Amortized Variational Inference for Simple Hierarchical Models

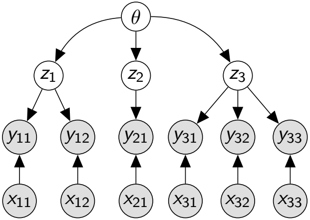
Abhinav Agrawal and Justin Domke

College of Information and Computer Science, UMass Amherst

Paper in a slide

Hierarchical Branch Distribution (HBD)

$p(\theta, z, y|x) = p(\theta) \prod_{i=1}^{N} p(z_i|\theta) p(y_i|\theta, z_i, x_i)$



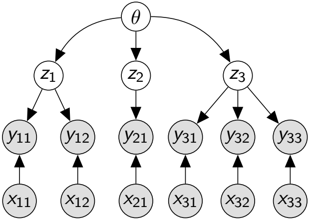
Paper in a slide

Hierarchical Branch Distribution (HBD)

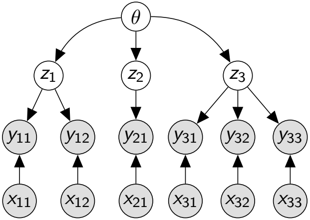
$p(\theta,z,y|x)=p(\theta)\prod_{i=1}^{N}p(z_i|\theta)p(y_i|\theta,z_i,x_i)$

Variational Inference (VI)

Approximate $p(\theta, z|x, y)$ with $q_\phi(\theta, z)$.



Paper in a slide



Hierarchical Branch Distribution (HBD)

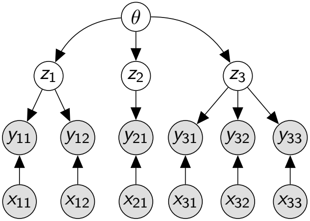
$p(\theta, z, y|x) = p(\theta) \prod_{i=1}^{N} p(z_i|\theta) p(y_i|\theta, z_i, x_i)$

Variational Inference (VI)

Approximate $p(\theta, z|x, y)$ with $q_\phi(\theta, z)$.

What we want? Make sub-sampling *work* for large models.

Paper in a slide



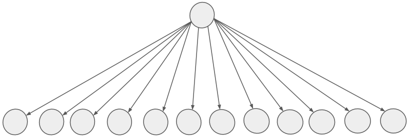
Hierarchical Branch Distribution (HBD)

$p(\theta, z, y|x) = p(\theta) \prod_{i=1}^{N} p(z_i|\theta) p(y_i|\theta, z_i, x_i)$

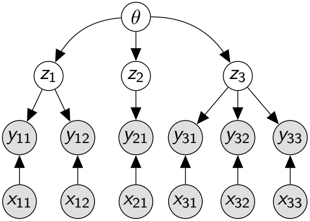
Variational Inference (VI)

Approximate $p(\theta, z|x, y)$ with $q_\phi(\theta, z)$.

What we want? Make sub-sampling *work* for large models.



Paper in a slide

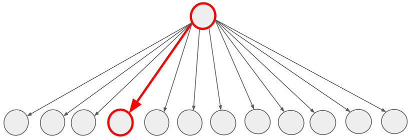


Hierarchical Branch Distribution (HBD)

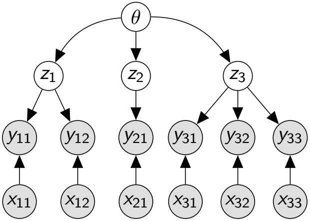
$p(\theta, z, y|x) = p(\theta) \prod_{i=1}^{N} p(z_i|\theta) p(y_i|\theta, z_i, x_i)$

Variational Inference (VI)
Approximate $p(\theta, z|x, y)$ with $q_\phi(\theta, z)$.

What we want? Make sub-sampling *work* for large models.



Paper in a slide



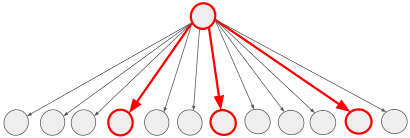
Hierarchical Branch Distribution (HBD)

$p(\theta, z, y|x) = p(\theta) \prod_{i=1}^{N} p(z_i|\theta) p(y_i|\theta, z_i, x_i)$

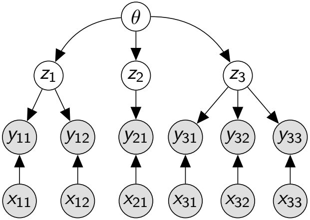
Variational Inference (VI)

Approximate $p(\theta, z|x, y)$ with $q_\phi(\theta, z)$.

What we want? Make sub-sampling *work* for large models.



Paper in a slide



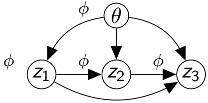
Hierarchical Branch Distribution (HBD)

$p(\theta,z,y|x) = p(\theta)\prod_{i=1}^{N} p(z_i|\theta)p(y_i|\theta,z_i,x_i)$

Variational Inference (VI)
Approximate $p(\theta, z|x, y)$ with $q_\phi(\theta, z)$.

What we want? Make sub-sampling *work* for large models.

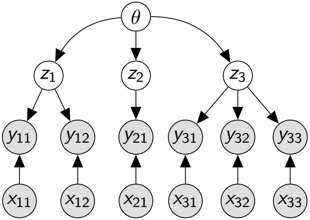
Joint Approach, q_ϕ^{Joint}



$q_\phi^{\mathrm{Joint}}(\theta,z) = q_\phi(\theta,z)$

Optimize ϕ

Paper in a slide



Hierarchical Branch Distribution (HBD)

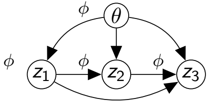
$p(\theta,z,y|x)=p(\theta)\prod_{i=1}^{N} p(z_i|\theta)p(y_i|\theta,z_i,x_i)$

Variational Inference (VI)

Approximate $p(\theta, z|x, y)$ with $q_\phi(\theta, z)$.

What we want? Make sub-sampling *work* for large models.

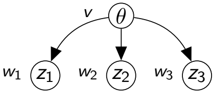
Joint Approach, q_ϕ^{Joint}



$q_\phi^{\text{Joint}}(\theta,z)=q_\phi(\theta,z)$

Optimize ϕ

Branch Approach, $q_{v,w}^{\text{Branch}}$



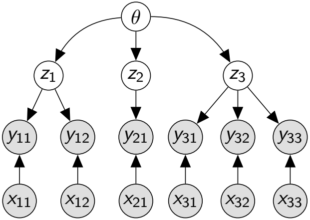
$q_{v,w}^{\text{Branch}}(\theta,z)=q_v(\theta)\prod_{i=1}^{3} q_{w_i}(z_i|\theta)$

Optimize $v, \{w_i\}_{i=1}^{N}$

Paper in a slide

Hierarchical Branch Distribution (HBD)

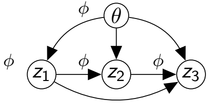
$p(\theta,z,y|x)=p(\theta)\prod_{i=1}^{N} p(z_i|\theta)p(y_i|\theta,z_i,x_i)$



Variational Inference (VI)
Approximate $p(\theta, z|x, y)$ with $q_\phi(\theta, z)$.

What we want? Make sub-sampling *work* for large models.

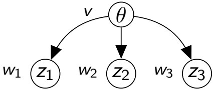
Joint Approach, q_ϕ^{Joint}



$q_\phi^{\mathrm{Joint}}(\theta,z)=q_\phi(\theta,z)$

Optimize ϕ

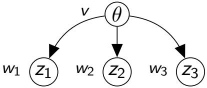
Branch Approach, $q_{v,w}^{\mathrm{Branch}}$



$q_{v,w}^{\mathrm{Branch}}(\theta,z)=q_v(\theta)\prod_{i=1}^{3} q_{w_i}(z_i|\theta)$

Optimize $v, \{w_i\}_{i=1}^{N}$

Our Approach, $q_{v,u}^{\mathrm{Amort}}$



$w_i=\mathtt{net}_u(x_i,y_i)$

Optimize v, u

Why use $q_{v,u}^{\mathrm{Amort}}$?

Why use $q_{v,u}^{\mathrm{Amort}}$?

Faster Optimization

$q_{v,u}^{\mathrm{Amort}}$ updates u with every step.

Why use $q_{v,u}^{\mathrm{Amort}}$?

Faster Optimization

$q_{v,u}^{\mathrm{Amort}}$ updates u with every step.

Provably as accurate

Assuming sufficiently capable $\mathtt{net}_u$,

$$\min_{v,u} KL\left(q_{v,u}^{\mathrm{Amort}} \| p\right) \leq \min_{\phi} KL\left(q_{\phi}^{\mathrm{Joint}} \| p\right).$$

Why use $q^{\mathrm{Amort}}_{v,u}$?

Faster Optimization

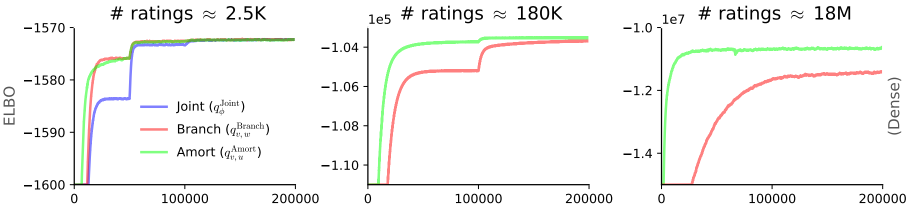
$q^{\mathrm{Amort}}_{v,u}$ updates u with every step.

Provably as accurate

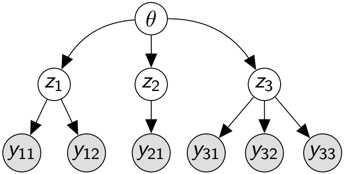
Assuming sufficiently capable $\mathtt{net}_u$,

$$\min_{v,u} KL\left(q^{\mathrm{Amort}}_{v,u} \| p\right) \leq \min_{\phi} KL\left(q^{\mathrm{Joint}}_{\phi} \| p\right).$$

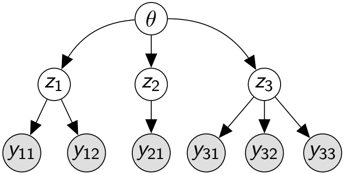
MovieLens dataset. ≈ 160K users. ≈ 18M ratings. ≈ 1.6M variables.

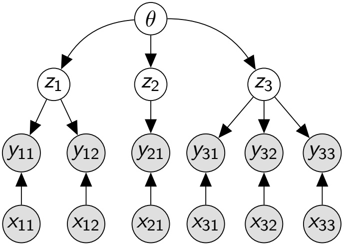


Hierarchical Branch Distributions

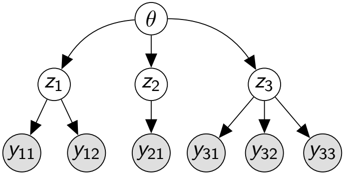


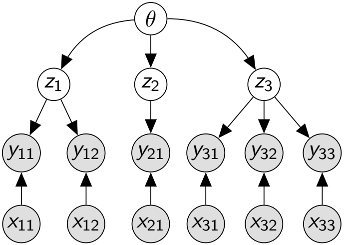
Hierarchical Branch Distributions

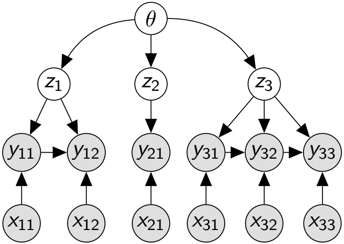




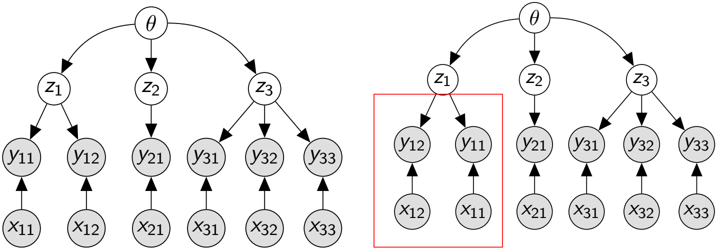
Hierarchical Branch Distributions





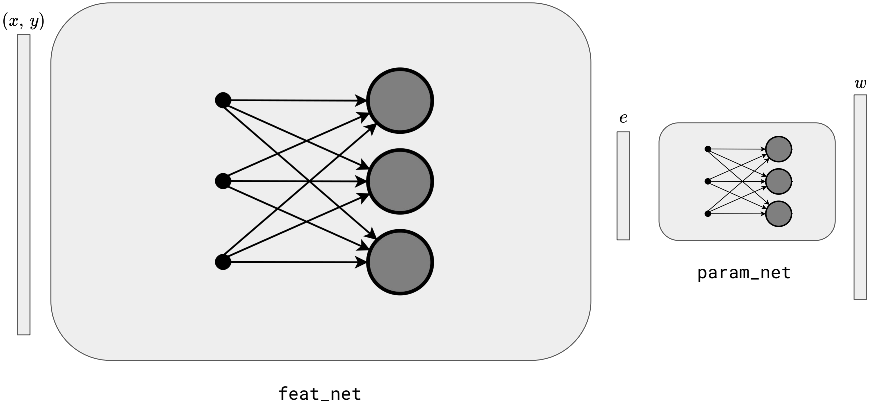


Locally i.i.d

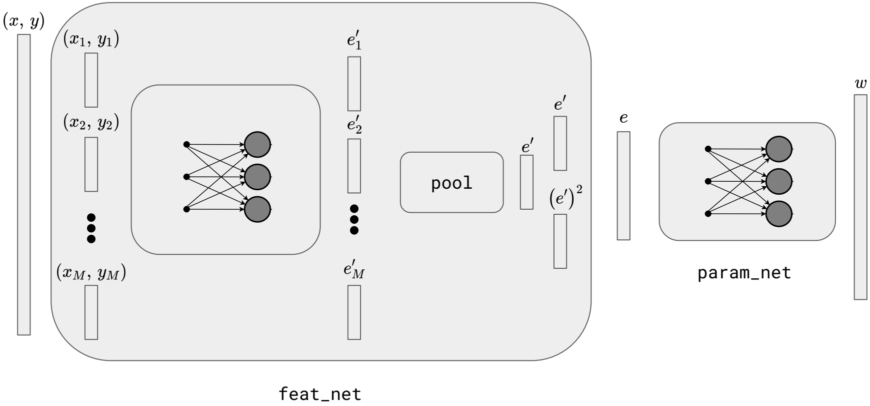


θ
z1
z2
z3
y11
y12
y21
y31
y32
y33
x11
x12
x21
x31
x32
x33
θ
z1
z2
z3
y12
y11
y21
y31
y32
y33
x12
x11
x21
x31
x32
x33

Designing $\mathtt{net}_u$



Designing $\texttt{net}_u$



Amortized Gaussian VI

Dense Gaussian

Joint Approach	Branch Approach	Our Approach

Amortized Gaussian VI

Dense Gaussian

Joint Approach	Branch Approach	Our Approach
$\mathcal{N}(\theta, z \mid \mu, \Sigma)$		
$\phi = (\mu, \Sigma)$		

Amortized Gaussian VI

Dense Gaussian

Joint Approach	Branch Approach	Our Approach
$\mathcal{N}(\theta, z \mid \mu, \Sigma)$	$\mathcal{N}(\theta \mid \mu_0, \Sigma_0) \prod_{i=1}^{N} \mathcal{N}(z_i \mid \mu_i + A_i\theta, \Sigma_i)$	
$\phi = (\mu, \Sigma)$	$v = (\mu_0, \Sigma_0),\ w = \{\mu_i, A_i, \Sigma_i\}_{i=1}^{N}$	

Amortized Gaussian VI

Dense Gaussian

Joint Approach	Branch Approach	Our Approach
$\mathcal{N}(\theta, z \mid \mu, \Sigma)$	$\mathcal{N}(\theta \mid \mu_0, \Sigma_0) \prod_{i=1}^{N} \mathcal{N}(z_i \mid \mu_i + A_i \theta, \Sigma_i)$	$\mathcal{N}(\theta \mid \mu_0, \Sigma_0) \prod_{i=1}^{N} \mathcal{N}(z_i \mid \mu_i + A_i \theta, \Sigma_i)$ where, $(\mu_i, A_i, \Sigma_i) = \mathtt{net}_u(x_i, y_i)$
$\phi = (\mu, \Sigma)$	$v = (\mu_0, \Sigma_0)$, $w = \{\mu_i, A_i, \Sigma_i\}_{i=1}^{N}$	$v = (\mu_0, \Sigma_0), u$

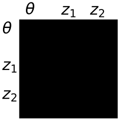
Amortized Gaussian VI

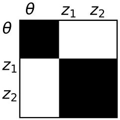
Dense Gaussian

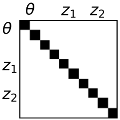
Joint Approach	Branch Approach	Our Approach
$\mathcal{N}(\theta, z \vert \mu, \Sigma)$	$\mathcal{N}(\theta \vert \mu_0, \Sigma_0) \prod_{i=1}^{N} \mathcal{N}(z_i \vert \mu_i + A_i\theta, \Sigma_i)$	$\mathcal{N}(\theta \vert \mu_0, \Sigma_0) \prod_{i=1}^{N} \mathcal{N}(z_i \vert \mu_i + A_i\theta, \Sigma_i)$ where, $(\mu_i, A_i, \Sigma_i) = \texttt{net}_u(x_i, y_i)$
$\phi = (\mu, \Sigma)$	$v = (\mu_0, \Sigma_0)$, $w = \{\mu_i, A_i, \Sigma_i\}_{i=1}^{N}$	$v = (\mu_0, \Sigma_0), u$

Scalable Faster Accurate

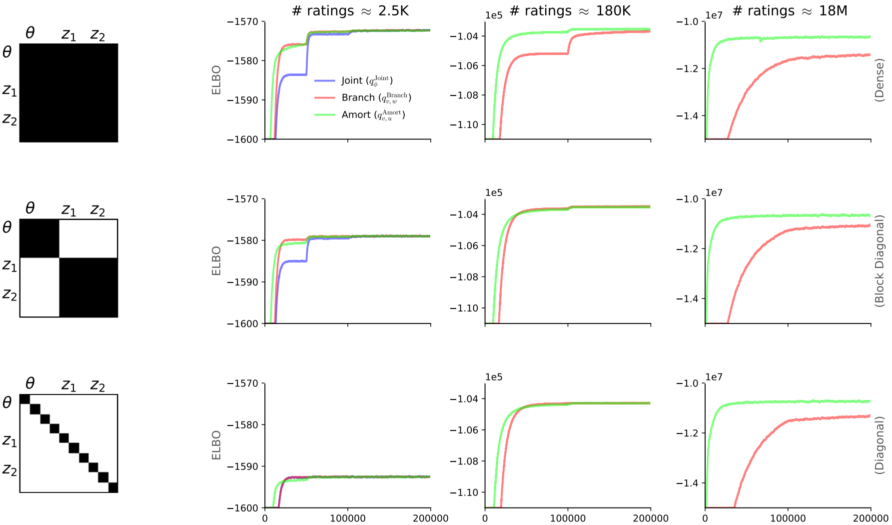
More results







More results



Insights

When N is large, θ collapses.

No need to condition on it.