Latent variable session based recommendation

Motivating Example

criteo

Motivating Example

Product view

Recommend

criteo

Motivating Example

Product view

Recommend

criteo

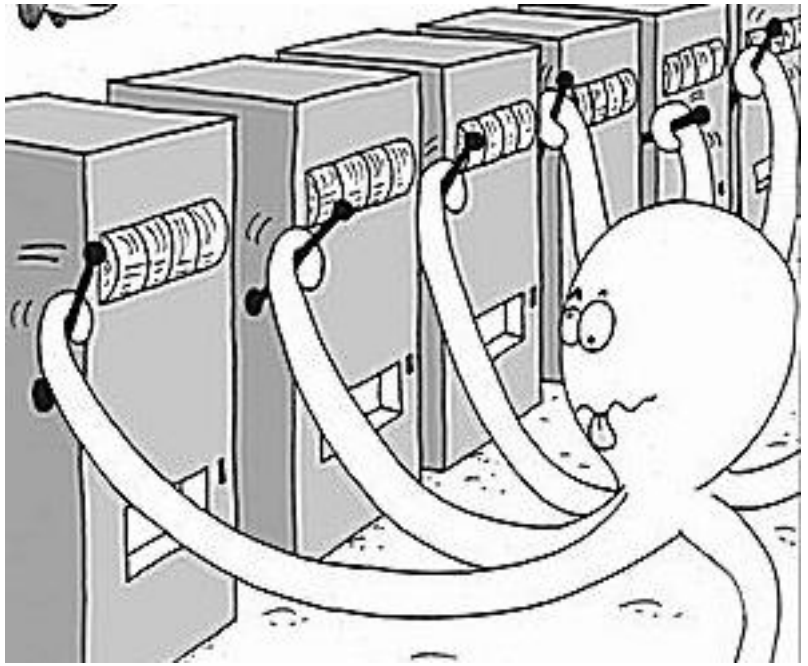
History: Split History of Recommender Systems

Next Item or Missing Link Prediction

e.g. Netflix Prize or Movie Lens

	movie 1	movie 2	movie 3	movie 4	movie 5	movie 6	movie 7	movie 8	movie 9	movie 10
user 1			1		2					
user 2		2		3	3			4		
user 3							5	3		4
user 4	2				3			2		
user 5		4				5			3	
user 6			2							
user 7			2					4	2	3
user 8	3	4				4				
user 9									3	
user 10			1		2					

Computational Advertising, e.g. Bandits, Conterfactual Risk Minimisation, Contextual Bandits, Reinforcement Learning.



criteo

Motivating Example

Product view

Recommend

criteo

Motivating Example

Product view

Recommend

criteo

Motivating Example

Product view

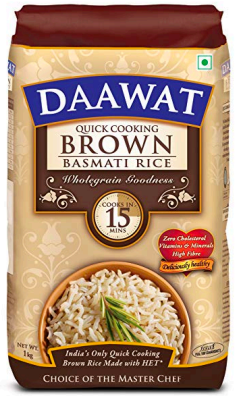
Recommend

criteo

Motivating Example

Product view

Recommend



criteo

Motivating Example

Product view

Recommend

criteo

The Notation

Symbol	Dimension	Description
u	Scalar	A given users id.
t	Scalar	sequential time.
P	Scalar	Total number of products.
K	Scalar	The size of the embedding.
$v_{u,t}$	Scalar	Product id for user u at time t.
$\boldsymbol{\omega}_u$	$K \times 1$	A given users state.
$\boldsymbol{\Psi}$	$P \times K$	Product embedding matrix.
$\boldsymbol{\Psi}_v$	$1 \times K$	Product embedding for v.
$\boldsymbol{\mu}_q$	$K \times 1$	The mean of $\boldsymbol{\omega}_u$.
$\boldsymbol{\Sigma}_q$	$K \times K$	The covariance of $\boldsymbol{\omega}_u$.
$\boldsymbol{\rho}$	$P \times 1$	Item popularity shift.
T_u	Scalar	Session length for u.

Table 1: Notations and Definitions

The Model

$$\omega_u \sim \mathcal{N}(0_K, I_K)$$

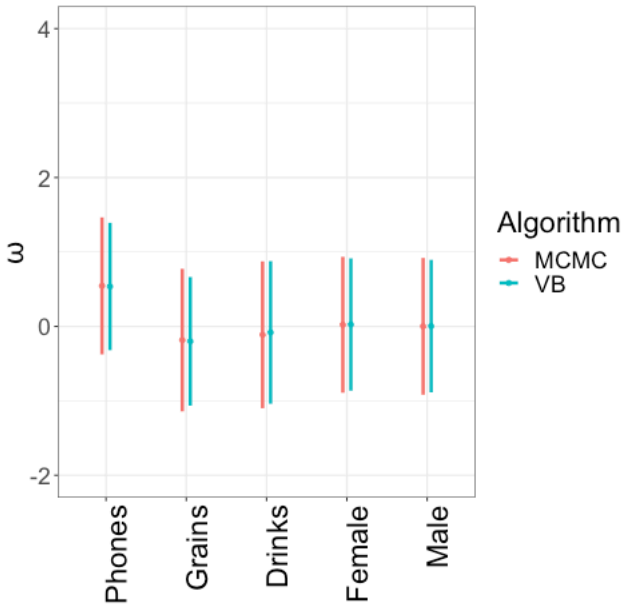
$$v_{u,1}, .., v_{u,T_u} \sim \text{categorical}(\text{softmax}(\Psi\omega_u + \rho))$$

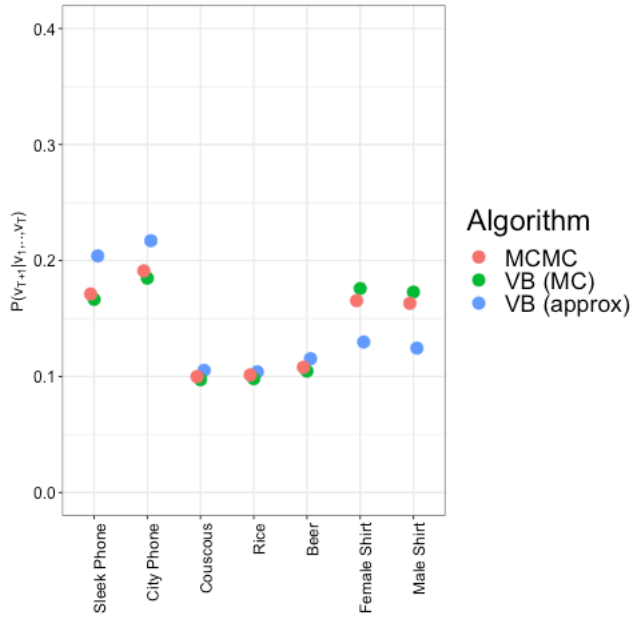
An Example: Imagine (pre) trained embeddings

$$\Psi = \begin{bmatrix} \Psi_{\text{Sleek Phone}} \\ \Psi_{\text{City Phone}} \\ \Psi_{\text{Couscous}} \\ \Psi_{\text{Rice}} \\ \Psi_{\text{Beer}} \\ \Psi_{\text{Female Shirt}} \\ \Psi_{\text{Male Shirt}} \end{bmatrix} = \begin{bmatrix} .9 & 0.05 & 0 & 0.05 & 0 \\ 1 & 0 & 0 & 0 & 0 \\ 0 & .95 & 0 & 0.1 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 0.2 & .7 & 0 & 0 \\ 0 & 0 & 0 & 1 & -1 \\ 0 & 0 & 0 & -1 & 1 \end{bmatrix}$$

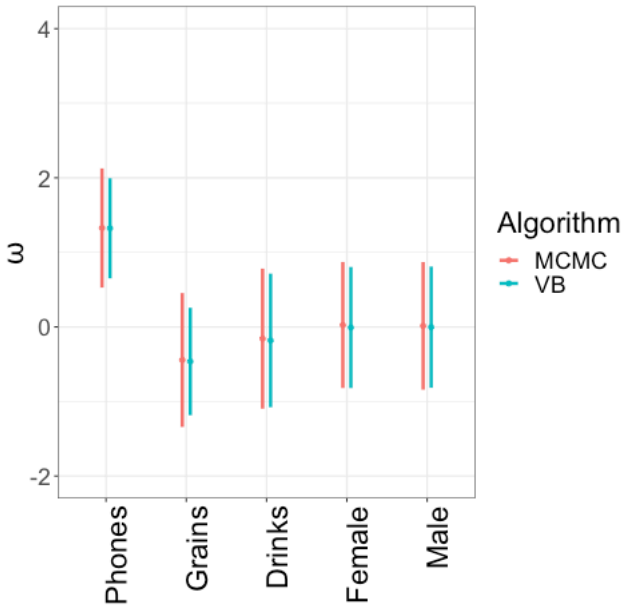
$$\rho = 0$$

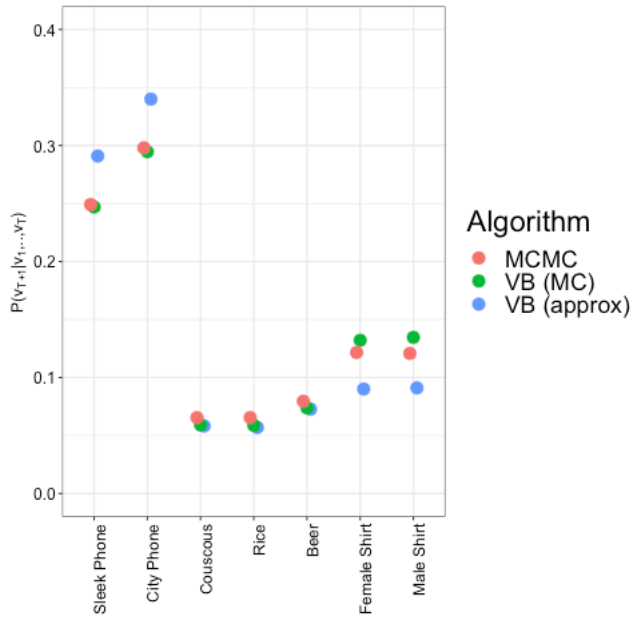
An Example: One Sleek Phone



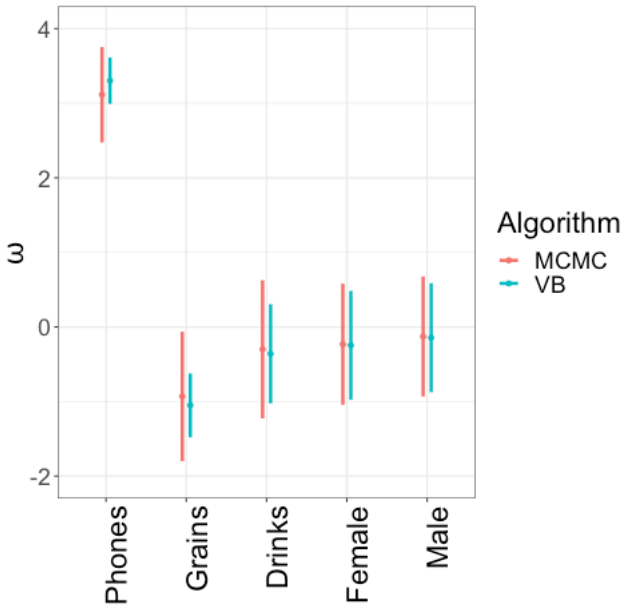


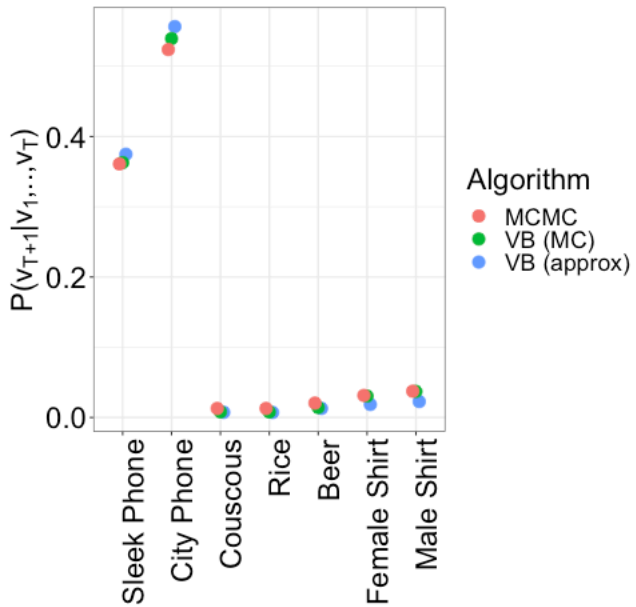
An Example: One Sleek Phone, two City Phones





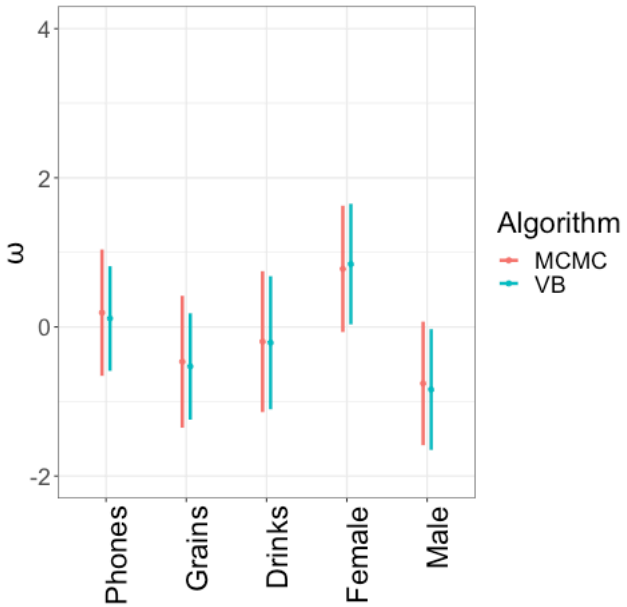
An Example: One Sleek Phone, twenty City phones

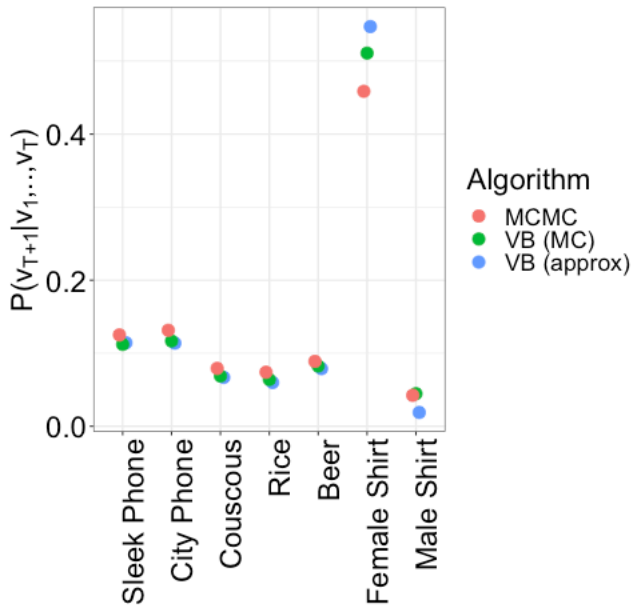




criteo

An Example: Two Female Shirts, one Sleek Phone





criteo

Three Computational Tasks

1. Given the embeddings produce (quickly) an approximate posterior over: ω

2. Learn the embeddings Ψ, ρ (integrating over the user embedding ω)

3. Next item prediction (this one isn't a real problem actually)

$$p(v_{u,T+1}|v_{u,1},..,v_{u,T})$$
$$= \int p(v_{u,T+1}|\omega,\Psi,\rho)p(\omega|v_{u,1},.v_{u,T})d\omega_u$$

Complete Data Likelihood

$$\log p(v_1, .., v_T, \boldsymbol{\omega}_u | \boldsymbol{\Psi}) = \left(\sum_t^T \boldsymbol{\Psi}_{v_t} \boldsymbol{\omega}_u + \boldsymbol{\rho}_{v_t} \right)$$

$$- T \log \{ \sum_p^P \exp(\boldsymbol{\Psi}_p \boldsymbol{\omega}_u + \boldsymbol{\rho}_p) \} - \frac{K}{2} \log(2\pi) - \frac{1}{2} \boldsymbol{\omega}_u^T \boldsymbol{\omega}_u$$

Variational Bound: Mean Field Approximation with Gaussian

$$\boldsymbol{\omega} \sim \mathcal{N}(\boldsymbol{\mu}_q, \Sigma_q),$$

$$\mathcal{L} = \mathop{\mathbb{E}}_{q(\boldsymbol{\omega})}[\log\, p(v_1, .., v_T, \boldsymbol{\omega}_u | \Psi) - \log q(\boldsymbol{\omega})] =$$

$$\left(\sum_t^T \Psi_{v_t} \boldsymbol{\mu}_q + \boldsymbol{\rho}_{v_t}\right) - T \mathop{\mathbb{E}}_{q(\boldsymbol{\omega})}[\log\{\sum_p^P \exp(\Psi_p \boldsymbol{\omega}_u + \boldsymbol{\rho}_p)\}]$$

$$-\frac{K}{2}\log(2\pi) - \frac{1}{2}\{\boldsymbol{\mu}_q^T \boldsymbol{\mu}_q + \text{trace}(\Sigma_q)\} + \frac{1}{2}\log|2\pi e \Sigma_q|$$

Dealing with expectations of the log sum exp

$$\mathop{\mathbb{E}}_{q(\omega)}[\log\{\sum_{p}^{P}\exp(\Psi_p\omega_u+\rho_p)\}]$$

Dealing with expectations of the log sum exp – Reparameterization Trick

$$\epsilon^{(s)} \sim \mathcal{N}(0_K, I_K)$$

$$\omega^{(s)} = L_{\Sigma_q}\epsilon^{(s)} + \mu_q$$

$$L_{\Sigma_q}L_{\Sigma_q}^T = \Sigma_q$$

$$\mathcal{L}_{MC} =$$

$$\left(\sum_t^T \Psi_{v_t}\mu_q + \rho_{v_t}\right) - T\log[\sum_p^P \exp\{\Psi_p(L_{\Sigma_q}\epsilon^{(s)} + \mu_q) + \rho_p\}]$$

$$-\frac{K}{2}\log(2\pi) - \frac{1}{2}\{\mu_q^T\mu_q + \mathrm{trace}(\Sigma_q)\} + \frac{1}{2}\log|2\pi e\Sigma_q|.$$

criteo

Dealing with expectations of the log sum exp – Bouchard Bound

For any: $a,\ \xi$:

$$\mathcal{L} \geq \mathcal{L}_{\text{Bouch}} = \left(\sum_t^T \Psi_{v_t} \mu_q + \rho_{v_t} \right)$$

$$- T[a + \sum_p^P \frac{\Psi_p \mu_q + \rho_p - a - \xi_p}{2}$$

$$+ \lambda_{\text{JJ}}(\xi_p)\{(\Psi_p \mu_q + \rho_p - a)^2 + \Psi_p \Sigma_q \Psi_p^T - \xi_p^2\} + \log(1 + e^{\xi_p})]$$

$$- \frac{K}{2} \log(2\pi) - \frac{1}{2}\{\mu_q^T \mu_q + \text{trace}(\Sigma_q)\} + \frac{1}{2} \log |2\pi e \Sigma_q|.$$

Where $\lambda_{\text{JJ}}(\cdot)$ is the Jaakola and Jordan function [13]:

$$\lambda_{\text{JJ}}(\xi) = \frac{1}{2\xi} \left(\frac{1}{1 + e^{-\xi}} - \frac{1}{2} \right).$$

Latent variable size (and variational parameters) grow with number of users

$$\mu_q, \Sigma_q$$

$$\mu_q, \Sigma_q, a, \xi$$

$$\mu_q, \Sigma_q = f_\Xi(v_1, \ldots v_T),$$

$$\mu_q, \Sigma_q, a, \xi = f_\Xi^{\text{Bouch}}(v_1, \ldots v_T)$$

Variational Bayes Algorithm for Organic Data

$$\Sigma_q^{-1} = I_k + 2\sum_p JJ(\xi_p)\Psi_p^T\Psi_p$$

$$\mu_q = \Sigma_q^{-1}\left(\left(\sum_t^T \Psi_{v_t}^T\right) - T\left(\sum_p^P (\frac{1}{2} + 2(\rho_p - a)JJ(\xi_p))\Psi_p^T\right)\right)$$

$$a = \frac{-1 + \frac{P}{2} + \sum_p 2JJ(\xi_p)(\Psi_p\mu_q + \rho_p)}{2\sum_p JJ(\xi_p)}$$

$$\xi_p = \sqrt{\Psi_p\Sigma_q\Psi_p^T + (\Psi_p\mu_q + \rho_p - a)^2}$$

Three Computational Tasks (where do the above fit in)

1. Given the embeddings produce (quickly) an approximate posterior over: ω
    1. We can run the EM algorithm to get a VB approx of theposterior on ω
    2. We can evaluate an AE to get an approx of the posterior on ω
2. Learn the embeddings Ψ, ρ (integrating over the user embedding ω)
    1. We can train using SGD using an auto-encoder either using the re-parameterization trick or the Bouchard bound.
    2. You could use the EM algorithm between SGD updates – this didn't really work, EM too slow.
3. Next item prediction (this one isn't a real problem actually)
    1. Either Monte Carlo or just ignore uncertainty and use the posterior mean.

Train Algorithm	Online Latent	Online Next Item	RC@5	DCG@5
Pop			0.456	0.440
ItemKNN			0.461	0.492
RNN			0.620	0.646
Bouch/AE	AE	MC	0.712	0.796
Bouch/AE	AE	mean	0.712	0.777
Bouch/AE	EM	MC	0.738	0.796
Bouch/AE	EM	mean	**0.748**	0.796
RT/AE	AE	MC	0.707	**0.802**
RT/AE	AE	mean	0.697	0.784
RT/AE	EM	MC	0.738	**0.802**
RT/AE	EM	mean	0.733	**0.802**
RT/Deep AE	AE	MC	0.697	0.785
RT/Deep AE	AE	mean	0.717	0.775
RT/Deep AE	EM	MC	0.733	0.785
RT/Deep AE	EM	mean	0.733	0.787

Table 2: Results on the testset for all approaches on the RecoGym dataset with 20 products. For both metrics, a higher value is better.

criteo

Train Algorithm	Online Latent	Online Next Item	RC@5	DCG@5
ItemKNN			0.020	0.024
Pop			0.020	0.016
RNN			0.035	0.033
Bouch/AE	AE	MC	0.082	0.128
Bouch/AE	AE	mean	0.082	0.079
Bouch/AE	EM	MC	**0.117**	0.128
Bouch/AE	EM	mean	**0.117**	**0.130**
RT/AE	AE	MC	0.061	0.047
RT/AE	AE	mean	0.056	0.059
RT/AE	EM	MC	0.051	0.047
RT/AE	EM	mean	0.051	0.047
RT/Deep AE	AE	MC	0.090	0.105
RT/Deep AE	AE	mean	0.080	0.068
RT/Deep AE	EM	MC	0.090	0.105
RT/Deep AE	EM	mean	0.090	0.106

Table 3: Results on the testset for all approaches on the RecoGym dataset with 2000 products. For both metrics, a higher value is better.

Train Algorithm	Online Latent	Online Next Item	RC@5	DCG@5
Pop			0.143	0.147
ItemKNN			**0.804**	**0.921**
RNN			0.690	0.781
Bouch/AE	AE	MC	0.433	0.420
Bouch/AE	AE	mean	0.451	0.562
Bouch/AE	EM	MC	0.386	0.420
Bouch/AE	EM	mean	0.429	0.497
RT/AE	AE	MC	0.495	0.731
RT/AE	AE	mean	0.616	0.658
RT/AE	EM	MC	0.693	0.731
RT/AE	EM	mean	0.707	0.768
RT/Deep AE	AE	MC	0.751	0.868
RT/Deep AE	AE	mean	0.771	0.876
RT/Deep AE	EM	MC	0.772	0.868
RT/Deep AE	EM	mean	0.775	0.873

Table 4: Results on the testset for all approaches on the YooChoose dataset with 100 products. For both metrics, a higher value is better.

Recent Work

1. Negative Sampling as stochastic approximation
2. Speed up the EM algorithm
3. Add in temporal behaviour
4. Production data with p(click) model

Robbins Monro Algorithm

SGD form

$$D_\theta L(\theta) = \sum_n \log f'(x_n|\theta) = 0$$

Which has a fixed point:

$$\theta = \theta - D_\theta L(\theta)$$

EM Algorithm can often be written directly as a fixed point algorithm (by nesting the updates)

$$\theta = \sum_n g(x_n \theta)$$

In both cases the sums are over the number of records (e.g. number of users).

The Likelihood is not a sum over P

$$\log p(v_1, .., v_T, \omega_u | \Psi) = \left(\sum_t^T \Psi_{v_t} \omega_u + \rho_{v_t} \right)$$

$$- T \log\{ \sum_p^P \exp(\Psi_p \omega_u + \rho_p) \} - \frac{K}{2} \log(2\pi) - \frac{1}{2} \omega_u^T \omega_u$$

Negative sampling is a heuristic to deal with this.. But it isn't really easily justified as maximising the above..

criteo

But the Bouchard bound *is*

$$\mathcal{L} \geq \mathcal{L}_{\text{Bouch}} = \left(\sum_t^T \Psi_{v_t} \mu_q + \rho_{v_t} \right)$$

$$- T[a + \sum_p^P \frac{\Psi_p \mu_q + \rho_p - a - \xi_p}{2}$$

$$+ \lambda_{\text{JJ}}(\xi_p)\{(\Psi_p \mu_q + \rho_p - a)^2 + \Psi_p \Sigma_q \Psi_p^T - \xi_p^2\} + \log(1 + e^{\xi_p})]$$

$$- \frac{K}{2} \log(2\pi) - \frac{1}{2}\{\mu_q^T \mu_q + \text{trace}(\Sigma_q)\} + \frac{1}{2} \log |2\pi e \Sigma_q|.$$

criteo

Online EM Algorithm

$$\Sigma_q^{-1} = I_k + 2\sum_p \lambda_{JJ}(\sqrt{\Psi_p \Sigma_q \Psi_p^T + (\Psi_p \mu_q + \rho_p - a)^2})\Psi_p^T \Psi_p,$$

$$\Sigma_q^{-1}\mu_q = \left((\sum_t^T \Psi_{v_t}^T) - T\left[\sum_p^P \{\frac{1}{2} + 2(\rho_p - a)\lambda_{JJ}(\sqrt{\Psi_p \Sigma_q \Psi_p^T + (\Psi_p \mu_q + \rho_p - a)^2})\}\Psi_p^T\right]\right)$$

$$a = a + \frac{-1 + \frac{P}{2}}{2} + \sum_p \lambda_{JJ}(\xi_p)(\Psi_p \mu_q + \rho_p) - a\sum_p \lambda_{JJ}(\xi_p)$$

Original EM Algorithm (why is it slow)

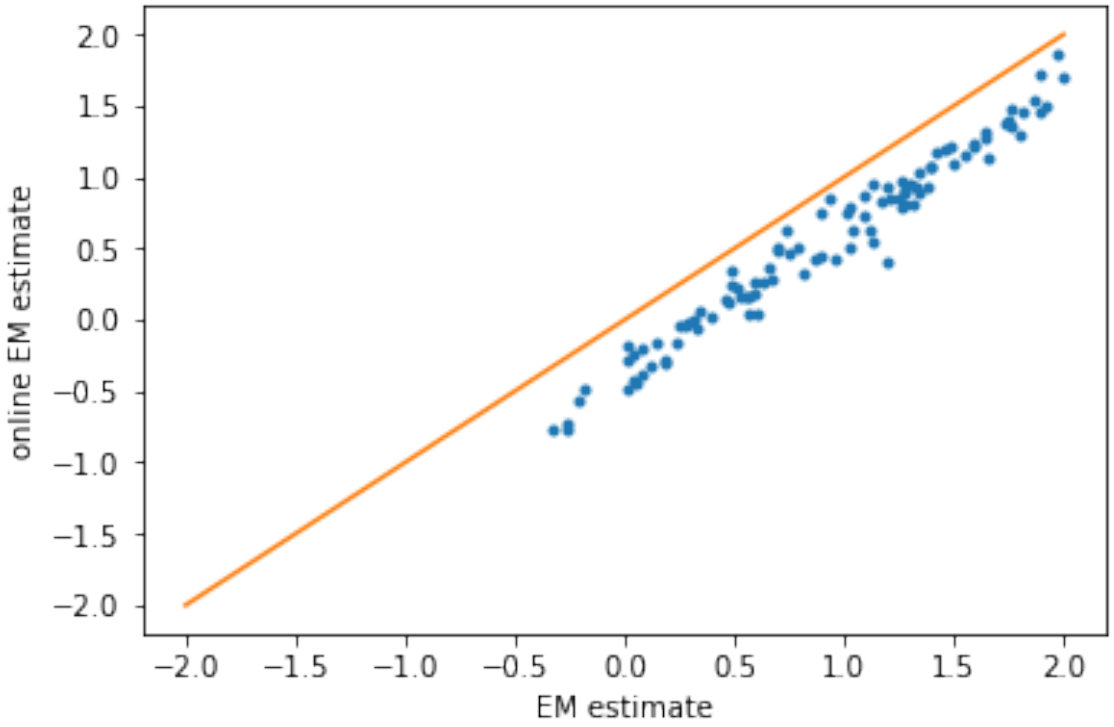
$$\Sigma_q^{-1} = I_k + 2\sum_p JJ(\xi_p)\Psi_p^T\Psi_p$$

$$\mu_q = \Sigma_q^{-1}\left(\left(\sum_t^T \Psi_{v_t}^T\right) - T\left(\sum_p^P (\frac{1}{2} + 2(\rho_p - a)JJ(\xi_p))\Psi_p^T\right)\right)$$

$$a = \frac{-1 + \frac{P}{2} + \sum_p 2JJ(\xi_p)(\Psi_p\mu_q + \rho_p)}{2\sum_p JJ(\xi_p)}$$

$$\xi_p = \sqrt{\Psi_p\Sigma_q\Psi_p^T + (\Psi_p\mu_q + \rho_p - a)^2}$$

Online EM Example



P=1000
K=100

EM Algorithm 23 s
Online EM 40 ms

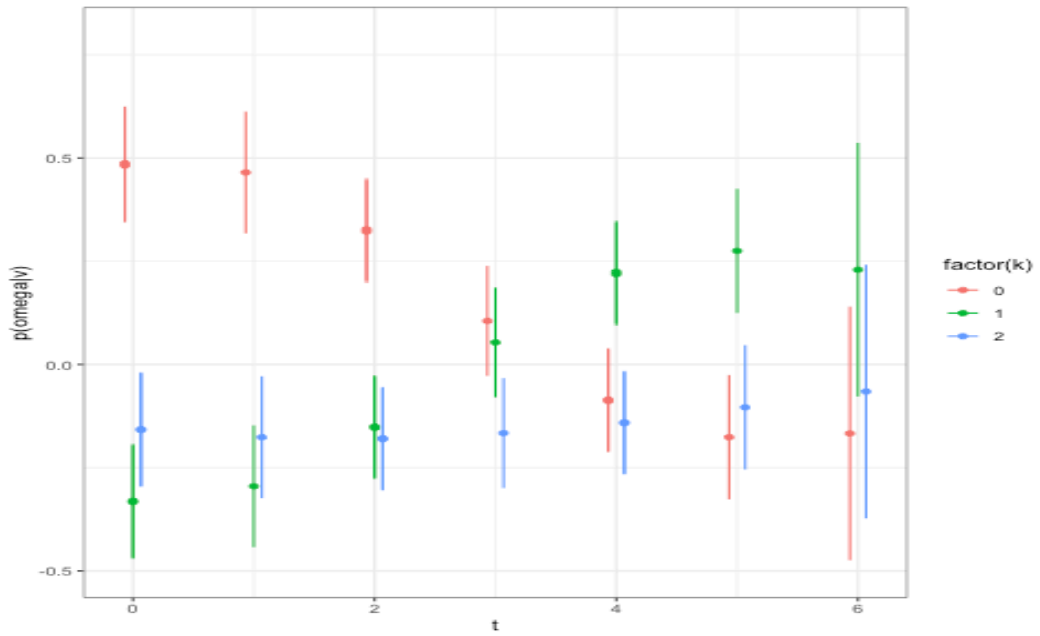
Approx 500 x faster (but increase depends on P).

It is still too slow $P=10^6$ is a couple of seconds

criteo

Temporal Banners

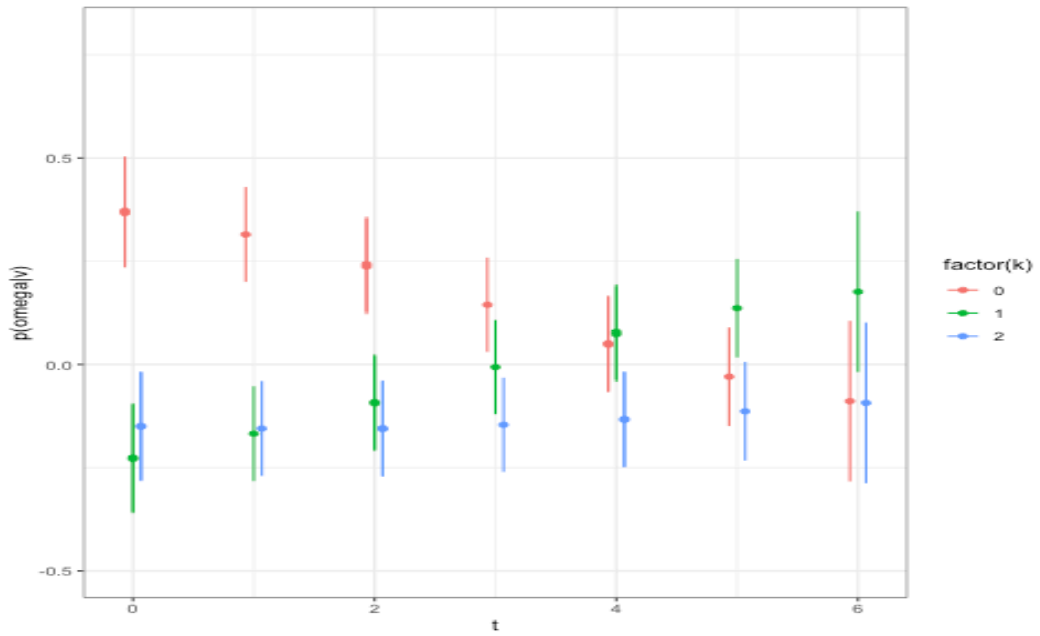
We extend the model so that ω_i for a user now varies in time. This results in a new EM algorithm (and variational parameters for every point in time).



Length scale 3

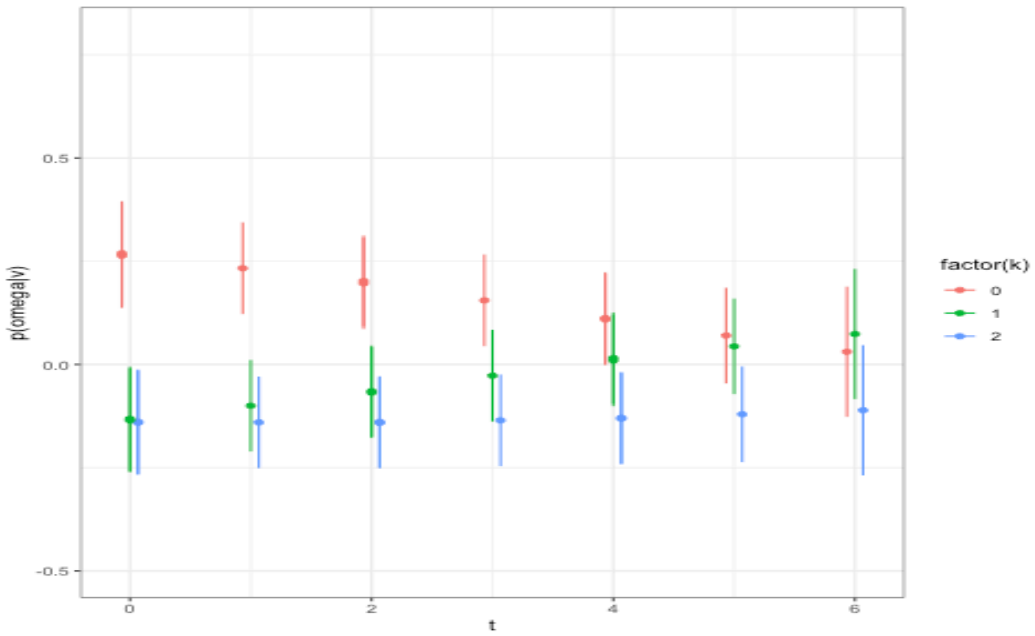
criteo

Temporal Banners



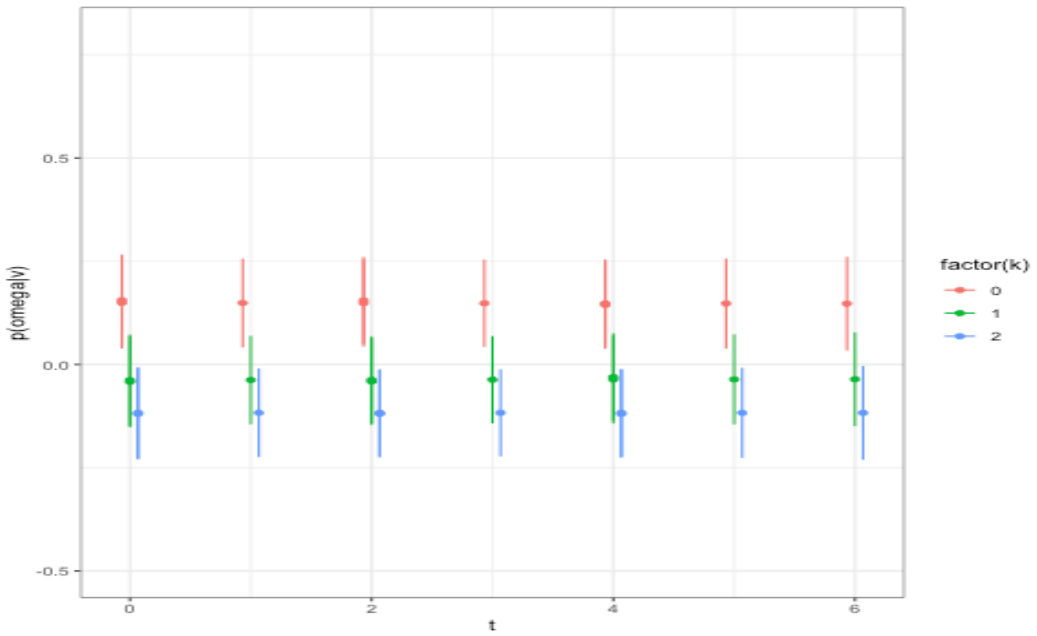
Length scale 6

Temporal Banners



Length scale 10

Temporal Banners



Length scale 100

Unlike Word2Vec we use the User Item Matrix.. Why?

$$C_{u,p} = \sum_{t=1}^{T_u} \mathbf{1}\{v_t = p\}$$

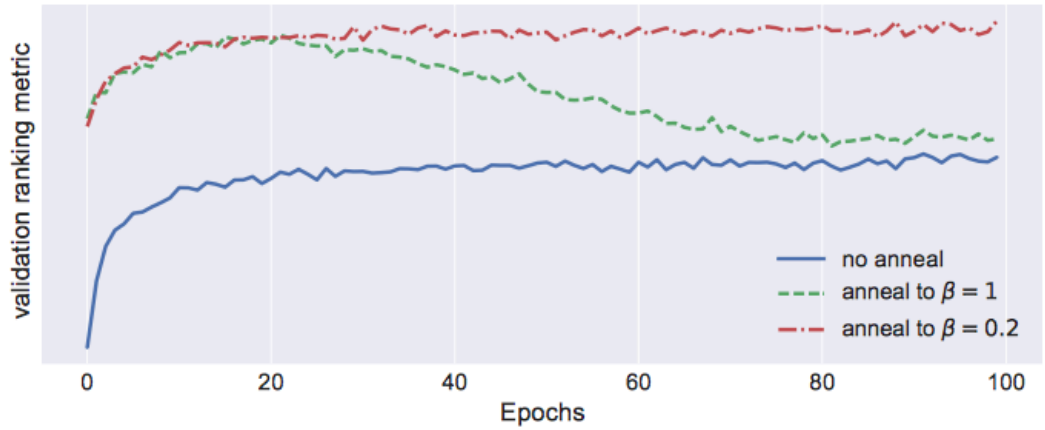
C is then thresholded so it is either 0 or 1.

$$C^T C = \begin{pmatrix} 1 & 1 & 0 \\ 1 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix}.$$

$$C^T C \neq \begin{pmatrix} 1 & 1 & 1 \\ 1 & 1 & 0 \\ 1 & 0 & 1 \end{pmatrix}$$

Liang et al

$$\mathbf{z}_u \sim \mathcal{N}(0, \mathbf{I}_K), \quad \pi(\mathbf{z}_u) \propto \exp\{f_\theta(\mathbf{z}_u)\},$$
$$\mathbf{x}_u \sim \text{Mult}(N_u, \pi(\mathbf{z}_u)).$$



criteo