Augment and Reduce: Stochastic Inference for Large Categorical Distributions

Francisco J. R. Ruiz

April 27th, 2018
Joint Work With
Categorical Distributions

- A probability distribution on a set of $K$ outcomes
- Normalized, $\sum_k p_k = 1$
- Ubiquitous in machine learning and many other disciplines
Our Contribution

- Goal: Speed up training for models with large categoricals ($K \gg 1$)
- Contribution: A fast algorithm with controlled complexity
- Key ideas: Variable augmentation, stochastic variational inference
Softmax

- One widely applied parameterization of a categorical,

\[ p(y = k | \psi) = \frac{e^{\psi_k}}{\sum_{k'} e^{\psi_{k'}}} \]

- Transforms reals into probabilities
A Case Study: Classification

- Observations are features and labels, \( \{x_n, y_n\}_{n=1}^N \)
- Each label \( y_n \in \{1, \ldots, K\} \)
A Case Study: Classification

- Observations are features and labels, \( \{x_n, y_n\}_{n=1}^N \)
- Each label \( y_n \in \{1, \ldots, K\} \)
- Each observation \( n \) is assigned a real value, \( \psi_k^{(n)} = w_k^\top x_n \)
- Goal: Find the weights \( w = (w_1, \ldots, w_K) \)
Maximize the likelihood of the data w.r.t. the weights,

\[
\text{find } w \text{ to maximize } \mathcal{L}_{\text{log-lik}} = \sum_n \log p(y_n | x_n, w)
\]
Maximize the likelihood of the data w.r.t. the weights,

\[ \text{find } w \text{ to maximize } \mathcal{L}_{\text{log-lik}} = \sum_n \log p(y_n | x_n, w) \]

Assuming the softmax transformation,

\[ \log p(y_n | x_n, w) = \log \left( \frac{e^{w_{y_n}^T x_n}}{\sum_{k'} e^{w_{k'}^T x_n}} \right) \]
A Case Study: Classification

- Optimization w.r.t. $w$
- Gradient ascent

The gradient is

$$\nabla_w \mathcal{L}_{\text{log-lik}} = \sum_n \nabla_w \log p(y_n | x_n, w)$$
A Case Study: Classification

\[ \nabla_w \log p(y_n \mid x_n, w) = \nabla_w \log \left( \frac{e^{w_{y_n}^\top x_n}}{\sum_{k'} e^{w_{k'}^\top x_n}} \right) \]

- **Problem:** Evaluating the gradient is $O(K)$
A Case Study: Classification

\[ \nabla_w \log p(y_n \mid x_n, w) = \nabla_w \log \left( \frac{e^{w_{y_n}^T x_n}}{\sum_k e^{w_k^T x_n}} \right) \]

- **Problem:** Evaluating the gradient is \( O(K) \)
- Evaluation is needed for each \( n = 1, \ldots, N \) and at each iteration of gradient ascent
A Case Study: Classification

\[ \nabla_w \log p(y_n \mid x_n, w) = \nabla_w \log \left( \frac{e^{w_{yn}^\top x_n}}{\sum_{k'} e^{w_{k'}^\top x_n}} \right) \]

- **Problem:** Evaluating the gradient is \( \mathcal{O}(K) \)
- Evaluation is needed for each \( n = 1, \ldots, N \) and at each iteration of gradient ascent

\[ w^{(0)} \rightarrow x \rightarrow w^{(1)} \rightarrow w^{(2)} \rightarrow w^{(3)} \]

- For large values of \( K \), this is prohibitive
Large Categoricals

- The $O(K)$ cost is not unique to the softmax
- Other models (multinomial probit/logistic) are also $O(K)$
Large Categoricals

- The $O(K)$ cost is not unique to the softmax
- Other models (multinomial probit/logistic) are also $O(K)$
- When $K$ is large, this is not OK
Large Categoricals

- The $\mathcal{O}(K)$ cost is not unique to the softmax

- Other models (multinomial probit/logistic) are also $\mathcal{O}(K)$

- When $K$ is large, this is not OK

- Examples: language models, recommendation systems, discrete choice models, reinforcement learning
Our Contribution

- An algorithm with reduced complexity, $\mathcal{O}(|S|)$ instead of $\mathcal{O}(K)$
- Complexity controlled by parameter $|S|$
Our Contribution

- An algorithm with reduced complexity, $O(|S|)$ instead of $O(K)$
- Complexity controlled by parameter $|S|$
- Two steps
  1. Augment the model with an auxiliary variable
  2. Reduce complexity via subsampling (stochastic optimization)
Let’s Take A Step Back...

- Where does the softmax come from?

\[ p(y = k \mid \psi) = \frac{e^{\psi_k}}{\sum_{k'} e^{\psi_{k'}}} \]
The Utility Perspective

- Draw random errors i.i.d., \( \varepsilon_k \sim \phi(\cdot) \)
The Utility Perspective

- Draw random errors i.i.d., $\varepsilon_k \sim \phi(\cdot)$
- Define a utility for each outcome $k$,

$$\psi_k + \varepsilon_k$$

(mean utility plus noise)
The Utility Perspective

- Draw random errors i.i.d., $\varepsilon_k \sim \phi(\cdot)$
- Define a utility for each outcome $k$,

$$\psi_k + \varepsilon_k$$

(mean utility plus noise)
- Choose the outcome with the largest utility,

$$y = \arg \max_k (\psi_k + \varepsilon_k)$$
The Utility Perspective

- Draw random errors i.i.d., $\varepsilon_k \sim \phi(\cdot)$
- Define a utility for each outcome $k$,
  \[ \psi_k + \varepsilon_k \]
  (mean utility plus noise)
- Choose the outcome with the largest utility,
  \[ y = \arg \max_k (\psi_k + \varepsilon_k) \]
- Integrate out the error terms ($\varepsilon_k$'s) to find the marginal $p(y | \psi)$
The Utility Perspective

- Different priors $\phi(\epsilon)$ lead to different categoricals

...
The Utility Perspective

- Different priors $\phi(\varepsilon)$ lead to different categoricals

- For $\phi(\varepsilon) = \text{Gumbel}(\varepsilon \mid 0, 1)$, we recover the softmax

$$p(y = k \mid \psi) = \frac{e^{\psi_k}}{\sum_{k'} e^{\psi_{k'}}}$$
The Utility Perspective

- Different priors $\phi(\varepsilon)$ lead to different categoricals.

- For $\phi(\varepsilon) = \text{Gumbel}(\varepsilon | 0, 1)$, we recover the softmax:

  $$p(y = k | \psi) = \frac{e^{\psi_k}}{\sum_{k'} e^{\psi_{k'}}}$$

- Other models: multinomial probit (Gaussian prior), multinomial logistic (logistic prior).
The Utility Perspective

- Augment the model with only one error term
- Work with the joint $p(y, \varepsilon | \psi)$
- Nice property: Amenable to stochastic optimization
The Utility Perspective

Augment the model with only one error term

Work with the joint $p(y, \epsilon | \psi)$

Nice property: Amenable to stochastic optimization
The Utility Perspective

- Augment the model with *only one* error term
The Utility Perspective

- Augment the model with *only one* error term
- Work with the joint $p(y, \varepsilon | \psi)$
- Nice property: Amenable to stochastic optimization
Let’s Do Some Maths

- The marginal likelihood is the probability that the realized utility \( \psi_k + \varepsilon_k \) is greater than the others,

\[
\begin{align*}
\Pr(y = k | \psi) &= \int_{-\infty}^{+\infty} \phi(\varepsilon_k) \left( \prod_{k' \neq k} \Phi(\varepsilon_k + \psi_k - \psi_{k'}) \right) d\varepsilon_k \\
\Phi(\cdot) &\text{ is the CDF of the distribution } \phi(\cdot) \\
\end{align*}
\]

- Augment the model,

\[
\Pr(y = k, \varepsilon | \psi) = \phi(\varepsilon) \left( \prod_{k' \neq k} \Phi(\varepsilon + \psi_k - \psi_{k'}) \right)
\]
Let’s Do Some Maths

- The marginal likelihood is the probability that the \textit{realized utility} \( \psi_k + \varepsilon_k \) is greater than the others,

\[
p(y = k \mid \psi) = \text{Prob} (\psi_k + \varepsilon_k \geq \psi_{k'} + \varepsilon_{k'} \quad \forall k' \neq k)
\]
Let’s Do Some Maths

▶ The marginal likelihood is the probability that the realized utility $\psi_k + \varepsilon_k$ is greater than the others,

$$p(y = k \mid \psi) = \text{Prob} (\psi_k + \varepsilon_k \geq \psi_{k'} + \varepsilon_{k'} \quad \forall k' \neq k)$$

▶ This is an integral,

$$p(y = k \mid \psi) = \int_{-\infty}^{+\infty} \phi(\varepsilon_k) \left( \prod_{k' \neq k} \int_{-\infty}^{\varepsilon_k + \psi_k - \psi_{k'}} \phi(\varepsilon_{k'}) d\varepsilon_{k'} \right) d\varepsilon_k$$
Let’s Do Some Maths

- The marginal likelihood is the probability that the realized utility $\psi_k + \epsilon_k$ is greater than the others,

\[
p(y = k \mid \psi) = \text{Prob} (\psi_k + \epsilon_k \geq \psi_{k'} + \epsilon_{k'} \ \forall k' \neq k)
\]

- This is an integral,

\[
p(y = k \mid \psi) = \int_{-\infty}^{+\infty} \phi(\epsilon_k) \left( \prod_{k' \neq k} \int_{-\infty}^{\epsilon_k + \psi_k - \psi_{k'}} \phi(\epsilon_{k'}) d\epsilon_{k'} \right) d\epsilon_k
\]

\[
= \int_{-\infty}^{+\infty} \phi(\epsilon) \left( \prod_{k' \neq k} \Phi(\epsilon + \psi_k - \psi_{k'}) \right) d\epsilon
\]

$\Phi(\cdot)$ is the CDF of the distribution $\phi(\cdot)$
Let’s Do Some Maths

- The marginal likelihood is the probability that the realized utility $\psi_k + \varepsilon_k$ is greater than the others,

\[ p(y = k \mid \psi) = \text{Prob}(\psi_k + \varepsilon_k \geq \psi_{k'} + \varepsilon_{k'} \quad \forall k' \neq k) \]

- This is an integral,

\[ p(y = k \mid \psi) = \int_{-\infty}^{+\infty} \phi(\varepsilon_k) \left( \prod_{k' \neq k} \int_{-\infty}^{\varepsilon_k + \psi_k - \psi_{k'}} \phi(\varepsilon_{k'}) d\varepsilon_{k'} \right) d\varepsilon_k \]

\[ = \int_{-\infty}^{+\infty} \phi(\varepsilon) \left( \prod_{k' \neq k} \Phi(\varepsilon + \psi_k - \psi_{k'}) \right) d\varepsilon \]

$\Phi(\cdot)$ is the CDF of the distribution $\phi(\cdot)$

- Augment the model,

\[ p(y = k, \varepsilon \mid \psi) = \phi(\varepsilon) \prod_{k' \neq k} \Phi(\varepsilon + \psi_k - \psi_{k'}) \]
The Augmented Model

We now have the augmented model,

\[ p(y = k, \varepsilon \mid \psi) = \phi(\varepsilon) \prod_{k' \neq k} \Phi(\varepsilon + \psi_k - \psi_{k'}) \]
The Augmented Model

- The augmented model,

\[ p(y = k, \varepsilon | \psi) = \phi(\varepsilon) \prod_{k' \neq k} \Phi(\varepsilon + \psi_k - \psi_{k'}) \]
The Augmented Model

- The augmented model,

\[ p(y = k, \varepsilon | \psi) = \phi(\varepsilon) \prod_{k' \neq k} \Phi(\varepsilon + \psi_k - \psi_{k'}) \]

- Nice property: The log-joint has a summation over \( k' \),

\[ \log p(y = k, \varepsilon | \psi) = \log \phi(\varepsilon) + \sum_{k' \neq k} \log \Phi(\varepsilon + \psi_k - \psi_{k'}) \]
The Augmented Model

The augmented model,

\[ p(y = k, \varepsilon \mid \psi) = \phi(\varepsilon) \prod_{k' \neq k} \Phi(\varepsilon + \psi_k - \psi_{k'}) \]

Nice property: The log-joint has a summation over \( k' \),

\[ \log p(y = k, \varepsilon \mid \psi) = \log \phi(\varepsilon) + \sum_{k' \neq k} \log \Phi(\varepsilon + \psi_k - \psi_{k'}) \]

This enables fast unbiased estimates,

1. Sample a subset of outcomes \( S \subseteq \{1, \ldots, K\} \setminus \{k\} \) of fixed size \( |S| \)
2. Compute an estimate of the log-joint in \( \mathcal{O}(|S|) \) complexity

\[ \log \phi(\varepsilon) + \frac{K - 1}{|S|} \sum_{k' \in S} \log \Phi(\varepsilon + \psi_k - \psi_{k'}) \]
Augment & Reduce: Variational EM

- We are not interested in the log-joint, but in the log-marginal
Augment & Reduce: Variational EM

- We are not interested in the log-joint, but in the log-marginal.

- Variational inference relates both quantities,

\[
\log p(y \mid \psi) \geq \mathbb{E}_{q(\varepsilon)} \left[ \log p(y, \varepsilon \mid \psi) - \log q(\varepsilon) \right]
\]
Augment & Reduce: Variational EM

- We are not interested in the log-joint, but in the log-marginal

- Variational inference relates both quantities,

\[
\log p(y \mid \psi) \geq \mathbb{E}_{q(\varepsilon)} \left[ \log p(y, \varepsilon \mid \psi) - \log q(\varepsilon) \right]
\]

- Maximize the bound using variational EM
  1. E step: Optimize w.r.t. the distribution \( q(\varepsilon) \)
  2. M step: Take a gradient step w.r.t. \( \psi \) (or its parameters \( w \))
Example: Augment & Reduce For Classification

- Recall the classification objective,

\[ \mathcal{L}_{\text{log-lik}} = \sum_n \log p(y_n | x_n, w) \]
Example: Augment & Reduce For Classification

Recall the classification objective,

\[ \mathcal{L}_{\text{log-lik}} = \sum_n \log p(y_n \mid x_n, \theta) \]

Replace each term with its variational bound,

\[ \mathcal{L}_{\text{bound}} = \sum_n \mathbb{E}_{q(\varepsilon^{(n)})} \left[ \log p(y_n, \varepsilon^{(n)} \mid x_n, \theta) - \log q(\varepsilon^{(n)}) \right] \]
Example: Augment & Reduce For Classification

▶ Recall the classification objective,

\[ \mathcal{L}_{\text{log-lik}} = \sum_{n} \log p(y_n \mid x_n, w) \]

▶ Replace each term with its variational bound,

\[ \mathcal{L}_{\text{bound}} = \sum_{n} \mathbb{E}_{q(\varepsilon^{(n)})} \left[ \log p(y_n, \varepsilon^{(n)} \mid x_n, w) - \log q(\varepsilon^{(n)}) \right] \]

▶ Algorithm

1. Subsample datapoints \( B \subseteq \{1, \ldots, N\} \)
2. For each \( n \in B \), subsample classes \( S \subseteq \{1, \ldots, K\} \setminus \{y_n\} \)
3. (E step) For each \( n \in B \), update its \( q(\varepsilon^{(n)}) \) \( \mathcal{O}(|S|) \)
4. (M step) For each \( n \in B \), compute gradient w.r.t. \( w \) \( \mathcal{O}(|S|) \)
5. (M step) Take gradient step for \( w \)
6. Repeat
Example: Augment & Reduce For Classification

- Recall the log-joint in the augmented model,

\[
\log p(y = k, \varepsilon \mid \psi) = \log \phi(\varepsilon) + \sum_{k' \neq k} \log \Phi(\varepsilon + \psi_k - \psi_{k'})
\]
Example: Augment & Reduce For Classification

- Recall the log-joint in the augmented model,

\[
\log p(y = k, \varepsilon | \psi) = \log \phi(\varepsilon) + \sum_{k' \neq k} \log \Phi(\varepsilon + \psi_k - \psi_{k'})
\]

- Consider the gradient of the bound in the M step,

\[
\nabla_w \mathcal{L}_{\text{bound}} = \nabla_w \sum_n \mathbb{E}_{q(\varepsilon^{(n)})} \left[ \log p(y_n, \varepsilon^{(n)} | x_n, w) - \log q(\varepsilon^{(n)}) \right]
\]
Example: Augment & Reduce For Classification

- Recall the log-joint in the augmented model,

\[
\log p(y = k, \varepsilon \mid \psi) = \log \phi(\varepsilon) + \sum_{k' \neq k} \log \Phi(\varepsilon + \psi_k - \psi_{k'})
\]

- Consider the gradient of the bound in the M step,

\[
\nabla_w \mathcal{L}_{\text{bound}} = \nabla_w \sum_n \mathbb{E}_{q(\varepsilon^{(n)})} \left[ \log p(y_n, \varepsilon^{(n)} \mid x_n, w) - \log q(\varepsilon^{(n)}) \right]
\]

\[
= \sum_n \sum_{k' \neq y_n} \mathbb{E}_{q(\varepsilon^{(n)})} \left[ \nabla_w \log \Phi(\varepsilon^{(n)} + w_{y_n}^T x_n - w_{k'}^T x_n) \right]
\]
Example: Augment & Reduce For Classification

- Recall the log-joint in the augmented model,

\[
\log p(y = k, \varepsilon | \psi) = \log \phi(\varepsilon) + \sum_{k' \neq k} \log \Phi(\varepsilon + \psi_k - \psi_{k'})
\]

- Consider the gradient of the bound in the M step,

\[
\nabla_w \mathcal{L}_{\text{bound}} = \nabla_w \sum_n \mathbb{E}_{q(\varepsilon^{(n)})} \left[ \log p(y_n, \varepsilon^{(n)} | x_n, w) - \log q(\varepsilon^{(n)}) \right]
\]

\[
= \sum_n \sum_{k' \neq y_n} \mathbb{E}_{q(\varepsilon^{(n)})} \left[ \nabla_w \log \Phi(\varepsilon^{(n)} + w_{y_n}^T x_n - w_{k'}^T x_n) \right]
\]

\[
\approx \frac{N}{|B|} \frac{K - 1}{|S|} \sum_{n \in B} \sum_{k' \in S_n} \mathbb{E}_{q(\varepsilon^{(n)})} \left[ \nabla_w \log \Phi(\varepsilon^{(n)} + w_{y_n}^T x_n - w_{k'}^T x_n) \right]
\]
Example: Augment & Reduce For Classification

Recall the log-joint in the augmented model,

\[
\log p(y = k, \varepsilon \mid \psi) = \log \phi(\varepsilon) + \sum_{k' \neq k} \log \Phi(\varepsilon + \psi_k - \psi_{k'})
\]

Consider the gradient of the bound in the M step,

\[
\nabla_w \mathcal{L}_{\text{bound}} = \nabla_w \sum_n \mathbb{E}_{q(\varepsilon^{(n)})} \left[ \log p(y_n, \varepsilon^{(n)} \mid x_n, w) - \log q(\varepsilon^{(n)}) \right]
\]

\[
= \sum_n \sum_{k' \neq y_n} \mathbb{E}_{q(\varepsilon^{(n)})} \left[ \nabla_w \log \Phi(\varepsilon^{(n)} + w_{y_n}^T x_n - w_{k'}^T x_n) \right]
\]

\[
\approx \frac{N}{|\mathcal{B}|} \frac{K - 1}{|\mathcal{S}|} \sum_{n \in \mathcal{B}} \sum_{k' \in \mathcal{S}_n} \mathbb{E}_{q(\varepsilon^{(n)})} \left[ \nabla_w \log \Phi(\varepsilon^{(n)} + w_{y_n}^T x_n - w_{k'}^T x_n) \right]
\]
Augment & Reduce For Softmax Model

- The softmax model is special
  - We can compute the optimal $q(\varepsilon)$ distribution
  - We can compute the integrals
Augment & Reduce For Softmax Model

- The softmax model is special
  - We can compute the optimal $q(\varepsilon)$ distribution
  - We can compute the integrals
- The optimal variational distribution is Gumbel,

\[
q^*(\varepsilon) = \text{Gumbel}(\log \eta^*, 1), \quad \eta^* = 1 + \sum_{k' \neq k} e^{\psi_{k'} - \psi_k}
\]
Augment & Reduce For Softmax Model

- The softmax model is special
  - We can compute the optimal $q(\varepsilon)$ distribution
  - We can compute the integrals
- The optimal variational distribution is Gumbel,

$$q^*(\varepsilon) = \text{Gumbel}(\log \eta^*, 1), \quad \eta^* = 1 + \sum_{k' \neq k} e^{\psi_{k'} - \psi_k}$$

- Instead, set

$$q^*(\varepsilon) = \text{Gumbel}(\log \eta, 1)$$
Augment & Reduce For Softmax Model

- The softmax model is special
  - We can compute the optimal $q(\varepsilon)$ distribution
  - We can compute the integrals
- The optimal variational distribution is Gumbel,

$$q^*(\varepsilon) = \text{Gumbel}(\log \eta^*, 1), \quad \eta^* = 1 + \sum_{k' \neq k} e^{\psi_{k'} - \psi_k}$$

- Instead, set

$$q^*(\varepsilon) = \text{Gumbel}(\log \eta, 1)$$

- Estimate the optimal natural parameter,

$$\tilde{\eta} = 1 + \frac{K - 1}{|S|} \sum_{k' \in S} e^{\psi_{k'} - \psi_k}$$

(to update $\eta$, take a step in the direction of the natural gradient)
Augment & Reduce For Other Models

- For other models, the expectations are intractable
Augment & Reduce For Other Models

- For other models, the expectations are intractable
- We form Monte Carlo gradient estimators using the reparameterization trick
Augment & Reduce For Other Models

- For other models, the expectations are intractable
- We form Monte Carlo gradient estimators using the reparameterization trick
- Useful for both E and M steps
Experiments

- Experiments: Linear classification

- Maximum likelihood estimation

- 5 datasets

<table>
<thead>
<tr>
<th>dataset</th>
<th>$N_{train}$</th>
<th>$N_{test}$</th>
<th>covariates</th>
<th>classes</th>
<th>minibatch (obs.)</th>
<th>minibatch (classes)</th>
<th>iterations</th>
</tr>
</thead>
<tbody>
<tr>
<td>MNIST</td>
<td>60,000</td>
<td>10,000</td>
<td>784</td>
<td>10</td>
<td>500</td>
<td>1</td>
<td>35,000</td>
</tr>
<tr>
<td>Bibtex</td>
<td>4,880</td>
<td>2,413</td>
<td>1,836</td>
<td>148</td>
<td>488</td>
<td>20</td>
<td>5,000</td>
</tr>
<tr>
<td>Omniglot</td>
<td>25,968</td>
<td>6,492</td>
<td>784</td>
<td>1,623</td>
<td>541</td>
<td>50</td>
<td>45,000</td>
</tr>
<tr>
<td>EURLex-4K</td>
<td>15,539</td>
<td>3,809</td>
<td>5,000</td>
<td>896</td>
<td>279</td>
<td>50</td>
<td>100,000</td>
</tr>
<tr>
<td>AmazonCat-13K</td>
<td>1,186,239</td>
<td>306,782</td>
<td>203,882</td>
<td>2,919</td>
<td>1,987</td>
<td>60</td>
<td>5,970</td>
</tr>
</tbody>
</table>
Experiments

- Comparisons against:
  - Exact softmax (only for MNIST and Bibtex)
Experiments

- Comparisons against:
  - Exact softmax (only for MNIST and Bibtex)
  - One-vs-each (OVE) bound,

\[
\mathcal{L}_{OVE} = \sum_{k' \neq k} \log \sigma(\psi_k - \psi_{k'})
\]

(it is a bound on the softmax)
Experiments

- Time complexity

<table>
<thead>
<tr>
<th>dataset</th>
<th>OVE (Titsias, 2016)</th>
<th>softmax</th>
<th>multi. probit</th>
<th>multi. logistic</th>
</tr>
</thead>
<tbody>
<tr>
<td>MNIST</td>
<td>0.336 s</td>
<td>0.337 s</td>
<td>0.431 s</td>
<td>0.511 s</td>
</tr>
<tr>
<td>Bibtex</td>
<td>0.181 s</td>
<td>0.188 s</td>
<td>0.244 s</td>
<td>0.246 s</td>
</tr>
<tr>
<td>Omniglot</td>
<td>4.47 s</td>
<td>4.65 s</td>
<td>5.63 s</td>
<td>5.57 s</td>
</tr>
<tr>
<td>EURLex-4K</td>
<td>5.54 s</td>
<td>5.65 s</td>
<td>6.46 s</td>
<td>6.23 s</td>
</tr>
<tr>
<td>AmazonCat-13K</td>
<td>2.80 h</td>
<td>2.80 h</td>
<td>2.82 h</td>
<td>2.91 h</td>
</tr>
</tbody>
</table>
Experiments

- Quality of the bound
Experiments

- **Quality of the classification weights $w_k$ (predictive performance)**

<table>
<thead>
<tr>
<th>dataset</th>
<th>exact</th>
<th>OVE (Titsias, 2016)</th>
<th>A&amp;R [this paper]</th>
<th>multi. probit A&amp;R [this paper]</th>
<th>multi. logistic A&amp;R [this paper]</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>log lik</td>
<td>log lik</td>
<td>log lik</td>
<td>log lik</td>
<td>log lik</td>
</tr>
<tr>
<td></td>
<td>acc</td>
<td>acc</td>
<td>acc</td>
<td>acc</td>
<td>acc</td>
</tr>
<tr>
<td>MNIST</td>
<td>−0.261</td>
<td>−0.276</td>
<td>−0.271</td>
<td>−0.302</td>
<td>−0.287</td>
</tr>
<tr>
<td>Bibtex</td>
<td>0.927</td>
<td>0.919</td>
<td>0.924</td>
<td>0.918</td>
<td>0.917</td>
</tr>
<tr>
<td>Omniglot</td>
<td>−3.188</td>
<td>−3.300</td>
<td>−3.036</td>
<td>−4.184</td>
<td>−3.151</td>
</tr>
<tr>
<td>EURLex-4K</td>
<td>0.361</td>
<td>0.352</td>
<td>0.361</td>
<td>0.346</td>
<td>0.353</td>
</tr>
<tr>
<td>AmazonCat-13K</td>
<td>−</td>
<td>−5.667</td>
<td>−5.171</td>
<td>−7.350</td>
<td>−5.395</td>
</tr>
<tr>
<td></td>
<td>−</td>
<td>0.179</td>
<td>0.201</td>
<td>0.178</td>
<td>0.184</td>
</tr>
<tr>
<td></td>
<td>−</td>
<td>−4.241</td>
<td>−4.593</td>
<td>−4.193</td>
<td>−4.299</td>
</tr>
<tr>
<td></td>
<td>−</td>
<td>0.247</td>
<td>0.207</td>
<td>0.263</td>
<td>0.226</td>
</tr>
<tr>
<td></td>
<td>−</td>
<td>−3.880</td>
<td>−3.795</td>
<td>−3.593</td>
<td>−4.081</td>
</tr>
<tr>
<td></td>
<td>−</td>
<td>0.388</td>
<td>0.420</td>
<td>0.411</td>
<td>0.350</td>
</tr>
</tbody>
</table>
Conclusion

- A method to scale up training for models involving large categorical distributions
- Stochastic variational EM
- Controlled complexity ($|S|$ is a parameter)
- Can be embedded in many different models
- Not limited to maximum likelihood estimation
Thank you for your attention!