Boosted Optimization for
Network Classification

Timothy Hancock       Hiroshi Mamitsuka
Bioinformatics Center
Kyoto University
Motivation

We want to construct a classifier that has good performance where the predictor variables have a known network structure.

For Example:

- Our response variable is
- Our predictor variables are

The relationship between \( y \) and \( x \) is unclear and could be obscured by noise.

- Feature selection or regularized methods (lasso etc.) focus on sparsity and may just pick \( x \) and some of its neighbors \( x \).
  - This could lead to very sparse graph features being used to represent the entire network.

Can we use the known network structure to resolve the relationship between \( x \) and \( y \) and improve classification performance?
Outline

1. Problem Setup
   I. Network classifiers and logistic regression
   II. Logistic regression and exponential loss
   III. Network classifiers as an ensemble of factors

2. Methods
   I. Boosting
   II. Expectation Propagation
   III. Boosted Expectation Propagation (BEP)
   IV. Message Passing
   V. Boosted Message Passing (BMP)
   VI. BEP vs BMP
   VII. Convergence & Complexity

3. Experiments

4. Summary & Conclusions
Network Classifiers and Logistic Regression

- The link between network classifiers and logistic regression is well established (Friedman, 1997)
- Each predictor variable is a node: $\beta_k x_k$
- Each edge is an interaction effect: $\beta_{km} x_k x_m$
- $\beta$ are the logistic regression coefficients
- All nodes have an edge with a binary response: $y \in \{-1, 1\}$
- The probability for classifying a binary response is:
  \[
P(y = 1|X) = \frac{e^{F(X)}}{1 + e^{F(X)}}
  \]
- Where $F(X)$ is a linear combination of node and edge terms:
  \[
  F(X) = \sum_k \left( \beta_k x_k + \sum_{m \in \text{ne}(x_k)} \beta_{km} x_k x_m \right)
  \]
Logistic Regression and Exponential Loss

- Optimizing the performance of a logistic regression can be seen as maximizing an exponential potential function.

\[ P(y = 1|X) = \frac{e^{F(X)}}{1 + e^{F(X)}} \]

- Increasing \( F(X) \) in the direction of \( y \), will optimize classification performance.

- Equivalently, as \( y = [-1, 1] \) we could minimize the exponential loss:

\[ \min \left\{ e^{-yF(X)} \right\} \]

- This link between minimizing the exponential loss and maximizing the performance of a logistic regression has been observed with boosted learning (Friedman et al., 2000).
Consider a factorization of our network classifier to minimize the exponential loss,

\[ e^{-yF(X)} = \prod_k e^{-\hat{y}(\beta_k x_k + \sum_{m \in \mathcal{E}(x_k)} \beta_{km} x_k x_m)} = \prod_k e^{-y\hat{f}_k(x_k, \mathcal{E}(x_k), \beta_k)} \]

- The exponential loss is minimized when \( \hat{f}_k \) is maximized in the direction of \( y \). \((y = [-1, 1])\)
- Each \( \hat{f}_k \) can be interpreted an individual classifier.
- Optimizing a linear combination of classifiers to minimize an exponential loss is similar to boosting.
  - Except the structure of all ensemble members is specified in advance and represents local potential functions of a known network.

Can we use the known network structure to estimate each classifier \( \hat{f}_k \) which minimizes the exponential loss over the whole network?
Outline

1. Problem Setup
   I. Network classifiers and logistic regression
   II. Logistic regression and exponential loss
   III. Network classifiers as an ensemble of factors

2. Methods
   I. Boosting
   II. Expectation Propagation (EP)
   III. Boosted Expectation Propagation (BEP)
   IV. Message Passing (MP)
   V. Boosted Message Passing (BMP)
   VI. BEP vs BMP
   VII. Convergence & Complexity

3. Experiments

4. Summary & Conclusions
Boosting

- Boosting constructs a linear combination $F_M(X)$ through a stage-wise addition of individual classifiers $f_m(X)$:

$$F_M(X) = \sum_{m=1}^{M} c_m f_m(X)$$

where each new classifier $f_m(X)$ found through minimization of an exponential loss:

$$\arg\min_{c_m} \left\{ e^{-y(F_{m-1}(X)+c_m f_m(X))} \right\} = \arg\min_{c_m} \left\{ w_{m-1} e^{-yc_m f_m(X)} \right\}$$

- The weights at each iteration $w_{m-1}$ are the errors of the current ensemble $F_{M-1}(X)$.

- The boosted coefficients $c_m$ weight the importance of each newly added model $f_m(X)$ to the entire ensemble:

$$e_m = E_{w_{m-1}}[1_{y\neq \hat{f}_m(X)}] \quad \text{and} \quad c_m = \frac{1}{2} \log \frac{1-e_m}{e_m}$$
Network Inference

Message Passing and Expectation Propagation are network inference algorithms that work on factor graphs:

- Starting from a factorization of pairwise loss functions:

\[ f_{ik} = f_{ik}(x_k, x_i) = e^{-y(\beta_k x_k + \beta_{ik} x_k x_i)} \]

- The contribution of \( x_k \) to the whole network is:

\[ q_k(x_k) = \prod_{i \in ne(x_k)} f_{ik} \]

- The entire network can be re-written as:

\[ p(X) = \prod_k q_k(x_k) = \prod_k \prod_{i \in ne(x_k)} f_{ik} \]

From this factorization we can directly use Expectation Propagation or Message Passing algorithms to optimize the performance of our network classifier.
Expectation Propagation (EP)

Expectation Propagation (EP) minimizes the Kullback-Leibler divergence of a factorized distribution by iteratively refining the estimates of each factor (Minka, 2001).

Given a factorized distribution: \( P(x_1, \ldots, x_m) = \prod_k f_k \)

**Step 1:** Remove the current estimate of \( f_k \)

\[ \hat{p}(X) / f_k = \hat{p}(X) / \hat{f}_k \]

**Step 2:** Re-estimate \( f_k \) given the current estimates of all other factors

\[ \hat{f}_k = \max \left\{ \hat{p}(X) / f_k f_k \right\} \]

**Step 3:** Insert the new \( f_k \) back into the full distribution

\[ \hat{p}(X) = Z_k \hat{p}(X) / f_k \hat{f}_k \]
**EP on a Network Classifier**

If we consider the factorized form of our network classifier:

$$p(X) = \prod_k \prod_{i \in \text{ne}(x_k)} f_{ik} \quad \text{where} \quad f_{ik} = e^{-y(\beta_i x_i + \beta_k x_i x_k)}$$

We can define an EP algorithm to estimate the classifier parameters $\beta$.

**Step 1:** Remove the current estimate of $f_{ik}$

$$\hat{p}(X)/\hat{f}_{ik} \propto e^{-y(F(X) - \hat{f}_{ik})}$$

**Step 2:** Re-estimate $f_{ik}$ given the current estimates of all other factors

$$\hat{f}_{ik} = \min_{\beta_{ik}} \left\{ \frac{\hat{p}(X)}{\hat{f}_{ik}} e^{-y f_{ik}} \right\}$$

**Step 3:** Insert the new $f_{ik}$ back into the full distribution

$$\hat{p}(X) \propto \hat{p}(X)/\hat{f}_{ik} e^{-y \hat{f}_{ik}}$$

**Step 2** is the minimization of the exponential loss of $f_{ik}$ weighted by the exponential loss of all other factors.

**Step 2** is analogous to a **Boosted Addition** of $f_{ik}$ to the entire network classifier.
Boosted Expectation Propagation (BEP)

Defines a **Boosted update** as the optimization step within an **Expectation Propagation** algorithm:

**Step 1:** Remove the current estimate of \( f_{ik} \)

\[
\hat{p}(X)/\hat{f}_{ik} \propto e^{-y(F(X) - c_{ik}\hat{f}_{ik})}
\]

**Step 2:** Re-estimate \( \hat{f}_{ik} \) given the current exponential loss from all other factors

\[
\hat{f}_{ik} = \min_{c_{ik}} \left\{ w_{ik}e^{-y c_{ik}\hat{f}_{ik}} \right\}
\]

**Step 3:** Insert the new \( \hat{f}_{ik} \) back into the full distribution

\[
\hat{p}(X) \propto w_{ik}e^{-y c_{ik}\hat{f}_{ik}}
\]

The **boosted update** introduces a new parameter \( c_{ik} \) for each \( f_{ik} \) which weights the importance of each factor to the network classifier.
Message Passing (MP)

Message Passing algorithms assume that all network information needed to estimate the distribution of node is contained within its immediate neighbors. - We use the max-product algorithm (Kschischang et al., 2001)

Given a factor graph:

\[ P(x_1, \ldots, x_m) = \prod_k f_k \]

On a factor graph the max-product algorithm defines 2 type of messages:

1) From a node \( x_i \) to a factor \( f_{ik} \):

\[ \mu_{x_i \rightarrow f_{ik}} = \prod_{j \in \text{ne}(x_i) \atop j \neq k} \mu_{f_{ji} \rightarrow x_i} \]

2) From a factor \( f_{ik} \) to a node \( x_k \):

\[ \mu_{f_{ik} \rightarrow x_k} = \max \left\{ f_{ik} \prod_{j \in \text{ne}(f_{ik}) \atop j \neq k} \mu_{x_j \rightarrow f_{ik}} \right\} \]
MP on a Network Classifier

If we consider the factorized form of our network classifier:

\[ p(X) = \prod_k \prod_{i \in \text{ne}(x_k)} f_{ik} \quad \text{where} \quad f_{ik} = e^{-y(\beta_i x_i + \beta_{ik} x_i x_k)} \]

We can define a max-product algorithm to estimate the classifier parameters \( \beta \)

1) From a node \( x_i \) to a factor \( f_{ik} \):

\[ \mu_{x_i \rightarrow f_{ik}} = \prod_{j \in \text{ne}(x_i) \atop j \neq k} \mu_{f_{ji} \rightarrow x_i} \]

2) From a factor \( f_{ik} \) to a node \( x_k \):

\[ \mu_{f_{ik} \rightarrow x_k} = \min \left\{ e^{-y f_{ik}} \prod_{j \in \text{ne}(f_{ik}) \atop j \neq k} \mu_{x_j \rightarrow f_{ik}} \right\} \]

Step 2 is analogous to a Boosted Addition of \( f_{ik} \) weighted by the exponential loss of the neighboring nodes.

2) is the minimization of the exponential loss of \( f_{ik} \) weighted by the exponential loss of the neighboring nodes.
Boosted Message Passing (BMP)

Defines a **Boosted** update as the maximization step within an loopy max-product **Message Passing** algorithm.

1) From a node $x_i$ to a factor $f_{ik}$:

$$\mu_{x_i \rightarrow f_{ik}} = \prod_{j \in ne(x_i), j \neq i} \mu_{f_{ji} \rightarrow x_i}$$

2) From a factor $f_{ik}$ to a node $x_k$:

$$\mu_{f_{ik} \rightarrow x_k} = \min \left\{ e^{-yc_{ik}f_{ik}} \prod_{j \in ne(f_{ik}), j \neq k} \mu_{x_j \rightarrow f_{ik}} \right\}$$

The **boosted update** introduces a new parameter $c_{ik}$ for each $f_{ik}$ which weights the importance of each factor to the network classifier.
**BEP vs BMP**

**BEP** updates each factor based on the error of the entire network:

- Base network
- Remove each node
- Re-estimate & re-insert

The network is single ensemble of factors

**BMP** updates each factor based only on the error within the local network:

- Base network
- 1st iteration (estimate and propagate)
- 2nd iteration (estimate and propagate)

Separate ensembles are built along each edge in both directions
Convergence

Both MP and EP seek to minimize the Kullback-Leibler divergence. For classification we seek to minimize the Conditional Kullback-Leibler divergence (CKL):

$$CKL(P||Q) = \sum_{y,X} P(X|y) \log \frac{P(X|y)}{Q(X|y)}$$

given,

$$P(X|y) = \frac{1}{Z(X)} \prod_k e^{-y f_k} \quad \text{and} \quad Q(X|y) = \prod_k q_k$$

Then the CKL is:

$$CKL = -\log Z(X) - \sum_{y,X} y F(X) P(X|y)$$

Boosting only increases $f_k$ linearly, $P(X|y) < 1$ and $Z(X)$ decreases exponentially.
Simulation Experiments

We assess the performance of BEP and BMP to classify a 2D grid structured known exponentially distributed network ($y = 1$):

$$p(X) = \prod_i e^{\theta_i x_i + \sum_{j \in \text{ne}(x_i)} \theta_{ij} x_i x_j}$$

embedded within a uniform random noise distribution ($y = -1$).

We define a network strength:

$$\alpha \in [0.5, 0.75, 1, \ldots, 3]$$

to scale the network coefficients:

$$\theta_i = \alpha \theta_i$$

We compare BEP and BMP on 3 grid sizes (8x8, 10x10, 12x12) with

- Standard logistic regression (LNC)
- Logistic Regression with RIDGE, LASSO and ELASTIC net penalties.
- Simple aggregation over all network factors (FNC)

Using 5x5 fold cross-validation and the area under a ROC curve (AUC).
2D Grid Simulation Results

- BMP performs best
- BEP performance is equivalent to penalized approaches
- As network strength increases all methods will perform around the same.
Gene Network Example

KEGG yeast carbohydrate metabolism network

203 genes & 1773 interactions

Classify “heat shock” specific response from other environmental stresses using the benchmark Gasch microarray data (Gasch et al., 2000).

<table>
<thead>
<tr>
<th>Model</th>
<th>AUC</th>
</tr>
</thead>
<tbody>
<tr>
<td>RIDGE</td>
<td>0.86 ± 0.028</td>
</tr>
<tr>
<td>LASSO</td>
<td>0.865 ± 0.022</td>
</tr>
<tr>
<td>ENET</td>
<td>0.88 ± 0.021</td>
</tr>
<tr>
<td>BEP</td>
<td>0.87 ± 0.022</td>
</tr>
<tr>
<td>BMP</td>
<td>0.94 ± 0.013</td>
</tr>
</tbody>
</table>
Summary

- We exploit the similarity between logistic regression, boosting and message passing algorithms and propose two novel network classifiers – BEP and BMP.
- BMP is shown to outperform commonly used penalized approaches and BEP shows equivalent performance.
- The results highlight the advantage of explicitly using the known network structure in constructing a classifier.
- BEP and BMP are flexible as they work on a factor graph and can be extended to use topological features of biological networks such as reactions, pathways or GO function information.
Acknowledgments

- All at the Pathway Engineering Laboratory in Kyoto University.
- Funding from JSPS and BIRD

Thanks to All!