libDAI - a FOSS library for Discrete Approximate Inference methods

Joris Mooij\textsuperscript{1}

\textsuperscript{1}Max Planck Institute for Biological Cybernetics
Tübingen, Germany
joris.mooij@tuebingen.mpg.de

Whistler, December 12, 2008
Graphical models

- **Bayesian Networks**
  \[
P(x) = \prod_i P(x_i | x_{pa(i)})
\]
  where \(pa(i)\) are the *parents* of a node \(i\) in a DAG;

- **Markov Random Fields**
  \[
P(x) = \frac{1}{Z} \prod_{C \in \mathcal{C}} \psi_C(x_C)
\]
  where \(\mathcal{C}\) are *cliques* of an undirected graph;

- **Generalization of both: factor graphs**
  \[
P(x) = \frac{1}{Z} \prod_{I \in \mathcal{F}} \psi_I(x_I)
\]
  where \(\mathcal{F}\) are *factor* nodes in a *factor graph.*
Graphical models

- **Bayesian Networks**

$$P(x) = \prod_i P(x_i \mid x_{\text{pa}(i)})$$

where $\text{pa}(i)$ are the *parents* of a node $i$ in a DAG;

- **Markov Random Fields**

$$P(x) = \frac{1}{Z} \prod_{C \in C} \psi_C(x_C)$$

where $C$ are *cliques* of an undirected graph;

- **Generalization of both: factor graphs**

$$P(x) = \frac{1}{Z} \prod_{I \in F} \psi_I(x_I)$$

where $F$ are *factor* nodes in a factor graph.
Graphical models

- **Bayesian Networks**

  \[ P(\mathbf{x}) = \prod_i P(x_i \mid x_{pa(i)}) \]

  where \( pa(i) \) are the *parents* of a node \( i \) in a DAG;

- **Markov Random Fields**

  \[ P(\mathbf{x}) = \frac{1}{Z} \prod_C \psi_C(\mathbf{x}_C) \]

  where \( C \) are *cliques* of an undirected graph;

- **Generalization of both: factor graphs**

  \[ P(\mathbf{x}) = \frac{1}{Z} \prod_{I \in \mathcal{F}} \psi_I(x_I) \]

  where \( \mathcal{F} \) are *factor* nodes in a *factor graph*. 
Approximate inference in graphical models

Given a factor graph

\[ P(x) = \frac{1}{Z} \prod_{I \in \mathcal{F}} \psi_I(x_I), \]

the following inference tasks are important:

- **Calculate the partition sum:**

  \[ Z = \sum_{x_1} \cdots \sum_{x_N} \prod_{I \in \mathcal{F}} \psi_I(x_I) \]

- **Calculate the marginal distribution of a subset of variables \( \{x_i\}_{i \in A} \):**

  \[ P(x_A) = \frac{1}{Z} \sum_{x \setminus A} \prod_{I \in \mathcal{F}} \psi_I(x_I) \]

- **Calculate the MAP state that has maximal probability mass:**

  \[ \text{argmax}_x \prod_{I \in \mathcal{F}} \psi_I(x_I) \]
Given a factor graph

$$\mathbb{P}(\mathbf{x}) = \frac{1}{Z} \prod_{l \in \mathcal{F}} \psi_l(x_l),$$

the following inference tasks are important:

- Calculate the **partition sum**:

  $$Z = \sum_{x_1} \cdots \sum_{x_N} \prod_{l \in \mathcal{F}} \psi_l(x_l)$$

- Calculate the **marginal** distribution of a subset of variables \(\{x_i\}_{i \in A}\):

  $$\mathbb{P}(\mathbf{x}_A) = \frac{1}{Z} \sum_{\mathbf{x} \setminus A} \prod_{l \in \mathcal{F}} \psi_l(x_l)$$

- Calculate the **MAP state** that has maximal probability mass:

  $$\arg\max_{x} \prod_{l \in \mathcal{F}} \psi_l(x_l)$$
Given a factor graph

\[ P(x) = \frac{1}{Z} \prod_{l \in F} \psi_l(x_l), \]

the following inference tasks are important:

- **Calculate the partition sum:**
  \[ Z = \sum_{x_1} \cdots \sum_{x_N} \prod_{l \in F} \psi_l(x_l) \]

- **Calculate the marginal distribution of a subset of variables \( \{x_i\}_{i \in A} \):**
  \[ P(x_A) = \frac{1}{Z} \sum_{x_{\setminus A}} \prod_{l \in F} \psi_l(x_l) \]

- **Calculate the MAP state that has maximal probability mass:**
  \[ \arg\max_x \prod_{l \in F} \psi_l(x_l) \]
libDAI provides implementations of various algorithms that solve these inference tasks (either exactly or approximately) for factor graphs with discrete variables.

Currently, libDAI contains implementations of the following algorithms:

- Exact inference by brute force enumeration
- Exact inference by junction-tree methods
- Mean Field
- (Loopy) Belief Propagation
- Tree Expectation Propagation (Minka & Qi 2004)
- Generalized Belief Propagation (Yedidia, Freeman & Weiss 2005)
- Double-loop GBP (Heskes, Albers & Kappen 2003)
- Loop Corrected Belief Propagation (Mooij & Kappen 2007; Montanari & Rizzo 2005)
- Gibbs sampling
The following algorithms are planned to be added to the next release:

- Iterative Join-Graph Propagation (Dechter, Kask & Mateescu, 2002)
- Tree Reweighted approximations/bounds (Wainwright, Jaakkola & Willsky, 2005)
- Methods for bounding marginals:
  - Box Propagation (Mooij & Kappen, 2008)
  - Ihler’s BP accuracy bounds (Ihler, 2007)
  - Bound Propagation (Leisink & Kappen, 2003)
Key features of libDAI are:

- Free and open source (license: GPL v2+)
- C++ library (MatLab would be orders of magnitude slower)
- Command line interface and a (rudimentary) MatLab interface
- Modular design: easy to add algorithms
- Doxygen documentation (target for next release)
- Compiles out-of-the-box with GCC versions 4.1 and higher under GNU/Linux, and also with MS Visual Studio 2008 under Windows.
libDAI is targeted at researchers that have a good understanding of graphical models.

The best way to use libDAI is by writing C++ code that invokes the library.

libDAI can be used to implement new approximate inference algorithms and to easily compare the performance with existing methods.

Non-features of libDAI

- No learning algorithms
- Only supports libDAI factor graph file format
- No GUI provided
- Limited visualization functionalities
Releases can also be obtained from MLOSS at

http://mloss.org/software/view/77/

The newest code can be obtained from the public git repository

git://git.tuebingen.mpg.de/libdai.git

which can be accessed with a web browser as well at

http://git.tuebingen.mpg.de/libdai

Last release: 0.2.2 - September 30, 2008
Other open source software packages supporting both directed and undirected graphical models are:

- **Bayes Net Toolbox (BNT)** by Kevin Murphy, written in MatLab/C
- **Probabilistic Networks Library (PNL)** from Intel, written in C++

<table>
<thead>
<tr>
<th></th>
<th>libDAI</th>
<th>BNT</th>
<th>PNL</th>
</tr>
</thead>
<tbody>
<tr>
<td>Exact (brute force)</td>
<td>+</td>
<td>+</td>
<td>+</td>
</tr>
<tr>
<td>Exact (junction tree)</td>
<td>+</td>
<td>+</td>
<td>+</td>
</tr>
<tr>
<td>(Loopy) Belief Propagation</td>
<td>+</td>
<td>+</td>
<td>+</td>
</tr>
<tr>
<td>Gibbs sampling</td>
<td>+</td>
<td>+</td>
<td>+</td>
</tr>
<tr>
<td>Mean Field</td>
<td>+</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>Tree Expectation Propagation</td>
<td>+</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>Generalized Belief Propagation</td>
<td>+</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>Loop Corrected Belief Propagation</td>
<td>+</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>Continuous variables</td>
<td>-</td>
<td>+</td>
<td>+</td>
</tr>
<tr>
<td>Dynamic Bayes Nets</td>
<td>-</td>
<td>+</td>
<td>+</td>
</tr>
<tr>
<td>Parameter Learning</td>
<td>-</td>
<td>+</td>
<td>+</td>
</tr>
<tr>
<td>Structure Learning</td>
<td>-</td>
<td>+</td>
<td>+</td>
</tr>
</tbody>
</table>
Thank you for your attention!

Thanks to all people who contributed to libDAI! (Martijn Leisink, Giuseppe Passino, Frederik Eaton, Bastian Wemmenhove, Christian Wojek, Claudio Lima, Jiuxiang Hu, Peter Gober)

In particular, thanks to Martijn Leisink for providing the basis from which libDAI has been derived.
Acknowledgments

This work is part of the Interactive Collaborative Information Systems (ICIS) project, supported by the Dutch Ministry of Economic Affairs, grant BSIK03024.

References


