iPMR

Our Journey

Graphical Models	Decision Theory	Learning Probabilistic Models	Mixture and Factor Models	Markov Models	Approximate Inference

- Approximate Inference and Learning
  - Sampling (MLPR), Gibbs Sampling (PMR)
  - Variational Methods (MLPR)
  - Message Passing (PMR)
  - Contrastive Divergence and Stochastic Gradient Methods (PMR)

Factor Graphs

- Remember we often write our models in the form

$$P(\mathbf{v}, \mathbf{h}) = \frac{1}{Z} \exp(-E(\mathbf{v}, \mathbf{h}))$$

- We found we could use the elimination algorithm to do inference in this.
- This involved passing messages.
- Worked well in trees.
- But for other network structures it got complicated quickly:
- e.g. eliminating a node causes a joint message to all the nodes it connects to (causing a joint factor).
- However what if we pretended the messages were independent.
- Use as an approximation scheme.

The Free Energy

We have

$$P(\mathbf{x}) = \sum_{\mathbf{y}} P(\mathbf{x}, \mathbf{y}) = \sum_{\mathbf{y}} \frac{1}{Z} \exp(-E(\mathbf{x}, \mathbf{y}))$$

and so we can show $\log(P(\mathbf{x}))$ can be written as

$$\arg\max_{Q} \left[-\sum_{\mathbf{y}} Q(\mathbf{y}) E(\mathbf{x}, \mathbf{y}) - \sum_{\mathbf{y}} Q(\mathbf{y}) \log Q(\mathbf{y}) \right] - \log Z.$$

The term $\sum_{\mathbf{y}} Q(\mathbf{y}) E(\mathbf{x}, \mathbf{y}) + \sum_{\mathbf{y}} Q(\mathbf{y}) \log Q(\mathbf{y})$ is the "free energy" (up to a constant).

Want to maximize this to get best $Q(\mathbf{y})$ (the optimum equals $P(\mathbf{y})$). But not always easy. So we approximate. Many different approximate free energies.

Variational Approximation

$$\arg\max_{Q}\left[-\sum_{\mathbf{y}} Q(\mathbf{y})E(\mathbf{x},\mathbf{y}) - \sum_{\mathbf{y}} Q(\mathbf{y})\log Q(\mathbf{y})\right] - \log Z.$$

- Approximation method 1:
  - Make the Q distribution simpler than a general distribution.
  - E.g. Make Q factorise for each y component
  - Optimize over restricted form for Q
- Variational Approximation
  - Leads to variational message passing

Bethe Free Energy

$$\arg\max_{Q} \left[-\sum_{\mathbf{y}} Q(\mathbf{y}) E(\mathbf{x}, \mathbf{y}) - \sum_{\mathbf{y}} Q(\mathbf{y}) \log Q(\mathbf{y}) \right] - \log Z.$$

- Alternatively we can replace the sum over the joint ***y*** by the sum over all pairs of *y* variables.
- This is the Bethe Free Energy in Physics.
- Turns out if we pretend elimination messages are independent, then if it converges, it converges to a fixed point of the Bethe Free Energy.
- This is called loopy belief propagation.

Expectation Propagation

- We did elimination message passing on discrete and Gaussian systems.
- Less easy with other systems: even on trees, the messages create potentials that are not in exponential family.
- Can solve by projection. Project the resulting factor back to the exponential family at each stage.
- Use “moment matching”
- This is the heart of expectation propagation.

Learning with samples

- Remember: Gibbs sampling for inference?
- But how do we do learning?
- Can just sample jointly from parameters and latent variables: learning as inference.
  - But that can be hard to get good mixing.
- Can we do gradient ascent?
  - Tough because gradient estimate is noisy (e.g. Contrastive Divergence). That effects some gradient method
- Use stochastic gradient ascent.

Stochastic Gradient Methods

- Take dataset and split it into minibatches.
- Now select a minibatch (sequentially or at random)
- Compute the gradient for the minibatch.
- Update the parameters.
- Move on to the next minibatch.
- Reduce the learning rate through time.
- Lots of details…
- Benefit – make parameter changes on minibatches not whole datasets. More steps, faster, but noisier learning.
- For large datasets, the minibatch may contain all the info you need to get the right gradient direction.

Summary of Course

- Probabilistic Models underpin Machine Learning.
- The fundamentals of probabilistic modelling are the same across all model types.
- Inference as Inference, Learning as Inference, Optimization as approximate inference.
- Fundamental decomposition of the model into structure, latent representation, composition (e.g. mixture, factor), distribution and parameterisation.
- Inference utilises these same structures for tractability and efficiency.
- Can’t think about the model without also thinking about how you are going to do inference and learning in that model. Intractable models can be super-exponentially hard to handle.