings of the data were used. Evidence and Predictive ARD-EPs use the Gaussian kernel with kernel width $= 5$. 

Test error rate
Ink application: using graphical models

• Three steps:
  – Subdivision of pen strokes into fragments,
  – Construction of a conditional random field that only contains pairwise features based on the fragments,
  – Training and inference on the network.
Low rank matrix computation

• Explore the structure of the problem
• Observation: each potential function only constraints the posterior in a subspace
• More efficiency with low-rank matrix computation
Compare to Belief Propagation in ML training

• Similarity: Both propagate probabilistic information between nodes in a graph
• Difference: Bayesian training averages the belief $q(t)$ over the potential parameters $w$, while belief propagation does not.