Bayesian Conditional Random Fields using Power EP

Tom Minka

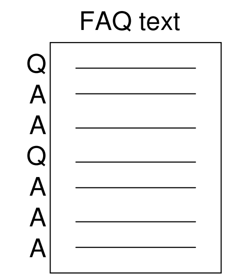
Joint work with Yuan Qi and Martin Szummer

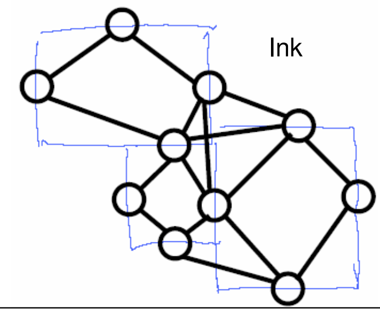
Why should you care?

- New way to train Conditional Random Fields
  - Significant improvement on small training sets
- Demonstration of Bayesian methods
- New computational scheme for Bayesian inference: Power EP
  - Benefits of Bayes at little computational cost

The task

- Want to label structured data:
  - Lines of text
  - Hyperlinked documents
  - Blocks of an image
  - Fragments of an ink diagram





Independent classification

- Classify each site independently, based on its features and those of its neighbors
- Problems:
  - Resulting labels may not make sense jointly
  - Requires lots of features (self + neighbors)
  - Performs redundant work in examining self + neighbors
- Want classifiers which are local but *linked*

Conditional Random Field (CRF)

- A linked set of classifiers
- Object x, possible labels t_i

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

$$Z(\mathbf{w}) = \sum_{\mathbf{t}} \prod_{\{i,j\}\in\mathcal{E}} g_{i,j}(t_i,t_j,\mathbf{x};\mathbf{w})$$

- g measures the three-way compatibility of the labels with the features *and* each other

$$g_{i,j}(t_i,t_j,\mathbf{x};\mathbf{w}) = \exp(\mathbf{w}_{t_i,t_j}^{\mathrm{T}}\phi_{i,j}(t_i,t_j,\mathbf{x}))$$

$$\mathbf{g}_{i,j}(t_i,t_j,\mathbf{x};\mathbf{w}) = (1-\epsilon)\Psi(\mathbf{w}_{t_i,t_j}^{\mathrm{T}}\phi_{i,j}(t_i,t_j,\mathbf{x})) + \epsilon(1-\Psi(\mathbf{w}_{t_i,t_j}^{\mathrm{T}}\phi_{i,j}(t_i,t_j,\mathbf{x})))$$

- w is parameter vector, E is linkage structure

Training CRFs

- Given labeled data $D = \{(\mathbf{x}^k, \mathbf{t}^k)\}$ we get a posterior

$$p(\mathbf{w}\,|\,D) \propto p(\mathbf{w})\prod_k p(\mathbf{t}^k\,|\,\mathbf{x}^k,\mathbf{w})$$

- Old way: assume w = most probable value
  - Easily overfits
- New (Bayesian) way: weight each possible w by posterior, average the results for *all* w's during testing
  - No overfitting (no fitting at all)
- Can this be done efficiently? Yes! (but approximately)
  - Use Power EP

Bayesian procedure

- Training: approximate the posterior of w

$$p(\mathbf{w}\,|\,D) \overset{\text{Power EP}}{\Rightarrow} q(\mathbf{w}) \sim Gaussian$$

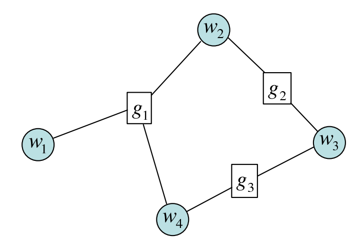
- Testing: approximate the posterior of t

$$p(\mathbf{t}\,|\,\mathbf{x},D) = \int p(\mathbf{t}\,|\,\mathbf{x},\mathbf{w})q(\mathbf{w})d\mathbf{w} \overset{\text{EP}}{\Rightarrow} q(\mathbf{t}) = \prod_i q(t_i)$$

Expectation Propagation (EP)

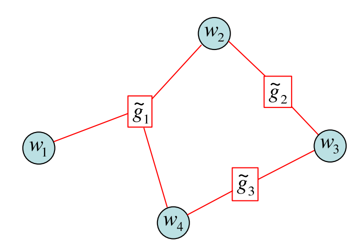
A method to approximate

$$p(\mathbf{w}) = \prod_a g_a(\mathbf{w})$$



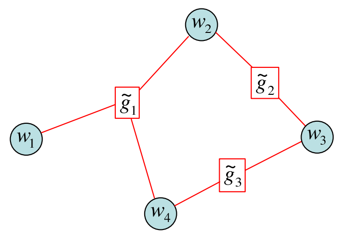
by

$$q(\mathbf{w}) = \prod_a \tilde{g}_a(\mathbf{w})$$



EP iteration

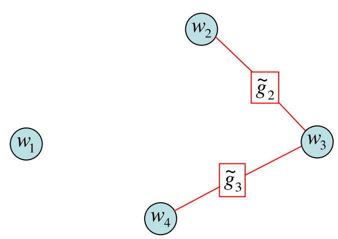
- Initial guess
- Delete
- Include
- Approximate



$$q(\mathbf{w}) = \tilde{g}_1(\mathbf{w})\ \tilde{g}_2(\mathbf{w})\ \tilde{g}_3(\mathbf{w})$$

EP iteration

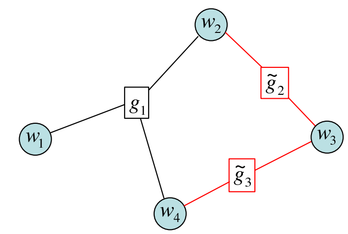
- Initial guess
- Delete
- Include
- Approximate



$$q(\mathbf{w}) = \quad \tilde{g}_2(\mathbf{w})\ \tilde{g}_3(\mathbf{w})$$

EP iteration

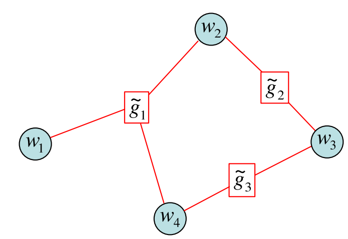
- Initial guess
- Delete
- Include
- Approximate



$$q(\mathbf{w}) = g_1(\mathbf{w})\ \tilde{g}_2(\mathbf{w})\ \tilde{g}_3(\mathbf{w})$$

EP iteration

- Initial guess
- Delete
- Include
- Approximate



$$q(\mathbf{w}) = \tilde{g}_1(\mathbf{w})\ \tilde{g}_2(\mathbf{w})\ \tilde{g}_3(\mathbf{w})$$

Approximating a factor

- Requires computing $< g_a(\mathbf{w}) >$ under

  $q^{\backslash a}(\mathbf{w}) = q(\mathbf{w}) / \tilde{g}_a(\mathbf{w})$
- Easy cases: $< e^w + 1 >$ $\quad < \psi(w) >$
- Hard cases: $< (e^w + 1)^{1.5} >$ $\quad \left\langle \frac{1}{e^w+1} \right\rangle$
- Variational methods require $< \log g_a(\mathbf{w}) >$
  - Minimizes a different error measure
  - "Exclusive" KL(q||p) vs. "Inclusive" KL(p||q)
  - Doesn't simplify the above cases

Power EP

- Instead of minimizing KL, minimize alpha-divergence:

$$D_\alpha(p \,||\, q) = \frac{4}{1-\alpha^2}\left(1 - \int_x p(x)^{(1+\alpha)/2} q(x)^{(1-\alpha)/2} dx\right)$$

- Only requires $< g_a^{(1+\alpha)/2} > \; = \; < g_a^{\beta} >$
- Choose beta to make integrals tractable:

$g_a = e^w + 1$	$\beta = 1$	$< e^w + 1 >$
$g_a = (e^w + 1)^{1.5}$	$\beta = \frac{1}{1.5}$	$< e^w + 1 >$
$g_a = \frac{1}{e_w + 1}$	$\beta = -1$	$< e^w + 1 >$

Power EP for CRFs

- Want to approximate

$$p(\mathbf{w}, D) = p(\mathbf{w}) \prod_k p(\mathbf{t}^k \mid \mathbf{x}^k, \mathbf{w}) \quad = p(\mathbf{w}) \prod_k \frac{1}{Z^k(\mathbf{w})} \prod_{(i,j)} g_{ij}^k(\mathbf{w})$$

- (prior)(partition fcn)(interactions)
- Process partition fcn using $\beta = -1$
- $< Z(\mathbf{w}) >$ is approximated by regular EP, where t is also a random variable

$$Z(\mathrm{w}) = \sum_{\mathbf{t}} \prod_{\{i,j\} \in \mathcal{E}} g_{i,j}(t_i, t_j, \mathrm{x}; \mathrm{w})$$

Algorithm structure

- Competing EP processes for numerator and denominator terms
- In numerator, t is known
- In denominator, t is inferred
- Helps to interleave the updates for each process, keeping them balanced
  - Otherwise may spiral out of control
- For testing, only one EP is required since denominator doesn't come into play
  - t is random in the numerator terms

Synthetic experiment

- Each object has 3 sites, fully linked, 24 random features per edge
- Labels generated from a random CRF
- Same model trained via MAP or Bayes

Algorithm	Test error		
	10 training objects	30 train.	100 train.
MAP	16%	12	10
Bayes	11	10	9

FAQ labeling

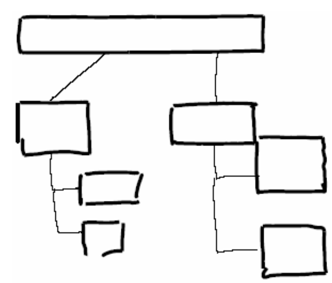
- Want to label each line of a FAQ list as being part of a question or an answer
- 19 training files, 500 lines on average, 24 binary features per line (contains question mark, indented, etc.)

- Lines linked in a chain
- MAP/Bayes used the same model and same priors

Algorithm	Test error
MAP	1.4%
Bayes	0.5

Ink labeling

- Want to label ink as part of container or connector
- 14 training diagrams
- Linkage by spatial proximity, probit-type interactions (except for MAP-Exp)



Algorithm	Test error
MAP	6.0%
MAP-Exp	5.2
Bayes	4.4

Conclusions

- Bayesian methods provide significant improvement on small training sets
- Using power EP, additional cost is minimal
- Power EP also provides model evidence, for hyperparameter selection (e.g. type of interactions)
- Other places reciprocals arise: multiclass BPMs, Markov Random Fields
  - Can use power EP to train them