LARGE SCALE MODEL-BASED MACHINE LEARNING

Tom Diethe
Microsoft Research Cambridge
LSOLDM Workshop 2013
MODEL-BASED MACHINE LEARNING

TRADITIONAL

“HOW DO I MAP MY PROBLEM INTO STANDARD TOOLS”? 
MODEL-BASED MACHINE LEARNING

TRADITIONAL

“How do I map my problem into standard tools”?

MODEL-BASED

“What is the model that represents my problem”? 
MODEL-BASED MACHINE LEARNING

TRADITIONAL

“How do I map my problem into standard tools”?

MODEL-BASED

“What is the model that represents my problem”?

Goal:
A *single* development framework which supports the creation of a wide range of bespoke models
STAGES OF MBML
STAGES OF MBML

1. **Build a model**: Joint probability distribution of all of the relevant variables (e.g. as a graph)
STAGES OF MBML

1. **Build a model**: Joint probability distribution of all of the relevant variables (e.g. as a graph)
2. **Incorporate the observed data**
STAGES OF MBML

1. **Build a model**: Joint probability distribution of all of the relevant variables (e.g. as a graph)

2. **Incorporate the observed data**

3. **Compute the distributions over the desired variables**: Inference
STAGES OF MBML

1. **BUILD A MODEL**: JOINT PROBABILITY DISTRIBUTION OF ALL OF THE RELEVANT VARIABLES (E.G. AS A GRAPH)

2. **INCORPORATE THE OBSERVED DATA**

3. **COMPUTE THE DISTRIBUTIONS OVER THE DESIRED VARIABLES**: INFERENCE

- **Iterate 2 and 3 in real-time applications**
STAGES OF MBML

1. **Build a model**: Joint probability distribution of all of the relevant variables (e.g. as a graph)

2. **Incorporate the observed data**

3. **Compute the distributions over the desired variables**: Inference

- **Iterate 2 and 3** in real-time applications
- **Extend model as required**
POTENTIAL BENEFITS OF MBML
POTENTIAL BENEFITS OF MBML

- Models optimised for each new application
POTENTIAL BENEFITS OF MBML

• **Models optimised for each new application**
• **Transparent functionality**
POTENTIAL BENEFITS OF MBML

- Models optimised for each new application
- Transparent functionality
- Segregate model from training/inference code
POTENTIAL BENEFITS OF MBML

- **Models optimised for each new application**
- **Transparent functionality**
- **Segregate model from training/inference code**
- **Newcomers learn one modelling environment**
COST OF BAYESIAN INFERENCE
COST OF BAYESIAN INFERENCE

- Classical algorithms tend to be slow and conservative
  - Only needed if you really care about the full posterior
COST OF BAYESIAN INFERENCE

- Classical algorithms tend to be slow and conservative
  - Only needed if you really care about the full posterior
- Modern algorithms are optimistic and fast
  - Competitive with gradient descent and EM
  - Practical for real-world machine learning
DETERMINISTIC APPROXIMATIONS
DETERMINISTIC APPROXIMATIONS

- Two algorithms:
  - Expectation Propagation
  - Variational Message Passing
DETERMINISTIC APPROXIMATIONS

- **Two algorithms:**
  - Expectation Propagation
  - Variational Message Passing
- **Choose an approximating density for each variable**
DETERMINISTIC APPROXIMATIONS

- **Two algorithms:**
  - Expectation Propagation
  - Variational Message Passing

- Choose an approximating density for each variable

- A fully-factorized model is fit to the original model
DETERMINISTIC APPROXIMATIONS

- **Two algorithms:**
  - **Expectation Propagation**
  - **Variational Message Passing**

- **Choose an approximating density for each variable**

- **A fully-factorized model is fit to the original model**

- **Microsoft Research have developed a tool for inference in graphical models:** Infer.NET
INFER.NET – HOW IT WORKS

Model (small .NET program)

Infer.NET Compiler

Optimized Runtime Inference Code

Data

Answers!
1. **Specify your machine learning problem as a probabilistic model in a small .NET program.**
INFER.NET – HOW IT WORKS

1. **Specify your machine learning problem as a probabilistic model in a small .NET program.**

2. **Use Infer.NET to compile the model into optimized runtime code.**

   ![Diagram](image-url)
1. **Specify your machine learning problem as a probabilistic model in a small .NET program.**

2. **Use Infer.NET to compile the model into optimized runtime code.**

3. **Run the code to make inferences on your data automatically.**
INFERENC IN INFER.NET

• Inference process is iterative
INFORMATION IN INFER.NET

- INFEERENCE PROCESS IS ITERATIVE
- INFEERENCE RESULTS ARE DETERMINISTIC
INFERENC ETHEN IN INFERNET

- Inference process is iterative
- Inference results are deterministic
- Inference may not converge
STANDARD MODELS SUPPORTED

- Clustering
- Classification (linear/non-linear/multi-class)
- Logistic regression
- Recommendation
- Latent Dirichlet Allocation (LDA)
- Factor analysis and Principal Component Analysis (PCA)
- Discrete Bayesian networks
- Ranking models
- Hidden Markov Models
- Gaussian Processes
- Sparse models (e.g. classifiers, Sparse PCA)
- Hierarchical models
SCALING UP INFERENCE

• How can we speed things up even more?
SCALING UP INFEERENCE

• **How can we speed things up even more?**

• **Variable Sharing → batch processing & parallel inference**
SCALING UP INFERENCE

- How can we speed things up even more?
- Variable Sharing $\Rightarrow$ batch processing & parallel inference
- Sparse Message Passing in EP
SCALING UP INFEERENCE

• How can we speed things up even more?

• Variable Sharing ➔ Batch Processing & Parallel Inference

• Sparse Message Passing in EP

• Customised Message Operators
SCALING UP INFERENCEx

• How can we speed things up even more?

• Variable Sharing ➔ batch processing & parallel inference

• Sparse Message Passing in EP

• Customised Message Operators

• More generally: Community/Personalisation Model
COMMUNITY MODELS

TRAINING DATA → Community Model

Community Model

Individual Model

Individual Model

Individual Model

Individual Model

Individual Model

ADF

ADF

ADF

ADF

Decisions

Decisions

Decisions

Decisions

Decisions

Decisions

Slow, Accurate

Fast, Approximate
### AN EXAMPLE: RECOMMENDATION

<table>
<thead>
<tr>
<th>Item 10,000</th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>User 10 Million</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
AN EXAMPLE: RECOMMENDATION

<table>
<thead>
<tr>
<th>User 10 Million</th>
<th>Item 10,000</th>
</tr>
</thead>
</table>

- **Matchbox: Large Scale Bayesian Recommendations. David Stern, Ralf Herbrich, and Thore Graepel (WWW 2009)**
MATCHBOX
MATCHBOX

user traits

$N \rightarrow S$
MATCHBOX

Factor

user traits

$N$

$S$
MATCHBOX

Factor

$S$

Variable

user traits

user bias
MATCHBOX

**User**

- **user traits**

**Factor**

- $N$

**Variable**

- $S$

- $N$

- **user bias**
MATCHBOX

User
user traits

Factor

\( \mathcal{N} \)

Variable

\( S \)

Plate

\( \mathcal{N} \)

user bias
MATCHBOX

User

user traits

\( S \)

Variable

\( \mathcal{N} \)

Factor

user bias

\( \mathcal{N} \)

Plate
MATCHBOX

User

user traits

 Trait

user bias
MATCHBOX

- **User**
  - user traits
  - \( s \) (user bias)

- **Item**
  - item traits
  - \( t \) (item bias)

\( \mathcal{N} \)
MATCHBOX

User

\textit{user traits}

\[ \mathcal{N} \]

\[ s \]

\[ \mathcal{N} \]

\textit{user bias}

\[ \mathcal{N} \]

Item

\textit{item traits}

\[ \mathcal{N} \]

\[ t \]

\[ \mathcal{N} \]

\textit{item bias}

\[ \mathcal{N} \]

Trait

\[ \mathcal{N} \]

\[ r \]

\[ \mathcal{N} \]

\textit{noise}

\[ \text{affinity} \]
MATCHBOX

User

\[ s \]

user traits

\[ \mathcal{N} \]

\[ \mathcal{N} \]

\[ \mathcal{N} \]

\[ \mathcal{N} \]

user bias

Item

\[ t \]

item traits

\[ \mathcal{N} \]

\[ \mathcal{N} \]

\[ \mathcal{N} \]

\[ \mathcal{N} \]

item bias

\[ r \]

affinity

\[ \mathcal{N} \]

noise

product

\[ \text{sum} \]

\[ \mathcal{N} \]
MATCHBOX

User

$s$

user traits

$\mathcal{N}$

product

Item

$t$

item traits

$\mathcal{N}$

noise

Level

$\theta$

user thresholds

$r > \theta$

$\mathcal{N}$

$\mathcal{N}$

user bias

sum

item bias

$\mathcal{N}$

affinity

$\mathcal{N}$
MATCHBOX

- COMMUNITY TRAINING:
  - RATINGS ARE OBSERVED
  - $T$ IS INFERRED
MATCHBOX

- **COMMUNITY TRAINING**:
  - RATINGS ARE OBSERVED
  - T IS INFERRED

- **PERSONALISATION**:
  - RATINGS AND T ARE OBSERVED
  - S IS INFERRED

---

![Diagram of MATCHBOX model]

- **User**
  - user traits $\mathcal{N}$
  - user bias $\mathcal{N}$
  - user thresholds $r > \theta$
  - affinity $r$
  - $s$

- **Item**
  - item traits $\mathcal{N}$
  - item bias $\mathcal{N}$
  - noise $\mathcal{N}$
  - $t$

- **Level**
  - sum
  - product
MATCHBOX

- **COMMUNITY TRAINING:**
  - RATINGS ARE OBSERVED
  - T IS INFERRED

- **PERSONALISATION:**
  - RATINGS AND T ARE OBSERVED
  - S IS INFERRED

- **FOR RECOMMENDATION:**
  - S AND T ARE OBSERVED
  - THE RATING IS PREDICTED
MATCHBOX

- **COMMUNITY TRAINING:**
  - RATINGS ARE OBSERVED
  - T IS INFERRED

- **PERSONALISATION:**
  - RATINGS AND T ARE OBSERVED
  - S IS INFERRED

- **FOR RECOMMENDATION:**
  - S AND T ARE OBSERVED
  - THE RATING IS PREDICTED

```csharp
using (Variable.ForEach(obs)) {
    var product = Variable.Array<double>(trait);
    product[trait] = s[userOf[rating]][trait] * t[itemOf[rating]][trait];
    var bias = sBias[userOf[rating]] + tBias[itemOf[rating]];
    var sum = bias + Variable.Sum(product);
    var r = Variable.GaussianFromMeanAndVariance(sum, noise);
    rGTTheta[rating][level] = (r > theta[userOf[rating]][level]);
}
```
RECOMMENDER - MATCHBOX

DEMO
- Over 50 million users
- Serves more than 100 million requests per day
- Spans verticals: games, TV programmes, movies
SUMMARY

- **Model Based Machine Learning** is well suited to **Large Scale and Online applications**

- **The Community/Personalisation model** is a general paradigm for **efficient Bayesian inference**

- **Infer.NET** is a framework for running **Bayesian inference in graphical models**

[http://research.microsoft.com/infernet](http://research.microsoft.com/infernet)

John Winn, Tom Minka, John Guiver, et al.