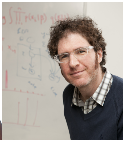
Augment and Reduce: Stochastic Inference for Large Categorical Distributions

Francisco J. R. Ruiz

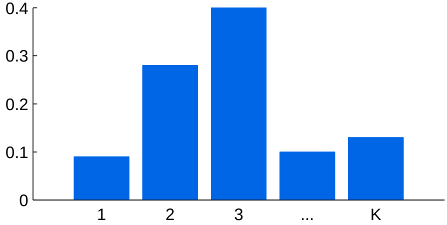
April 27th, 2018

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

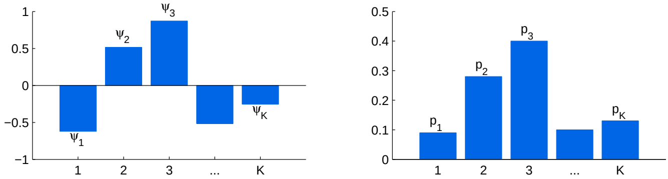
- Goal: Speed up training for models with large categoricals ($K \gg 1$)
- Contribution: A fast algorithm with controlled complexity
- Key ideas: Variable augmentation, stochastic variational inference

Softmax

- One widely applied parameterization of a categorical,

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

- 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 $w = (w_1, \ldots, w_K)$

A Case Study: Classification

- Maximize the likelihood of the data w.r.t. the weights,

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

A Case Study: Classification

- Maximize the likelihood of the data w.r.t. the weights,

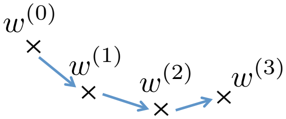
$$\text{find } w \text{ to maximize } \mathcal{L}_{\text{log-lik}} = \sum_n \log p(y_n \,|\, x_n, w)$$

- Assuming the softmax transformation,

$$\log p(y_n \,|\, x_n, w) = \log \left(\frac{e^{w_{y_n}^\top x_n}}{\sum_{k'} e^{w_{k'}^\top x_n}} \right)$$

A Case Study: Classification

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



- The gradient is

$$\nabla_w \mathcal{L}_{\text{log-lik}} = \sum_n \nabla_w \log p(y_n \mid x_n, w)$$

A Case Study: Classification

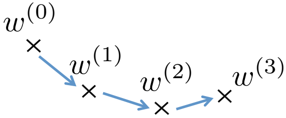
$$\nabla_w \log p(y_n \,|\, x_n, w) = \nabla_w \log \left(\frac{e^{w_{y_n}^\top x_n}}{\sum_{k'} e^{w_{k'}^\top x_n}} \right)$$

- **Problem:** Evaluating the gradient is $\mathcal{O}(K)$

A Case Study: Classification

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

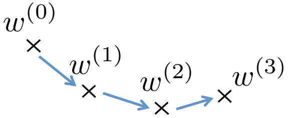
- **Problem:** Evaluating the gradient is $\mathcal{O}(K)$
- Evaluation is needed for each $n = 1, \ldots, N$ and at each iteration of gradient ascent



A Case Study: Classification

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

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



- For large values of K, this is prohibitive

Large Categoricals

- The $\mathcal{O}(K)$ cost is not unique to the softmax
- Other models (multinomial probit/logistic) are also $\mathcal{O}(K)$

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

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, $\mathcal{O}(|\mathcal{S}|)$ instead of $\mathcal{O}(K)$
- Complexity controlled by parameter $|\mathcal{S}|$

Our Contribution

- An algorithm with reduced complexity, $\mathcal{O}(|\mathcal{S}|)$ instead of $\mathcal{O}(K)$
- Complexity controlled by parameter $|\mathcal{S}|$
- 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 (ε_k's) to find the marginal $p(y \mid \psi)$

The Utility Perspective

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

The Utility Perspective

- Different priors $\phi(\varepsilon)$ lead to different categoricals
- For $\phi(\varepsilon) = \mathrm{Gumbel}(\varepsilon \mid 0, 1)$, we recover the softmax

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

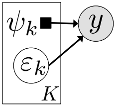
The Utility Perspective

- Different priors $\phi(\varepsilon)$ lead to different categoricals
- For $\phi(\varepsilon) = \mathrm{Gumbel}(\varepsilon \mid 0, 1)$, we recover the softmax

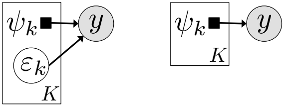
$$p(y = k \mid \psi) = \frac{e^{\psi_k}}{\sum_{k'} e^{\psi_{k'}}}$$

- Other models: multinomial probit (Gaussian prior), multinomial logistic (logistic prior)

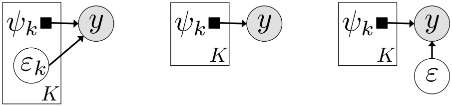
The Utility Perspective



The Utility Perspective

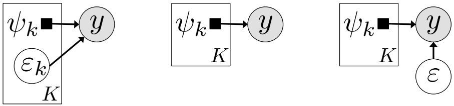


The Utility Perspective



- Augment the model with *only one* error term

The Utility Perspective



- Augment the model with *only one* error term
- Work with the joint $p(y, \varepsilon \mid \psi)$
- 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,

Let's Do Some Maths

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

$$p(y = k \mid \psi) = \mathrm{Prob}\left(\psi_k + \varepsilon_k \geq \psi_{k'} + \varepsilon_{k'} \;\; \forall k' \neq k\right)$$

Let's Do Some Maths

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

$$p(y = k \mid \psi) = \mathrm{Prob}\left(\psi_k + \varepsilon_k \geq \psi_{k'} + \varepsilon_{k'} \;\; \forall k' \neq k\right)$$

- 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* $\psi_k + \varepsilon_k$ is greater than the others,

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

- 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 $\phi(\cdot)$

Let's Do Some Maths

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

$$p(y = k \mid \psi) = \mathrm{Prob}\left(\psi_k + \varepsilon_k \geq \psi_{k'} + \varepsilon_{k'} \;\; \forall k' \neq k\right)$$

- This is an integral,

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

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

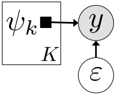
  $\Phi(\cdot)$ is the CDF of the distribution $\phi(\cdot)$
- Augment the model,

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

The Augmented Model

- We now have the augmented model,

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



The Augmented Model

- The augmented model,

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

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',

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

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

$$\log \phi(\varepsilon) + \frac{K-1}{|\mathcal{S}|} \sum_{k' \in \mathcal{S}} \log \Phi(\varepsilon + \psi_k - \psi_{k'})$$

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. ψ (or its parameters w)

Example: Augment & Reduce For Classification

- Recall the classification objective,

$$\mathcal{L}_{\text{log-lik}} = \sum_n \log p(y_n \mid x_n, w)$$

Example: Augment & Reduce For Classification

- Recall the classification objective,

$$\mathcal{L}_{\text{log-lik}} = \sum_n \log p(y_n \mid x_n, w)$$

- Replace each term with its variational bound,

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

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,

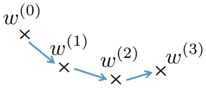
$$\mathcal{L}_{\text{bound}} = \sum_n \mathbb{E}_{q(\varepsilon^{(n)})}\left[\log p(y_n, \varepsilon^{(n)} \mid x_n, w) - \log q(\varepsilon^{(n)})\right]$$

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

Example: Augment & Reduce For Classification

- Recall the log-joint in the augmented model,

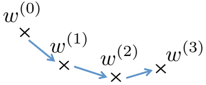
$$\log p(y = k, \varepsilon \,|\, \psi) = \log \phi(\varepsilon) + \sum_{k' \neq k} \log \Phi(\varepsilon + \psi_k - \psi_{k'})$$



Example: Augment & Reduce For Classification

- Recall the log-joint in the augmented model,

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



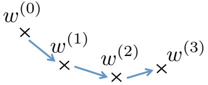
- Consider the gradient of the bound in the M step,

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

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,

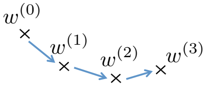
$$\nabla_w \mathcal{L}_{\text{bound}} = \nabla_w \sum_n \mathbb{E}_{q(\varepsilon^{(n)})} \left[\log p(y_n, \varepsilon^{(n)} \mid x_n, w) - \log q(\varepsilon^{(n)}) \right]$$

$$= \sum_n \sum_{k' \neq y_n} \mathbb{E}_{q(\varepsilon^{(n)})} \left[\nabla_w \log \Phi(\varepsilon^{(n)} + w_{y_n}^\top x_n - w_{k'}^\top x_n) \right]$$

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,

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

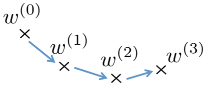
$$= \sum_n \sum_{k' \neq y_n} \mathbb{E}_{q(\varepsilon^{(n)})} \left[\nabla_w \log \Phi(\varepsilon^{(n)} + w_{y_n}^\top x_n - w_{k'}^\top x_n) \right]$$

$$\approx \frac{N}{|\mathcal{B}|} \frac{K-1}{|\mathcal{S}|} \sum_{n \in \mathcal{B}} \sum_{k' \in \mathcal{S}_n} \mathbb{E}_{q(\varepsilon^{(n)})} \left[\nabla_w \log \Phi(\varepsilon^{(n)} + w_{y_n}^\top x_n - w_{k'}^\top x_n) \right]$$

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,

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

$$= \sum_n \sum_{k' \neq y_n} \mathbb{E}_{q(\varepsilon^{(n)})} \left[\nabla_w \log \Phi(\varepsilon^{(n)} + w_{y_n}^\top x_n - w_{k'}^\top x_n) \right]$$

$$\approx \frac{N}{|\mathcal{B}|} \frac{K-1}{|\mathcal{S}|} \sum_{n \in \mathcal{B}} \sum_{k' \in \mathcal{S}_n} \mathbb{E}_{q(\varepsilon^{(n)})} \left[\nabla_w \log \Phi(\varepsilon^{(n)} + w_{y_n}^\top x_n - w_{k'}^\top x_n) \right]$$

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,

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

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,

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

- Instead, set

$$q^\star(\varepsilon) = \mathrm{Gumbel}(\log \eta, 1)$$

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,

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

- Instead, set

$$q^\star(\varepsilon) = \mathrm{Gumbel}(\log \eta, 1)$$

- Estimate the optimal natural parameter,

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

(to update η, 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

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

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,

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

    (it is a bound on the softmax)

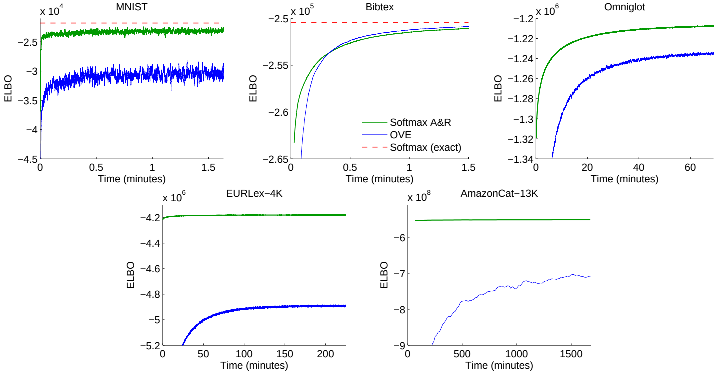
Experiments

- Time complexity

dataset	OVE (Titsias, 2016)	A&R [this paper] softmax	multi. probit	multi. logistic
MNIST	0.336 s	0.337 s	0.431 s	0.511 s
Bibtex	0.181 s	0.188 s	0.244 s	0.246 s
Omniglot	4.47 s	4.65 s	5.63 s	5.57 s
EURLex-4K	5.54 s	5.65 s	6.46 s	6.23 s
AmazonCat-13K	2.80 h	2.80 h	2.82 h	2.91 h

Experiments

- Quality of the bound



Experiments

- Quality of the classification weights w_k (predictive performance)

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

Conclusion

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

Thank you for your attention!