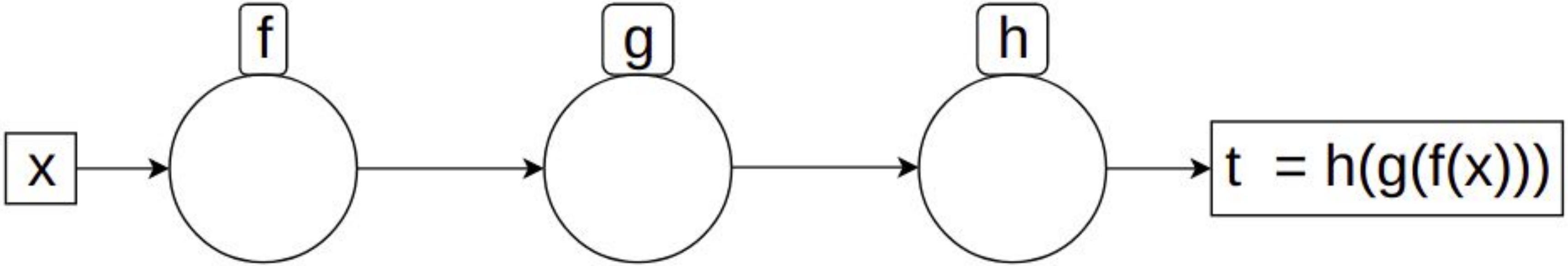
Gradient Estimation with Stochastic Softmax Tricks

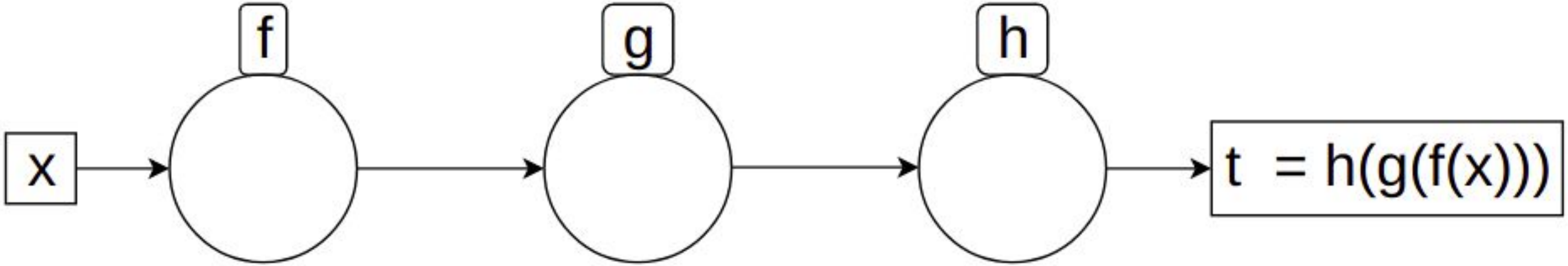
Max B. Paulus and Dami Choi and Daniel Tarlow and Andreas Krause and Chris J. Maddison

2020

Computational graph

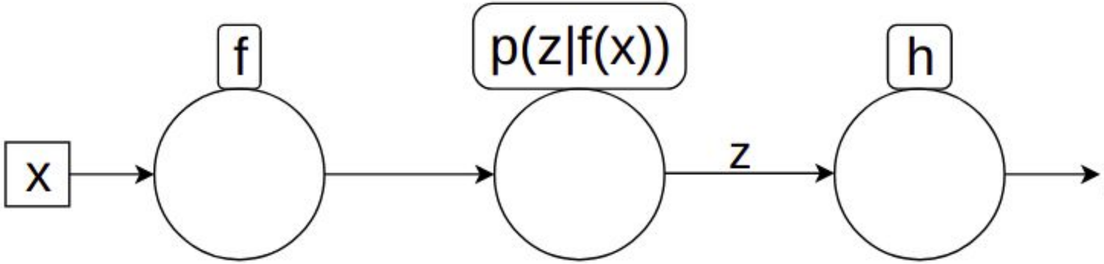


Computational graph & backprop



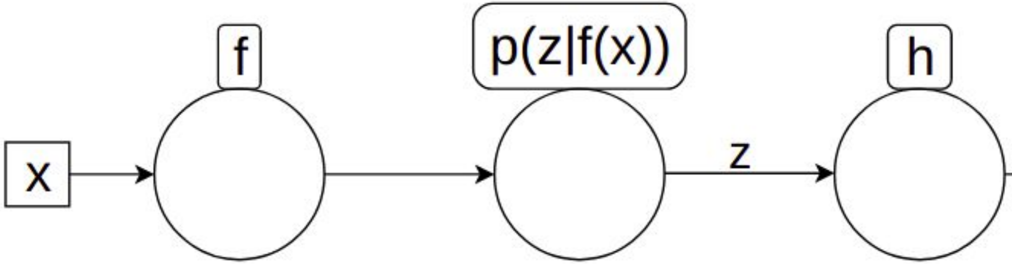
$$\frac{dt}{dx} = \frac{dh}{dg}\frac{dg}{df}\frac{df}{dx}$$

Stochastic computational graph



$$t = h(z), z \sim p(z|f(x))$$

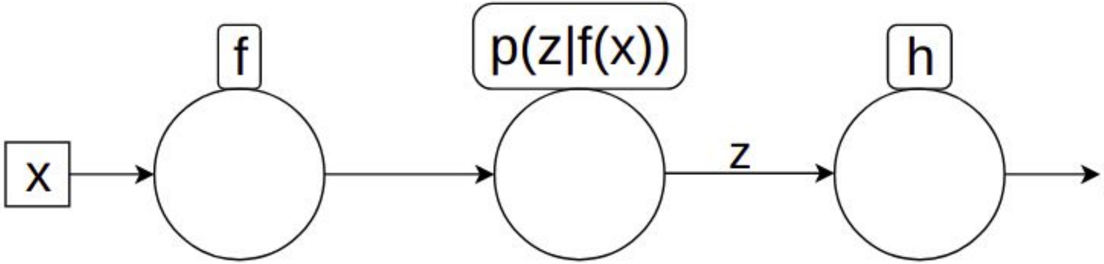
Stochastic computational graph



$$t = h(z), z \sim p(z|f(x))$$

Действия, которые мы принимаем, чтобы получить награду являются случайными от политики

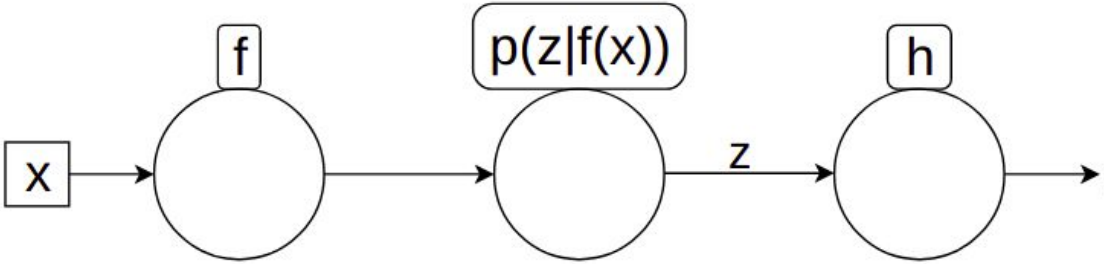
Stochastic computational graph & backprop ?



$$t = h(z), z \sim p(z|f(x))$$

$$\frac{d\mathbb{E}\mathrm{t}}{dx} = \ ?$$

Stochastic computational graph & backprop ?



$$t = h(z), z \sim p(z|f(x))$$

$$\frac{d\mathbb{E}\mathrm{t}}{dx} = \ ?$$

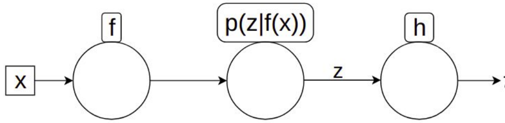
$$\frac{d\mathbb{E}\mathrm{t}}{dx} = \frac{1}{dx} \int p(z|f(x))h(z)dz$$

$$\frac{d\mathbb{E}\mathrm{t}}{dx} = \frac{1}{dx}\int p(z|f(x))h(z)dz$$

1) Reparameterization trick

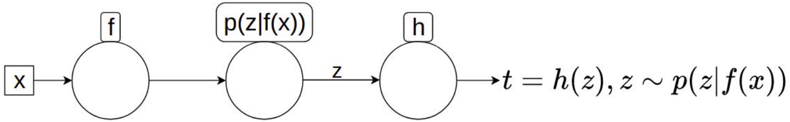
2) REINFORCE

Reparameterization trick



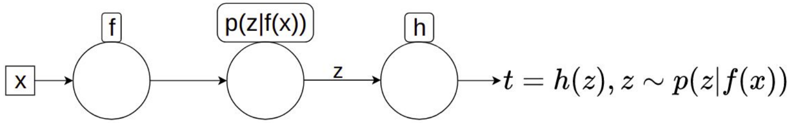
$t = h(z), z \sim p(z|f(x))$

Reparameterization trick



$$\frac{1}{dx}\int p(z|f(x))h(z)dz$$

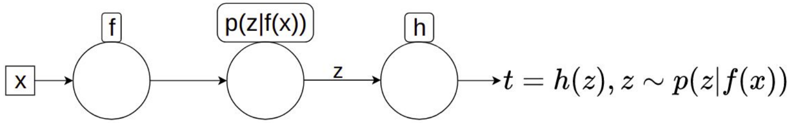
Reparameterization trick



$$\frac{1}{dx}\int p(z|f(x))h(z)dz \longrightarrow \left\{\begin{matrix} z = g(\hat{\varepsilon}, f(x)), \\ \hat{\varepsilon} \sim r(\varepsilon) \end{matrix}\right\}$$

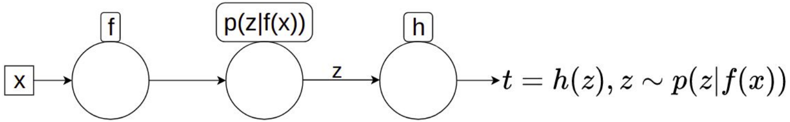
$p(x\|y)$	$r(\epsilon)$	$g(\epsilon, y)$
$\mathcal{N}(x\|\mu, \sigma^2)$	$\mathcal{N}(\epsilon\|0, 1)$	$x = \sigma\epsilon + \mu$
$\mathcal{G}(x\|1, \beta)$	$\mathcal{G}(\epsilon\|1, 1)$	$x = \beta\epsilon$
$\mathcal{E}(x\|\lambda)$	$\mathcal{U}(\epsilon\|0, 1)$	$x = -\frac{\log \epsilon}{\lambda}$
$\mathcal{N}(x\|\mu, \Sigma)$	$\mathcal{N}(\epsilon\|0, I)$	$x = A\epsilon + \mu$, where $AA^T = \Sigma$

Reparameterization trick



$$\frac{1}{dx}\int p(z|f(x))h(z)dz \longrightarrow \left\{\begin{matrix} z = g(\hat{\varepsilon}, f(x)), \\ \hat{\varepsilon} \sim r(\varepsilon) \end{matrix}\right\} \xrightarrow{\text{Integral over density function is 1}} \frac{d}{dx}h(g(\hat{\varepsilon}, f(x)))$$

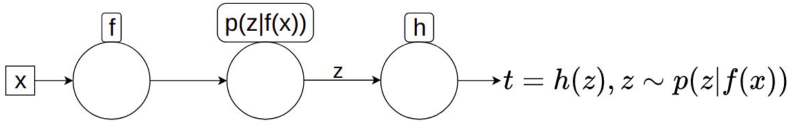
Reparameterization trick



$$\frac{1}{dx}\int p(z|f(x))h(z)dz \longrightarrow \begin{Bmatrix} z = g(\hat{\varepsilon}, f(x)), \\ \hat{\varepsilon} \sim r(\varepsilon) \end{Bmatrix} \xrightarrow{\text{Integral over density function is 1}} \frac{d}{dx}h(g(\hat{\varepsilon}, f(x))$$

$$\approx \frac{dh}{dg}\frac{\partial g(\hat{\varepsilon}, f(x))}{\partial f}\frac{df}{dx}$$

Reparameterization trick

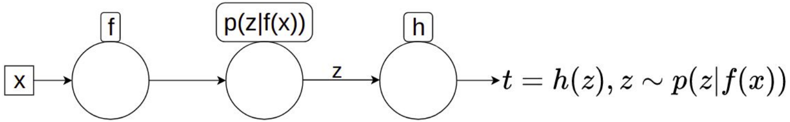


$$\frac{1}{dx}\int p(z|f(x))h(z)dz \longrightarrow \left\{\begin{matrix} z = g(\hat{\varepsilon}, f(x)), \\ \hat{\varepsilon} \sim r(\varepsilon) \end{matrix}\right\} \xrightarrow{\text{Integral over density function is 1}} \frac{d}{dx}h(g(\hat{\varepsilon}, f(x))$$

$$\approx \frac{dh}{dg}\frac{\partial g(\hat{\varepsilon}, f(x))}{\partial f}\frac{df}{dx}$$

Seems familiar?

Reparameterization trick

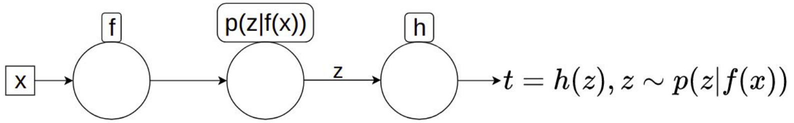


$$\frac{1}{dx}\int p(z|f(x))h(z)dz \longrightarrow \left\{\begin{matrix} z = g(\hat{\varepsilon}, f(x)), \\ \hat{\varepsilon} \sim r(\varepsilon) \end{matrix}\right\} \xrightarrow{\text{Integral over density function is 1}} \frac{d}{dx}h(g(\hat{\varepsilon}, f(x))$$

$$\approx \frac{dh}{dg}\frac{\partial g(\hat{\varepsilon}, f(x))}{\partial f}\frac{df}{dx}$$

$$\frac{dt}{dx} = \frac{dh}{dg}\frac{dg}{df}\frac{df}{dx}$$

Reparameterization trick



$$\frac{1}{dx}\int p(z|f(x))h(z)dz \longrightarrow \left\{\begin{matrix} z = g(\hat{\varepsilon}, f(x)), \\ \hat{\varepsilon} \sim r(\varepsilon) \end{matrix}\right\} \xrightarrow{\text{Integral over density function is 1}} \frac{d}{dx}h(g(\hat{\varepsilon}, f(x)))$$

g should be differentiable

$$\approx \frac{dh}{dg}\boxed{\frac{\partial g(\hat{\varepsilon}, f(x))}{\partial f}}\frac{df}{dx}$$

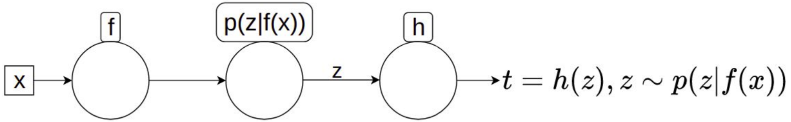
$$\frac{dt}{dx} = \frac{dh}{dg}\frac{dg}{df}\frac{df}{dx}$$

Reparametrization examples

$p(x\|y)$	$r(\epsilon)$	$g(\epsilon, y)$
$\mathcal{N}(x\|\mu, \sigma^2)$	$\mathcal{N}(\epsilon\|0, 1)$	$x = \sigma\epsilon + \mu$
$\mathcal{G}(x\|1, \beta)$	$\mathcal{G}(\epsilon\|1, 1)$	$x = \beta\epsilon$
$\mathcal{E}(x\|\lambda)$	$\mathcal{U}(\epsilon\|0, 1)$	$x = -\frac{\log \epsilon}{\lambda}$
$\mathcal{N}(x\|\mu, \Sigma)$	$\mathcal{N}(\epsilon\|0, I)$	$x = A\epsilon + \mu, \text{where } AA^T = \Sigma$

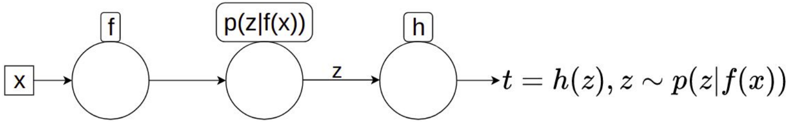
Slide credit: Dmitry Vetrov

REINFORCE



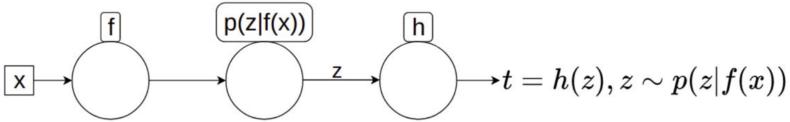
$$\frac{1}{dx}\int p(z|f(x))h(z)dz \; =$$

REINFORCE



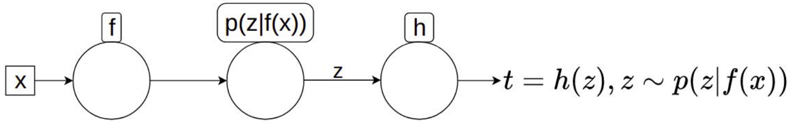
$$\frac{1}{dx}\int p(z|f(x))h(z)dz = \int \frac{\partial p(z|f(x))}{\partial x} h(z)dz =$$

REINFORCE



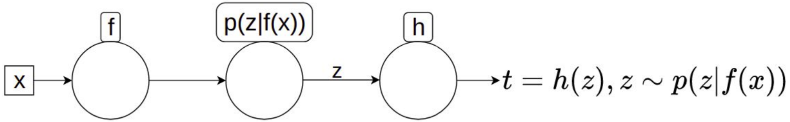
$$\frac{1}{dx}\int p(z|f(x))h(z)dz = \int \frac{\partial p(z|f(x))}{\partial x}h(z)dz = \left\{ \nabla_\theta \log p(x;\theta) = \frac{\nabla_\theta p(x;\theta)}{p(x;\theta)} \right\}$$

REINFORCE



$$\frac{1}{dx}\int p(z|f(x))h(z)dz = \int \frac{\partial p(z|f(x))}{\partial x} h(z)dz = \left\{ \nabla_\theta p(x;\theta) = \nabla_\theta \log p(x;\theta)p(x;\theta) \right\}$$

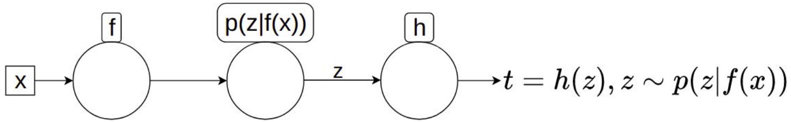
REINFORCE



$$\frac{1}{dx}\int p(z|f(x))h(z)dz = \int \frac{\partial p(z|f(x))}{\partial x}h(z)dz = \left\{ \nabla_\theta p(x;\theta) = \nabla_\theta \log p(x;\theta) p(x;\theta) \right\}$$

$$\approx \int p(z|f(x))\frac{\partial log p(z|f(x))}{\partial x}h(z)dz$$

REINFORCE

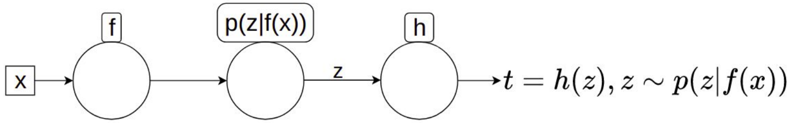


$$\frac{1}{dx}\int p(z|f(x))h(z)dz = \int \frac{\partial p(z|f(x))}{\partial x}h(z)dz = \left\{ \nabla_\theta p(x;\theta) = \nabla_\theta \log p(x;\theta)p(x;\theta) \right\}$$

random variable

$$\approx \underline{\int p(z|f(x))} \boxed{\frac{\partial log p(z|f(x))}{\partial x}h(z)} \underline{dz} =$$

REINFORCE



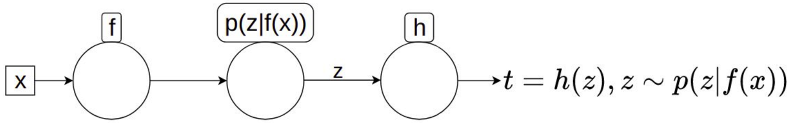
$$\frac{1}{dx}\int p(z|f(x))h(z)dz = \int \frac{\partial p(z|f(x))}{\partial x}h(z)dz = \left\{ \nabla_\theta p(x;\theta) = \nabla_\theta \log p(x;\theta)p(x;\theta) \right\}$$

random variable

Monte Carlo

$$\approx \underline{\int p(z|f(x))} \boxed{\frac{\partial log p(z|f(x))}{\partial x}h(z)}\underline{dz} = \left\{ \hat{z} \sim p(z|f(x)) \right\}$$

REINFORCE



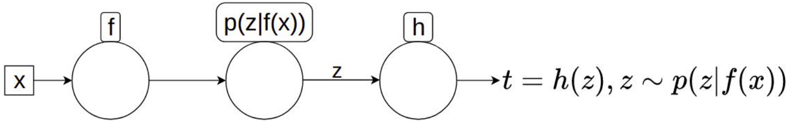
$$\frac{1}{dx}\int p(z|f(x))h(z)dz = \int \frac{\partial p(z|f(x))}{\partial x} h(z)dz = \left\{ \nabla_\theta p(x;\theta) = \nabla_\theta \log p(x;\theta)p(x;\theta) \right\}$$

random variable

Monte Carlo

$$\approx \int p(z|f(x)) \boxed{\frac{\partial logp(z|f(x))}{\partial x} h(z)} dz = \left\{ \hat{z} \sim p(z|f(x)) \right\} \approx \frac{\partial logp(\hat{z}|f(x))}{\partial f}\frac{\partial f}{\partial x} h(\hat{z})$$

REINFORCE



$$\frac{1}{dx}\int p(z|f(x))h(z)dz = \int \frac{\partial p(z|f(x))}{\partial x}h(z)dz = \left\{ \nabla_\theta p(x;\theta) = \nabla_\theta \log p(x;\theta)p(x;\theta) \right\}$$

$$\approx \int p(z|f(x))\frac{\partial logp(z|f(x))}{\partial x}h(z)dz = \overset{\text{Monte Carlo}}{\left\{ \hat{z} \sim p(z|f(x)) \right\}} \approx \frac{\partial logp(\hat{z}|f(x))}{\partial f}\frac{\partial f}{\partial x}\underline{h(\hat{z})}$$

Leads to high variance

$$\frac{d\mathbb{E}\mathrm{t}}{dx} = \frac{1}{dx}\int p(z|f(x))h(z)dz$$

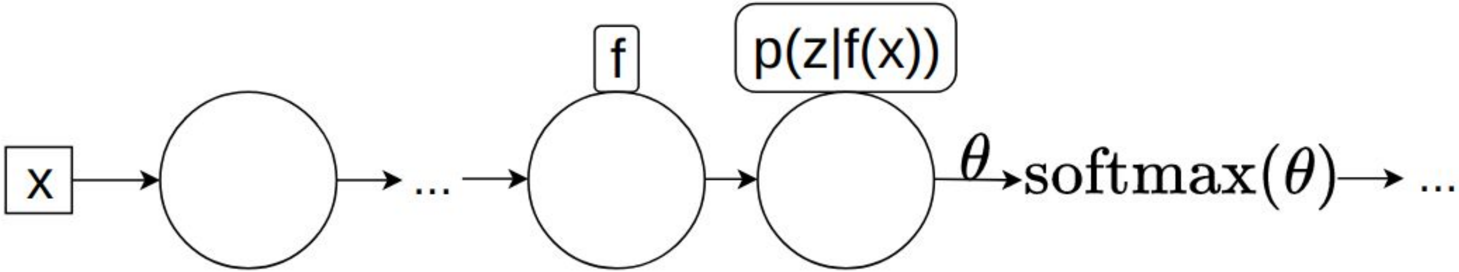
1) Reparameterization trick

+ Uses derivative of h
- Doesn’t work with categorical variables

2) REINFORCE

+ Works all the time
- Doesn’t use derivative of h

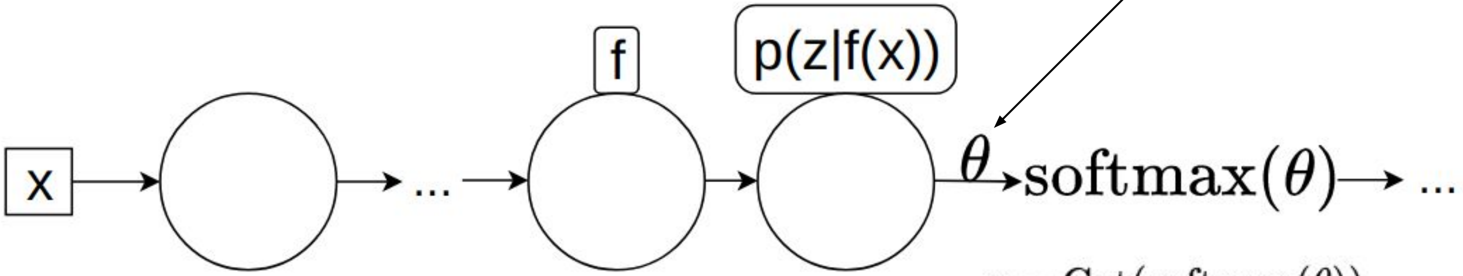
Gumbel argmax trick



$\theta \in \mathbb{R}^d$

Gumbel argmax trick

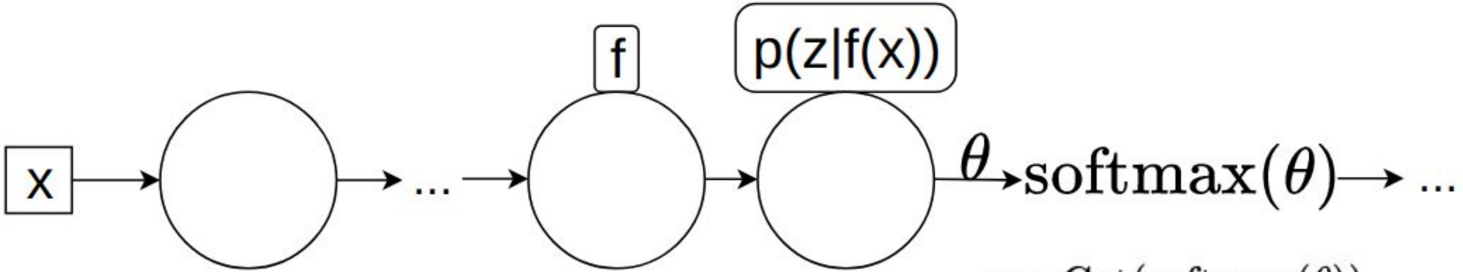
Logits of categorical random variable



$$\theta \in \mathbb{R}^d$$

$$z \sim \mathrm{Cat}(\mathrm{softmax}(\theta))$$

Gumbel argmax trick

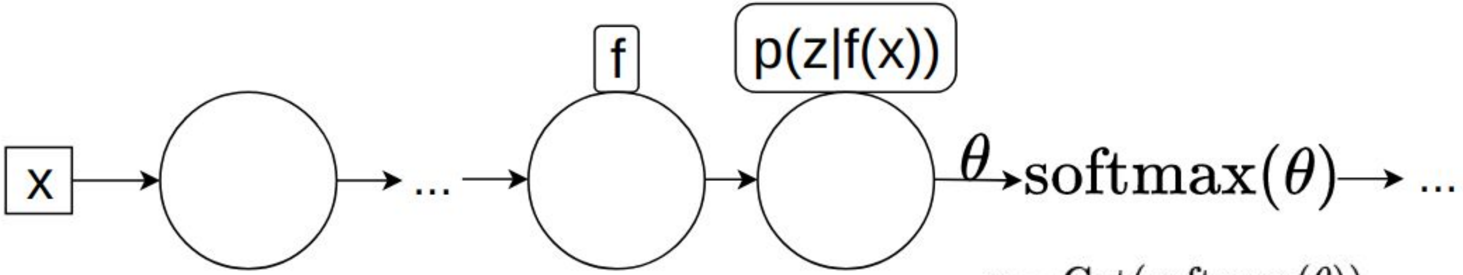


$\theta \in \mathbb{R}^d$

$z \sim \mathrm{Cat}(\mathrm{softmax}(\theta))$

- Gumbel trick defines $g_\theta(\varepsilon)$ for z
  - Let $\varepsilon_i = -\log(-\log u_i)$ for $u \sim U[0,1]^d$
  - Let $g_\theta(\varepsilon) = \operatorname*{argmax}_{i=1,\ldots,d}(\theta + \varepsilon)$
  - Then $g_\theta(\varepsilon) \overset{d}{=} z$

Gumbel argmax trick



$\theta \in \mathbb{R}^d$

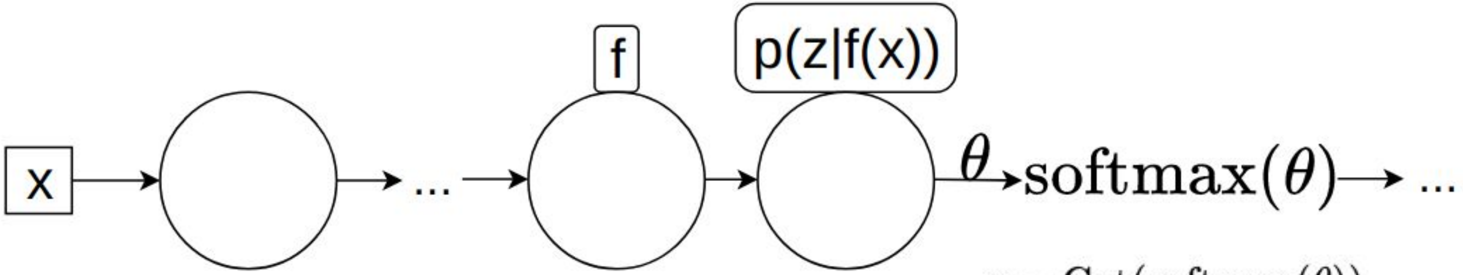
$z \sim \text{Cat}(\text{softmax}(\theta))$

$\text{soft}\max(\theta) = (0.1, 0.5, 0.4)$

- Gumbel trick defines $g_\theta(\varepsilon)$ for z
  - Let $\varepsilon_i = -\log(-\log u_i)$ for $u \sim U[0,1]^d$
  - Let $g_\theta(\varepsilon) = \underset{i=1,\ldots,d}{\text{argmax}}(\theta + \varepsilon)$
  - Then $g_\theta(\varepsilon) \overset{d}{=} z$

Gumbel argmax trick

$\theta \in \mathbb{R}^d$



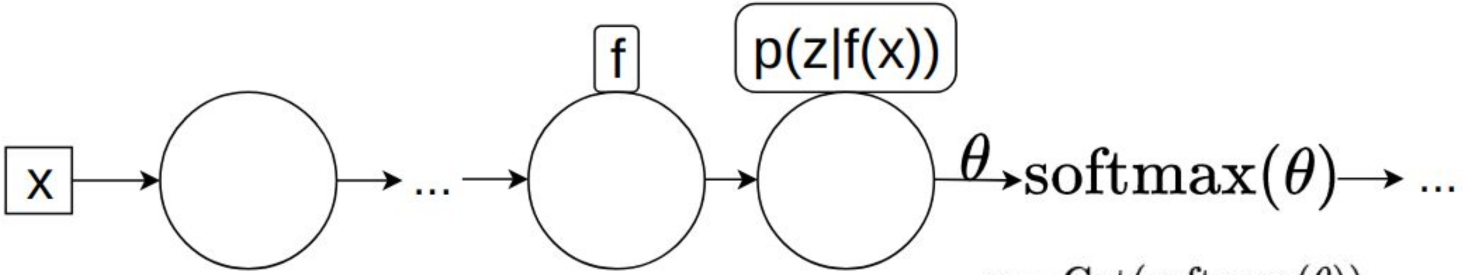
$z \sim \mathrm{Cat}(\mathrm{softmax}(\theta))$

- Gumbel trick defines $g_\theta(\varepsilon)$ for z
  - Let $\varepsilon_i = -\log(-\log u_i)$ for $u \sim U[0,1]^d$
  - Let $g_\theta(\varepsilon) = \underset{i=1,\ldots,d}{\mathrm{argmax}}(\theta + \varepsilon)$
  - Then $g_\theta(\varepsilon) \overset{d}{=} z$

$\mathrm{soft\,max}(\theta) = (0.1,0.5,0.4)$

$u = (0.4,0.3,0.7)$

Gumbel argmax trick



$\theta \in \mathbb{R}^d$

$z \sim \text{Cat}(\text{softmax}(\theta))$

- Gumbel trick defines $g_\theta(\varepsilon)$ for z
  - Let $\varepsilon_i = -\log(-\log u_i)$ for $u \sim U[0,1]^d$
  - Let $g_\theta(\varepsilon) = \underset{i=1,\ldots,d}{\text{argmax}}(\theta + \varepsilon)$
  - Then $g_\theta(\varepsilon) \overset{d}{=} z$

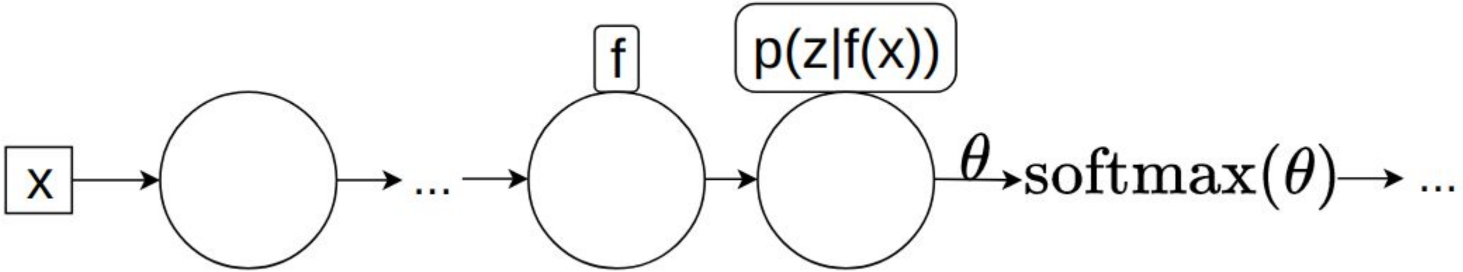
$\text{soft}\max(\theta) = (0.1,0.5,0.4)$

$u = (0.4,0.3,0.7)$

$g_\theta(\varepsilon) = \arg\max(-2.2, -0.9, +0.1)$

$= (0,0,1)$

Gumbel argmax trick

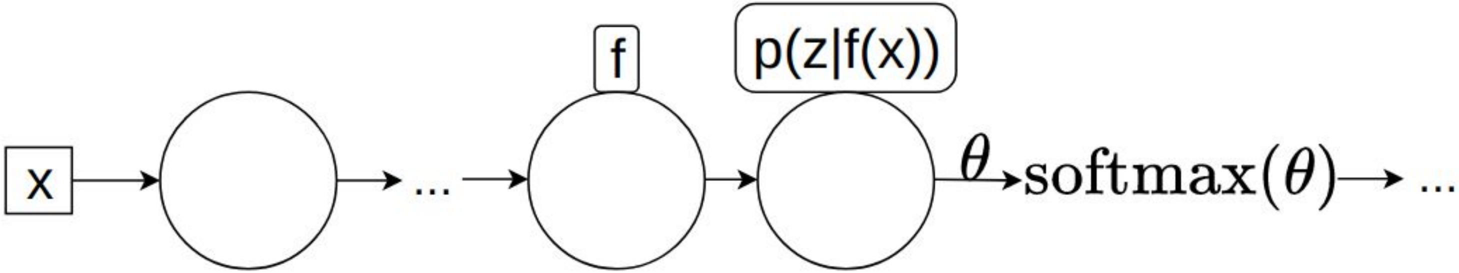


$\theta \in \mathbb{R}^d$

- Gumbel trick defines $g_\theta(\varepsilon)$ for z
  - Let $\varepsilon_i = -\log(-\log u_i)$ for $u \sim U[0,1]^d$
  - Let $g_\theta(\varepsilon) = \operatorname*{argmax}_{i=1,\ldots,d}(\theta + \varepsilon)$
  - Then $g_\theta(\varepsilon) \stackrel{d}{=} z$

$$\frac{dh}{dg}\,\frac{\partial g(\hat{\varepsilon}, f(x))}{\partial f}\,\frac{df}{dx}$$

Gumbel argmax trick



$\theta \in \mathbb{R}^d$

- Gumbel trick defines $g_\theta(\varepsilon)$ for z
  - Let $\varepsilon_i = -\log(-\log u_i)$ for $u \sim U[0,1]^d$
  - Let $g_\theta(\varepsilon) = \boxed{\operatorname*{argmax}_{i=1,\ldots,d}(\theta + \varepsilon)}$
  - Then $g_\theta(\varepsilon) \overset{d}{=} z$

$$\frac{dh}{dg}\boxed{\frac{\partial g(\hat{\varepsilon}, f(x))}{\partial f}}\frac{df}{dx}$$

Gumbel softmax trick (GST)

Gumbel softmax trick (GST)

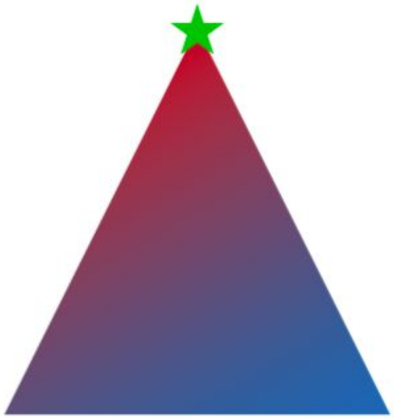
- **Idea:** replace $\arg\max(\cdot)$ with $\mathrm{soft}\max(\cdot)$

- $\mathrm{soft}\max(\frac{\theta+\varepsilon}{T}) \overset{T\to 0}{\to} \arg\max(\theta+\varepsilon)$

A different view on Argmax

- Rewrite $\underset{i=1,\ldots,d}{\operatorname{argmax}}\, w = \underset{z\in\Delta^d}{\operatorname{argmax}}\, w^T z$
  - Δ^d is a convex hull of one-hots for z

arg max :



A different view on Argmax

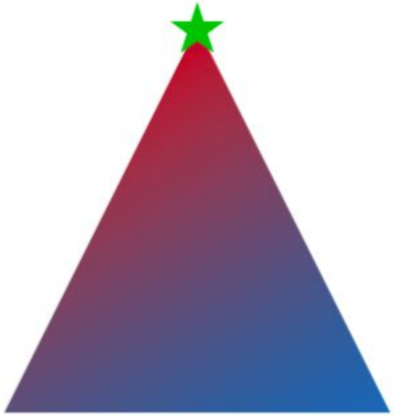
- Rewrite $\underset{i=1,\ldots,d}{\operatorname{argmax}}\, w = \underset{z\in\Delta^d}{\operatorname{argmax}}\, w^T z$

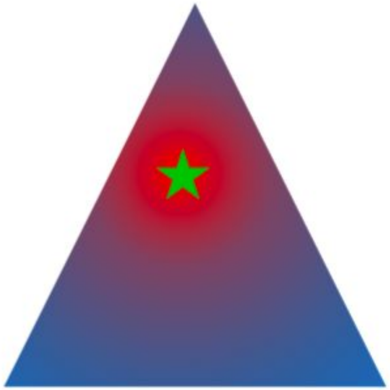
- Δ^d is a convex hull of one-hots for z

$$\underset{z\in\Delta^d}{\operatorname{argmax}}(w^T z + TH(z))$$

- $H(z) = -\sum_i z_i \log z_i$

arg max :

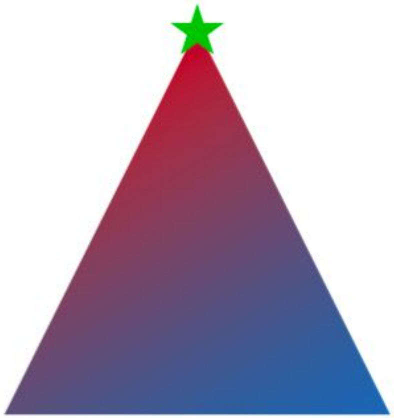




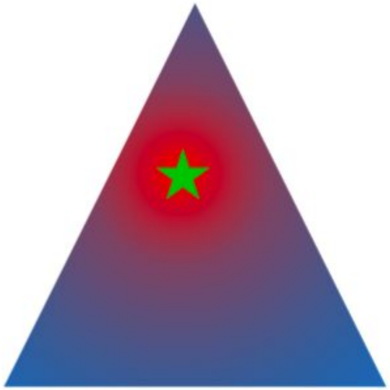
A different view on Argmax and Softmax

- Rewrite $\underset{i=1,\ldots,d}{\operatorname{argmax}}\, w = \underset{z \in \Delta^d}{\operatorname{argmax}}\, w^T z$
  - Δ^d is a convex hull of one-hots for z
- Then $\operatorname{soft\,max}(\frac{w}{T}) = \underset{z \in \Delta^d}{\operatorname{argmax}}(w^T z + TH(z))$
  - $H(z) = -\sum_i z_i \log z_i$

arg max :



soft max :



A different view on Gumbel softmax trick

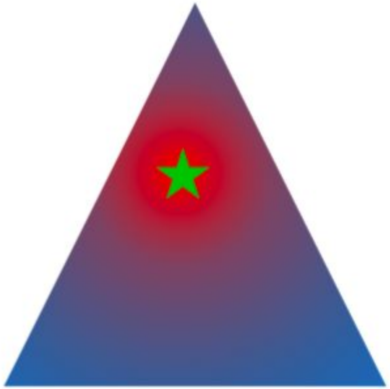
- Equivalently $z = \underset{z \in \Delta^d}{\operatorname{argmax}}((\theta + \varepsilon)^T z + TH(z))$

  1. Perturb θ with ε
  2. Find arg max

- Then $\operatorname{soft\,max}(\frac{w}{T}) = \underset{z \in \Delta^d}{\operatorname{argmax}}(w^T z + TH(z))$

- $H(z) = -\sum_i z_i \log z_i$

soft max :



The limitations of GST

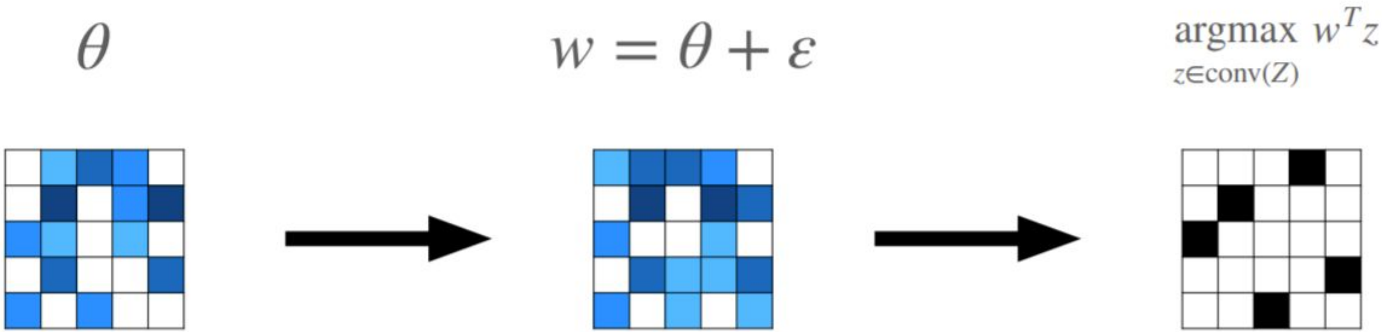
- Time is $O(d)$
  - Perturb each θ_i and find max
- For combinatorial z the support is $d \gg 1$
  - Gumbel softmax trick is too slow

Stochastic argmax trick (SMT)

	Gumbel Argmax Trick	Stochastic argmax Trick
Support	$Z = \{e_1, \ldots, e_d\} \subset \mathbb{R}^d$	$Z = \{z_1, \ldots, z_m\} \subset \mathbb{R}^d$
Perturbation	$w = \theta_i - \log(-\log(u_i)), u \sim U[0,1]^d$	$w = r_\theta(\varepsilon)$
Forward pass	$z = \underset{z' \in \operatorname{conv} Z}{\operatorname{argmax}} \, w^T z'$	$z = \underset{z' \in \operatorname{conv} Z}{\operatorname{argmax}} \, w^T z'$

Example

- $Z = \{z_1, \ldots, z_m\} \subset R^{n \times n}$ is a set of permutation matrices on n elements
- $m = n!$



Regularization & Relaxation

- Then $\text{soft}\max(\frac{w}{T}) = \underset{z \in \Delta^d}{\text{argmax}}(w^T z + TH(z))$
- $H(z) = -\sum_i z_i \log z_i$

Regularization & Relaxation

Stochastic *Argmax* → Stochastic *Soft*max

- Add a strongly convex regularizer $f : \mathbb{R}^d \rightarrow \{\mathbb{R}, \inf\}$

$$z = \underset{z' \in \mathrm{conv}(Z)}{\operatorname{argmax}} \; w^T z' \quad \rightarrow \quad z_T = \underset{z' \in \mathrm{conv}(Z)}{\operatorname{argmax}} \; (w^T z' - Tf(z'))$$

- Then $\operatorname{soft\,max}(\frac{w}{T}) = \underset{z \in \Delta^d}{\operatorname{argmax}}(w^T z + TH(z))$

- $H(z) = -\sum_i z_i \log z_i$

Regularization & Relaxation

Stochastic *Argmax* → Stochastic *Soft*max

- Add a strongly convex regularizer $f : \mathbb{R}^d \rightarrow \{\mathbb{R}, \inf\}$

$$z = \underset{z' \in \mathrm{conv}(Z)}{\mathrm{argmax}}\ w^T z' \quad \rightarrow \quad z_T = \underset{z' \in \mathrm{conv}(Z)}{\mathrm{argmax}}\ (w^T z' - Tf(z'))$$

Prop 1. If z is a.s. unique, then $\lim_{T \to 0} z_T = z$

- Then $\mathrm{soft\,max}(\frac{w}{T}) = \underset{z \in \Delta^d}{\mathrm{argmax}}(w^T z + TH(z))$
- $H(z) = -\sum_i z_i \log z_i$

Regularization & Relaxation

- Then $\text{soft max}(\frac{w}{T}) = \underset{z \in \Delta^d}{\text{argmax}}(w^T z + TH(z))$

- $H(z) = -\sum_i z_i \log z_i$

Stochastic *Argmax* → Stochastic *Soft*max

- Add a strongly convex regularizer $f : \mathbb{R}^d \to \{\mathbb{R}, \inf\}$

$$z = \underset{z' \in \text{conv}(Z)}{\text{argmax}} \; w^T z' \quad \rightarrow \quad z_T = \underset{z' \in \text{conv}(Z)}{\text{argmax}} \; (w^T z' - Tf(z'))$$

Prop 1. If z is a.s. unique, then $\lim_{T \to 0} z_T = z$

Prop 2. z_T exists, is unique and differentiable in w

Implementation requirements

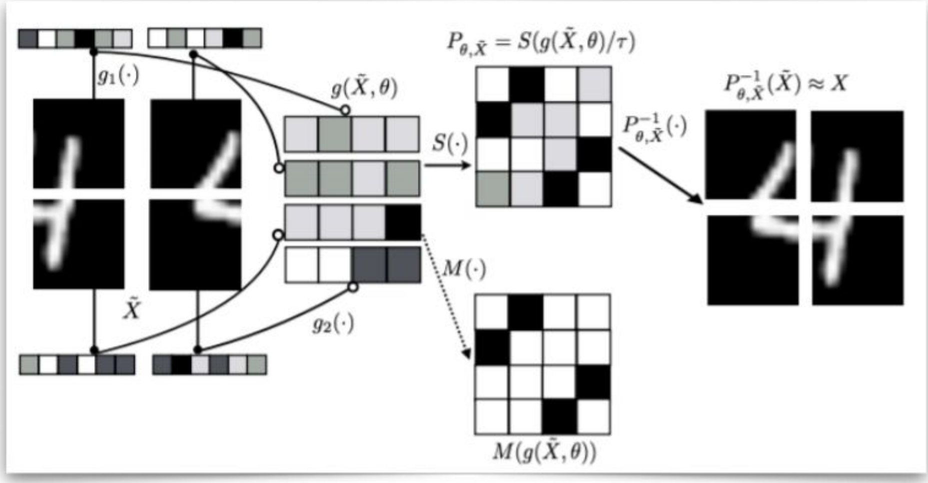
- Inference
  - Reparametrized r.v. $w = r_\theta(\varepsilon)$
  - Solver for $\operatorname*{argmax}_{z \in \operatorname{conv} Z} w^T z$
- Training
  - Strongly convex regularizer $f(z)$
  - Solver for $\operatorname*{argmax}_{z \in \operatorname{conv} Z} (w^T z - tf(z))$

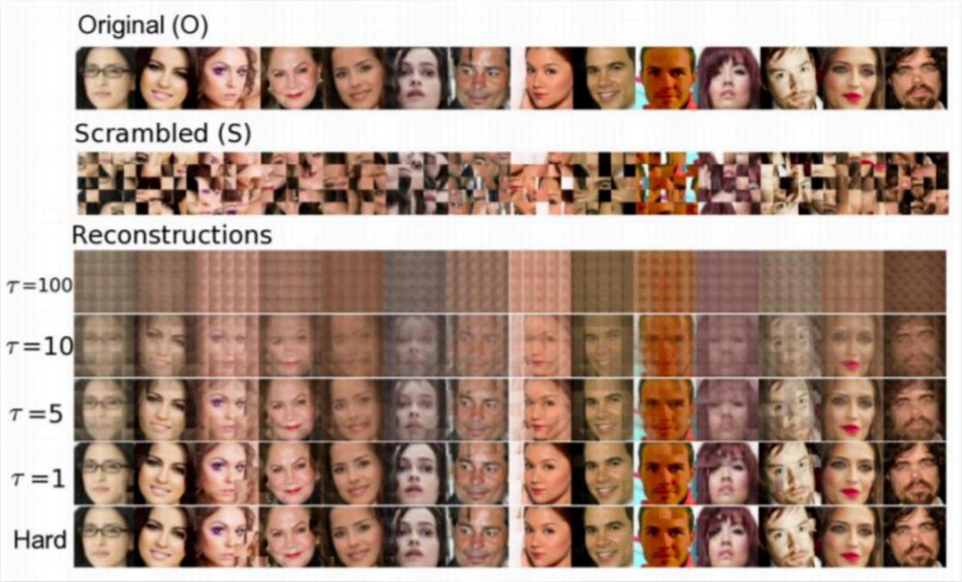
1) Gumbel Sinkhorn

- Take permutation matrices as Z
  - Then $\mathrm{conv}(Z)$ consists of doubly-stochastic
- **Hungarian algorithm** solves $\arg\max w^T z$
- Entropy $f(z) = \sum_{i,j} z_{i,j} \log z_{i,j}$
- **Sinkhorn algorithm** finds $\arg\max(w^T z - T \cdot f(z))$

Mena G. et al. Learning latent permutations with gumbel-sinkhorn networks

Finding latent permutations: Jigsaw





2) K-subset selection

- $Z = \{z \in \{0,1\}^d \mid \sum z_i = k\}$

- Sort to solve $\arg\max w^T z$

- $Z = \{z \in \{0,1\}^{2d-1} \mid \sum_{i=1}^{n} z_i = k, z_i = z_{i-d} z_{i-d+1}$ **for** $d < i < 2d - 1\}$
- Dynamic programming for $\arg\max$

- Exponential family relaxation

BeerAdvocate Interpretability

Pours a slight tangerine orange and straw yellow. The head is nice and bubbly but fades very quickly with a little lacing. Smells like Wheat and European hops, a little yeast in there too. There is some fruit in there too, but you have to take a good whiff to get it. The taste is of wheat, a bit of malt, and a little fruit flavour in there too. Almost feels like drinking Champagne, medium mouthful otherwise. Easy to drink, but not something I'd be trying every night.

Appearance: 3.5 Aroma: 4.0 Palate: 4.5 Taste: 4.0 Overall: 4.0

		$k=5$		$k=10$		$k=15$	
Model	Relaxation	MSE	Subs. Prec.	MSE	Subs. Prec.	MSE	Subs. Prec.
Simple	*L2X* [17]	3.6 ± 0.1	28.3 ± 1.7	3.0 ± 0.1	25.5 ± 1.2	2.6 ± 0.1	25.5 ± 0.4
	SoftSub [84]	3.6 ± 0.1	27.2 ± 0.7	3.0 ± 0.1	26.1 ± 1.1	2.6 ± 0.1	25.1 ± 1.0
	Euclid. Top k	3.5 ± 0.1	25.8 ± 0.8	2.8 ± 0.1	32.9 ± 1.2	2.5 ± 0.1	29.0 ± 0.3
	Cat. Ent. Top k	3.5 ± 0.1	26.4 ± 2.0	2.9 ± 0.1	32.1 ± 0.4	2.6 ± 0.1	28.7 ± 0.5
	Bin. Ent. Top k	3.5 ± 0.1	29.2 ± 2.0	2.7 ± 0.1	33.6 ± 0.6	2.6 ± 0.1	28.8 ± 0.4
	E.F. Ent. Top k	3.5 ± 0.1	28.8 ± 1.7	2.7 ± 0.1	32.8 ± 0.5	2.5 ± 0.1	29.2 ± 0.8
	Corr. Top k	$\mathbf{2.9 \pm 0.1}$	$\mathbf{63.1 \pm 5.3}$	$\mathbf{2.5 \pm 0.1}$	$\mathbf{53.1 \pm 0.9}$	$\mathbf{2.4 \pm 0.1}$	$\mathbf{45.5 \pm 2.7}$

Conclusions

- Learning structured latent variables is an active research direction
- *Stochastic softmax trick* generalize *gumbel softmax trick*
- There is more to be done

Questions

1) Write down either REINFORCEMENT or reparameterization trick final formula

2) Rewrite softmax function as it was discussed (using convex hull)

3) List implementation requirements for SST framework