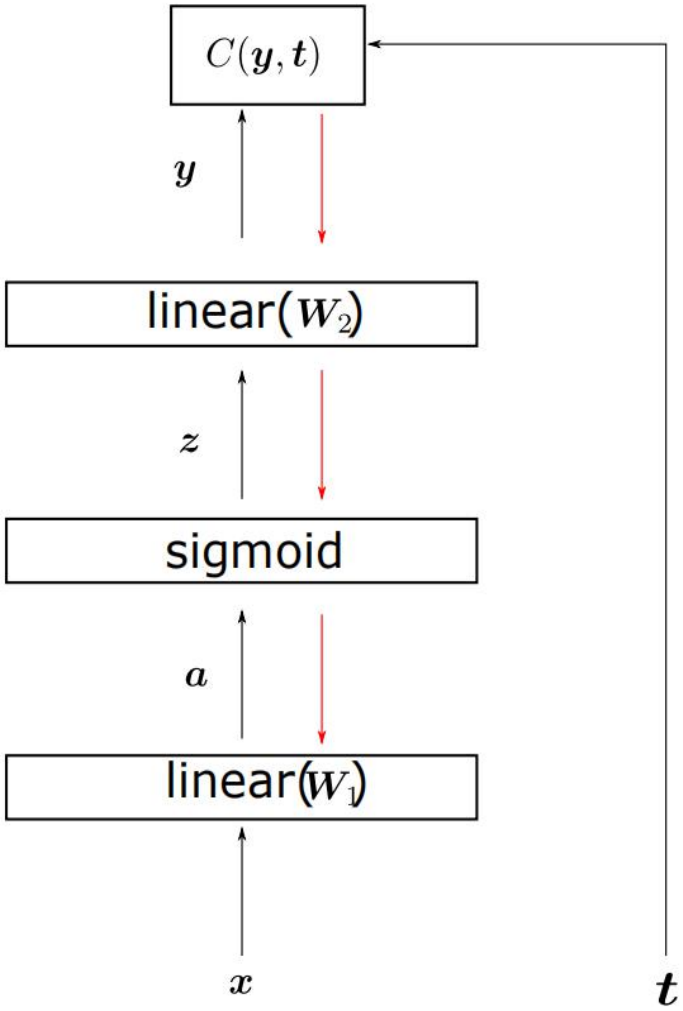
Reparametric Trick

Dong Wang

2021/01/11

Revisit BP

- BP propagate loss information back-ward.



$$\frac{\partial C}{\partial \boldsymbol{W}_2} = \frac{\partial C}{\partial \boldsymbol{y}} \frac{\partial \boldsymbol{y}}{\partial \boldsymbol{W}_2} = \frac{\partial C}{\partial \boldsymbol{y}} \boldsymbol{z}^T$$

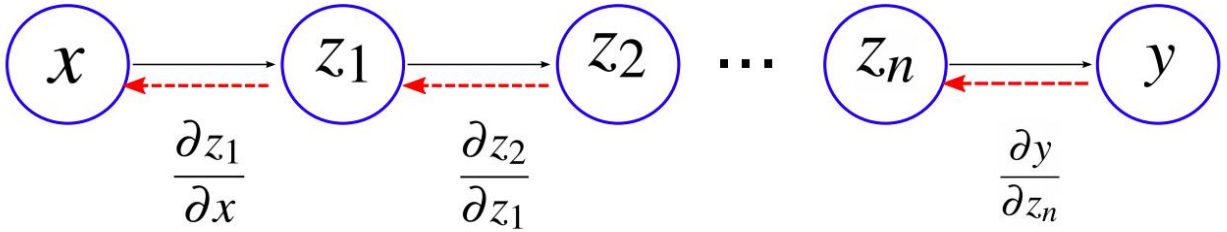
$$\frac{\partial C}{\partial \boldsymbol{z}} = ((\frac{\partial C}{\partial \boldsymbol{y}})^T \frac{\partial \boldsymbol{y}}{\partial \boldsymbol{z}})^T = ((\frac{\partial C}{\partial \boldsymbol{y}})^T \boldsymbol{W}_2)^T$$

$$\frac{\partial C}{\partial \boldsymbol{a}} = ((\frac{\partial C}{\partial \boldsymbol{z}})^T \frac{\partial \boldsymbol{z}}{\partial \boldsymbol{a}})^T = \frac{\partial C}{\partial \boldsymbol{z}} \odot \boldsymbol{z} \odot (1 - \boldsymbol{z})$$

$$\frac{\partial C}{\partial \boldsymbol{W}_1} = \frac{\partial C}{\partial \boldsymbol{a}} \frac{\partial \boldsymbol{a}}{\partial \boldsymbol{W}_1} = \frac{\partial C}{\partial \boldsymbol{a}} \boldsymbol{x}^T$$

Revisit BP

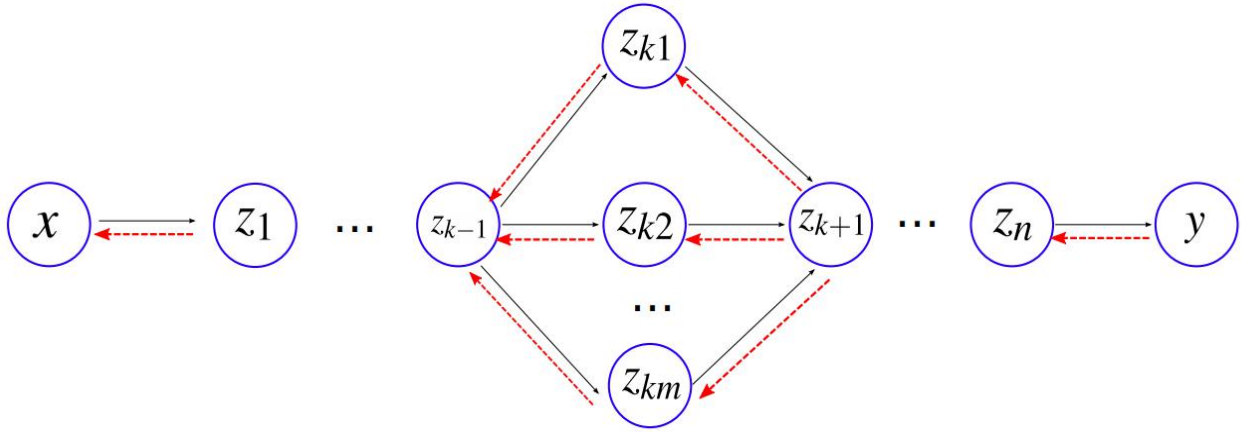
- BP relies on the chain rule of derivation



$$\frac{\partial y}{\partial x} = \frac{\partial y}{\partial z_n}\frac{\partial z_n}{\partial z_{n-1}}\cdots\frac{\partial z_2}{\partial z_1}\frac{\partial z_1}{\partial x}$$

Revisit BP

- All the paths traversing through the target parameters should be involved

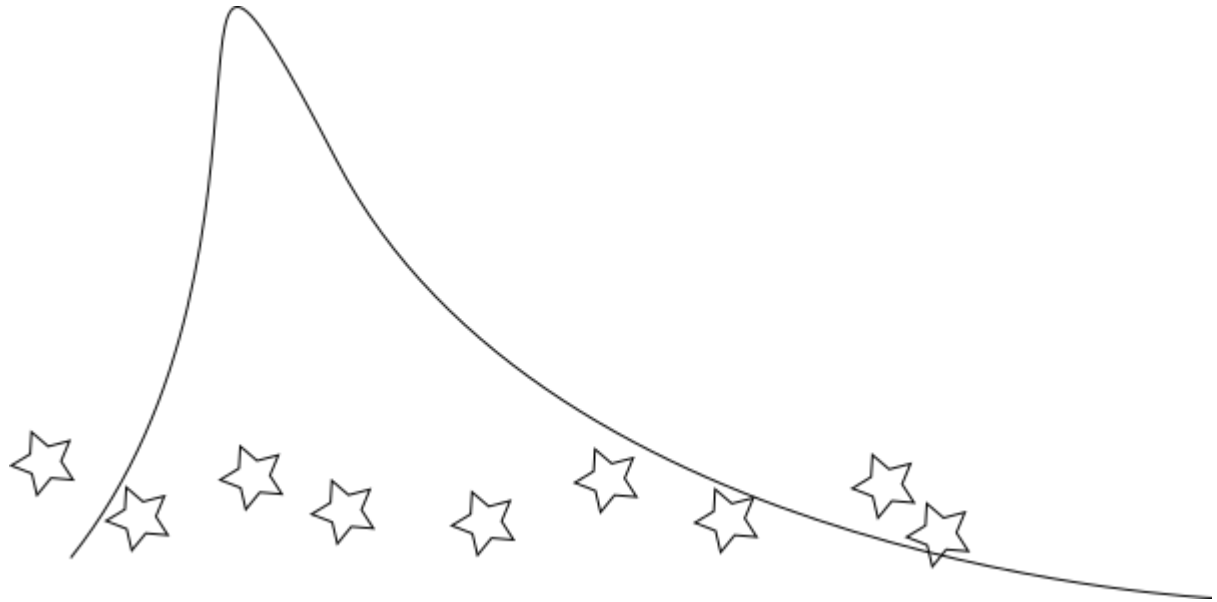


$$\frac{\partial y}{\partial x} = \frac{\partial y}{\partial z_{k+1}} \frac{\partial z_{k+1}}{\partial z_{k-1}} \frac{\partial z_{k-1}}{\partial x}$$

$$= \left\{ \frac{\partial y}{\partial z_n} \cdots \frac{\partial z_{k+2}}{\partial z_{k+1}} \right\} \left\{ \sum_{i=1}^{m} \frac{\partial z_{k+1}}{z_{ki}} \frac{\partial z_{ki}}{z_{k-1}} \right\} \left\{ \frac{\partial z_{k-1}}{\partial z_{k-2}} \cdots \frac{\partial z_1}{\partial x} \right\}$$

Consider a simple task

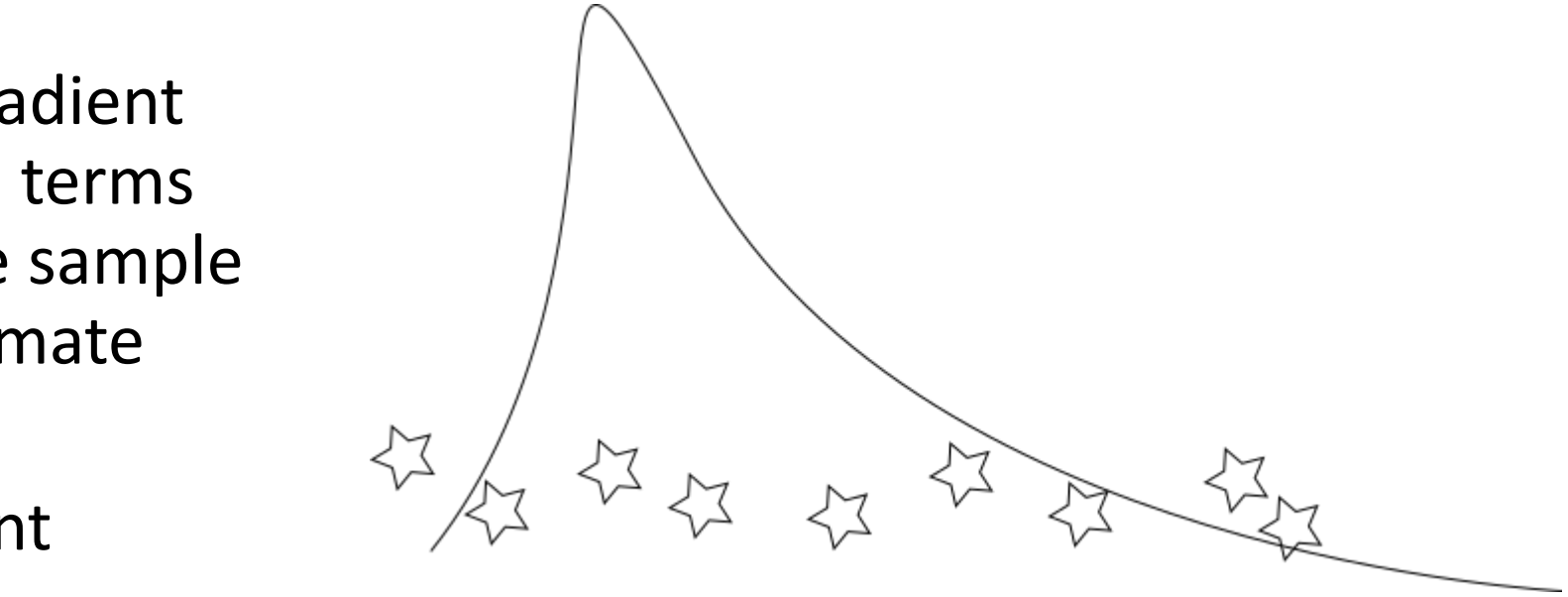
- Goal: optimize an intractable integration samples, with respect to the distribution.



$$C(\theta) = \int p_\theta(x) f(x) \mathrm{d}x \approx 1/N \sum_{n=1}^{N} f(x_n)$$

REINFORCE

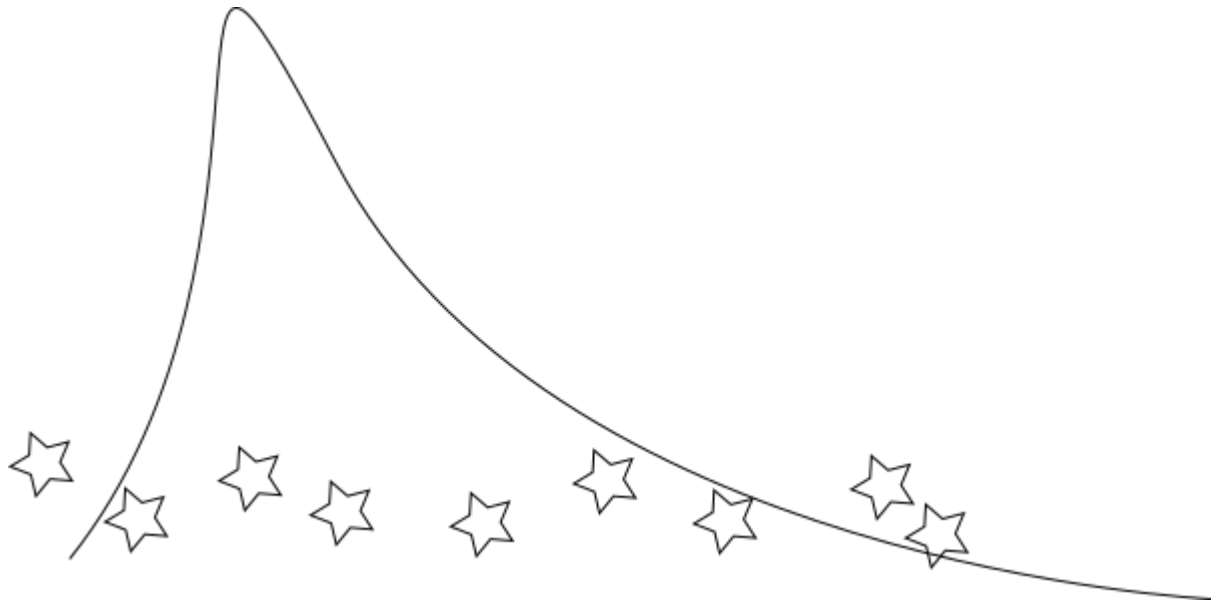
- Reformulate the gradient as an integration in terms of $p_\theta(x)$, so that the sample can be used to estimate the graident.
- Lose part of gradient represented by the samples themselves.



$$\frac{\partial C(\theta)}{\partial \theta} = \int p_\theta(x) \frac{\partial \ln p_\theta(x)}{\partial \theta} f(x) \mathrm{d}x \approx 1/N \sum_{n=1}^{N} \frac{\partial \ln p_\theta(x)}{\partial \theta} f(x_n)$$

Parameteric trick

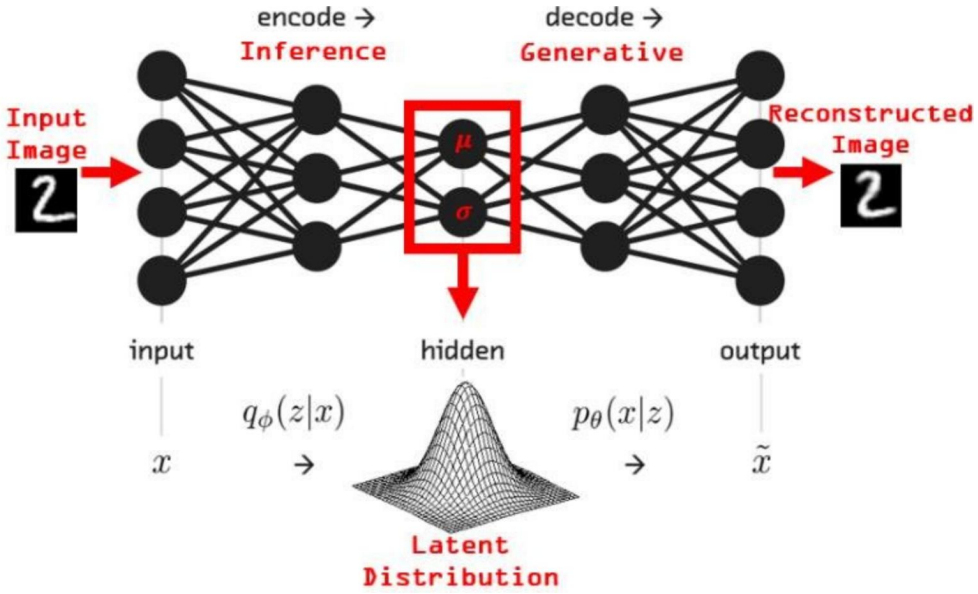
- Represent $p_\theta(x)$ as $k_\varsigma(\varepsilon)$ where ε is a standard distribution.
- All randomness is represented by ε, and learning ς will learn $p_\theta(x)$.



$$\frac{\partial C(\xi)}{\partial \xi} = \frac{1/N \sum_{n=1}^{N} \partial f(k_\xi(\varepsilon_n))}{\partial \xi}$$

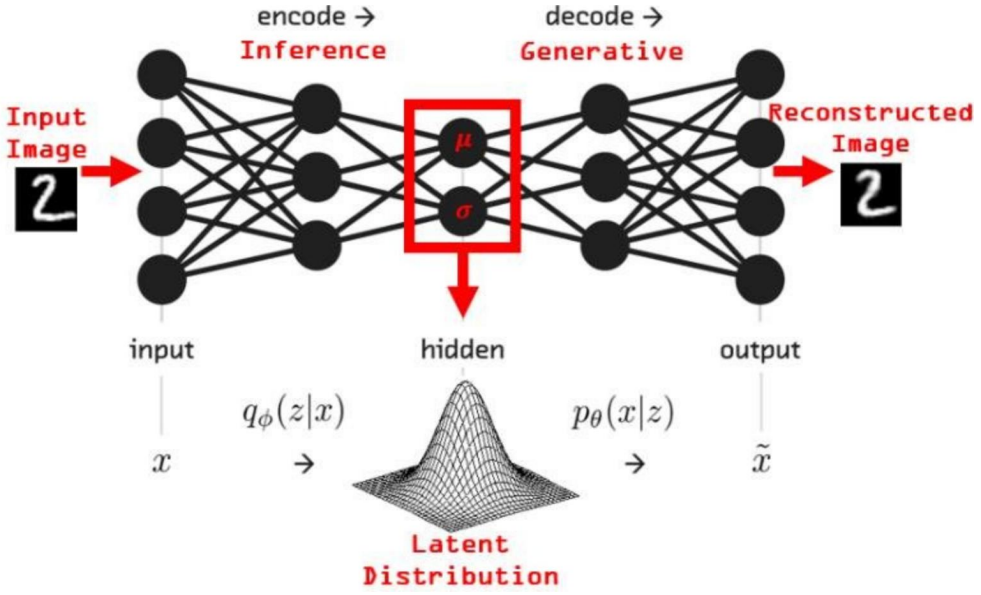
Reparameteric trick in VAE

- VAE: an encoder $g_\varphi(x)$ and a decoder f(z), trained simultaneously
- By sampling of z, gradient is blocked when backpropagated to the encoder



Reparameteric trick in VAE

- Using reparaemtric trick, let ε is a Gaussian, $z=k_{\varsigma}(\varepsilon) = \mu + \sigma \varepsilon$; $\varsigma=\{\mu,\sigma\}$
- Due to this trick, graident can BP to μ and σ, which further BP to the encoder.



$$C(\mu,\sigma) = 1/N \sum_n L(f(\mu+\sigma\varepsilon_n),x) + KL(N(0,1),N(\mu,\sigma))$$

Reparametric for Bayes neural net

- Using reparametric to represent model

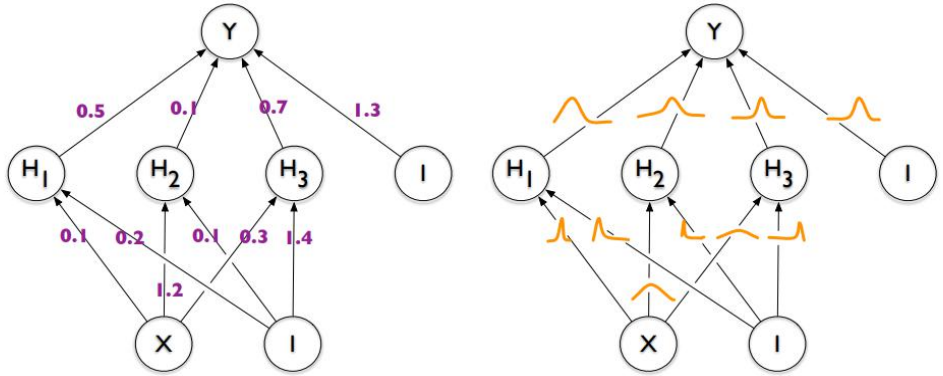


Figure 1. Left: each weight has a fixed value, as provided by classical backpropagation. Right: each weight is assigned a distribution, as provided by Bayes by Backprop.

1. Sample $\epsilon \sim \mathcal{N}(0, I)$.
2. Let $\mathbf{w} = \mu + \log(1 + \exp(\rho)) \circ \epsilon$.
3. Let $\theta = (\mu, \rho)$.
4. Let $f(\mathbf{w}, \theta) = \log q(\mathbf{w}|\theta) - \log P(\mathbf{w})P(\mathcal{D}|\mathbf{w})$.
5. Calculate the gradient with respect to the mean

$$\Delta_\mu = \frac{\partial f(\mathbf{w}, \theta)}{\partial \mathbf{w}} + \frac{\partial f(\mathbf{w}, \theta)}{\partial \mu}. \quad (3)$$

6. Calculate the gradient with respect to the standard deviation parameter ρ

$$\Delta_\rho = \frac{\partial f(\mathbf{w}, \theta)}{\partial \mathbf{w}} \frac{\epsilon}{1 + \exp(-\rho)} + \frac{\partial f(\mathbf{w}, \theta)}{\partial \rho}. \quad (4)$$

7. Update the variational parameters:

$$\mu \leftarrow \mu - \alpha\Delta_\mu \quad (5)$$
$$\rho \leftarrow \rho - \alpha\Delta_\rho. \quad (6)$$

Blundell et al., Weight Uncertainty in Neural Networks, ICML 2015

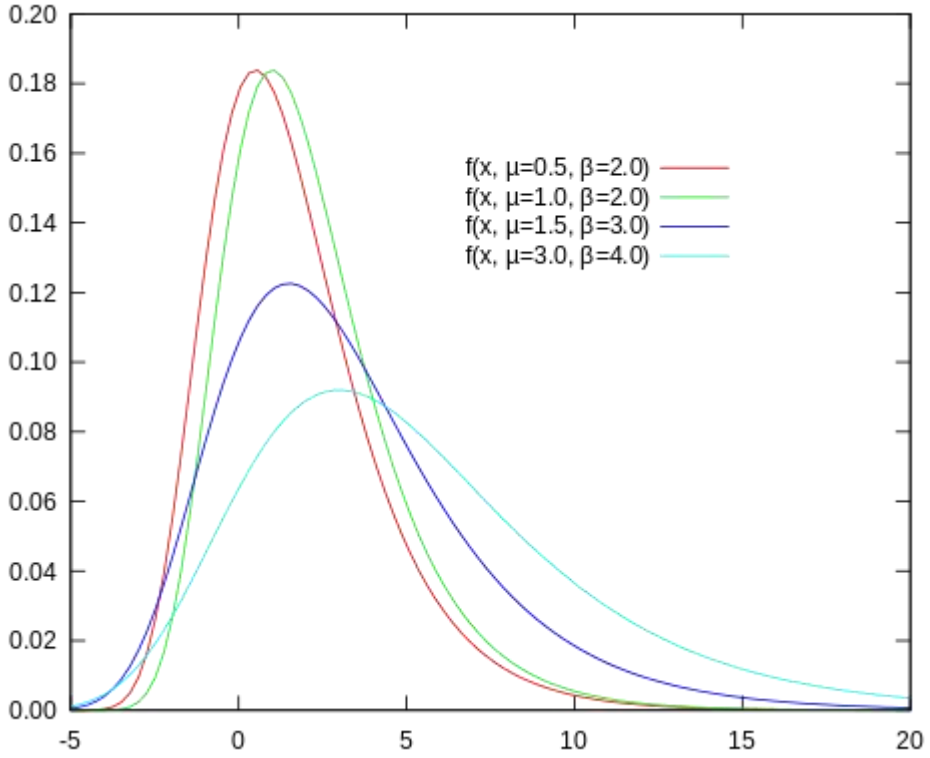
Appliation to speaker recognition

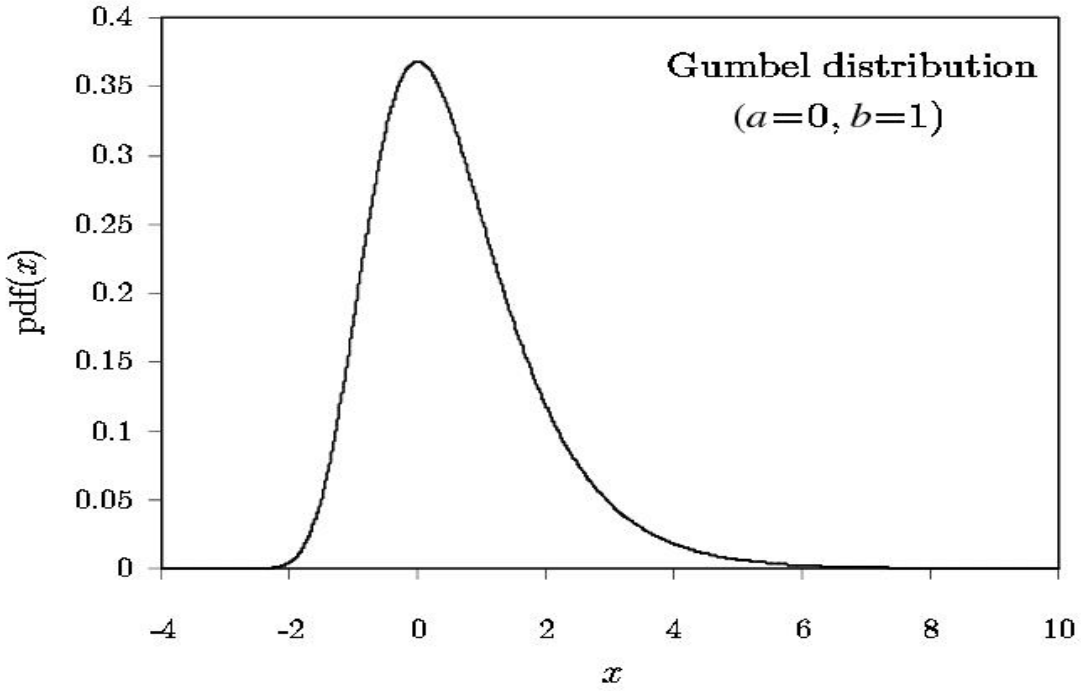
Table 2: In-domain evaluation: equal error rate (EER) and minimum detection cost function (min-DCF) in different conditions.

Training set	Evaluation set	System	Scoring back-end	x-vector extractor	EER(%)	DCF_{VOX}/DCF_{SRE10}
Voxceleb1	Voxceleb1	(1)	cosine	baseline	9.58	0.6899
		(2)		proposed	9.30	0.6508
		(3)		fusion	**8.64**	**0.6423**
		(4)	PLDA	baseline	6.68	0.6023
		(5)		proposed	6.52	**0.5423**
		(6)		fusion	**6.35**	0.5487
NIST SRE10	NIST SRE10	(7)	cosine	baseline	5.61	0.6830
		(8)		proposed	5.52	0.6555
		(9)		fusion	**5.47**	**0.6502**
		(10)	PLDA	baseline	3.29	0.3926
		(11)		proposed	3.19	**0.3835**
		(12)		fusion	**3.17**	0.3840

Li et al., Bayesian x-vector: Bayesian Neural Network based x-vector System for Speaker Verification

Reparametric for categorical values by Gumbel distribution





The Gumbel(0,1)distribution can be sampled using inverse transform sampling by drawing
u~ Uniform(0,1)and computing g=−log(−log(u))

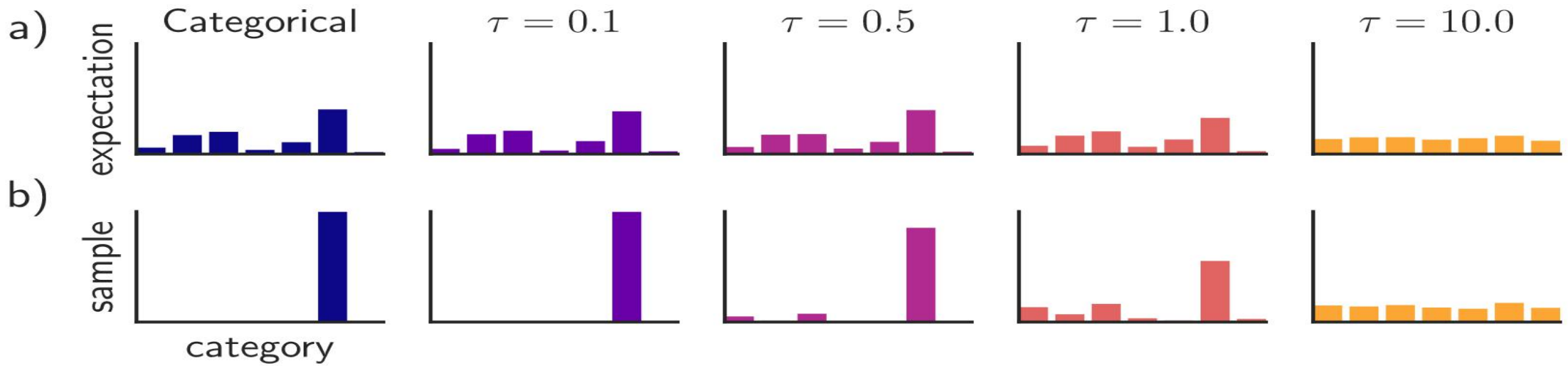
Gumbel-Max trick

$$z = \texttt{one\_hot}\left(\arg\max_i \left[g_i + \log \pi_i\right]\right)$$

- Gumbel-Max trick: draw k (k is the class number) g_i following Gumbel (0,1), z will be a unbiased sample following categorical distribution with parameter π

Jang E, Gu S, Poole B. Categorical reparameterization with gumbel-softmax[J]. arXiv preprint arXiv:1611.01144, 2016.

Smooth the transform function:Gumbel-softmax



$$y_i = \frac{\exp((\log(\pi_i) + g_i)/\tau)}{\sum_{j=1}^{k} \exp((\log(\pi_j) + g_j)/\tau)} \qquad \text{for } i = 1, ..., k$$

More on sampling

- Gumbel-softmax sample continuous vectors y
- Using argmax (y) =z as the sample, and treat the gradient on y is equal to the gradient on z, called STRAIGHT-THROUGH estimation.

Jang E, Gu S, Poole B. Categorical reparameterization with gumbel-softmax[J]. arXiv preprint arXiv:1611.01144, 2016.

Understand Gumbel-softmax

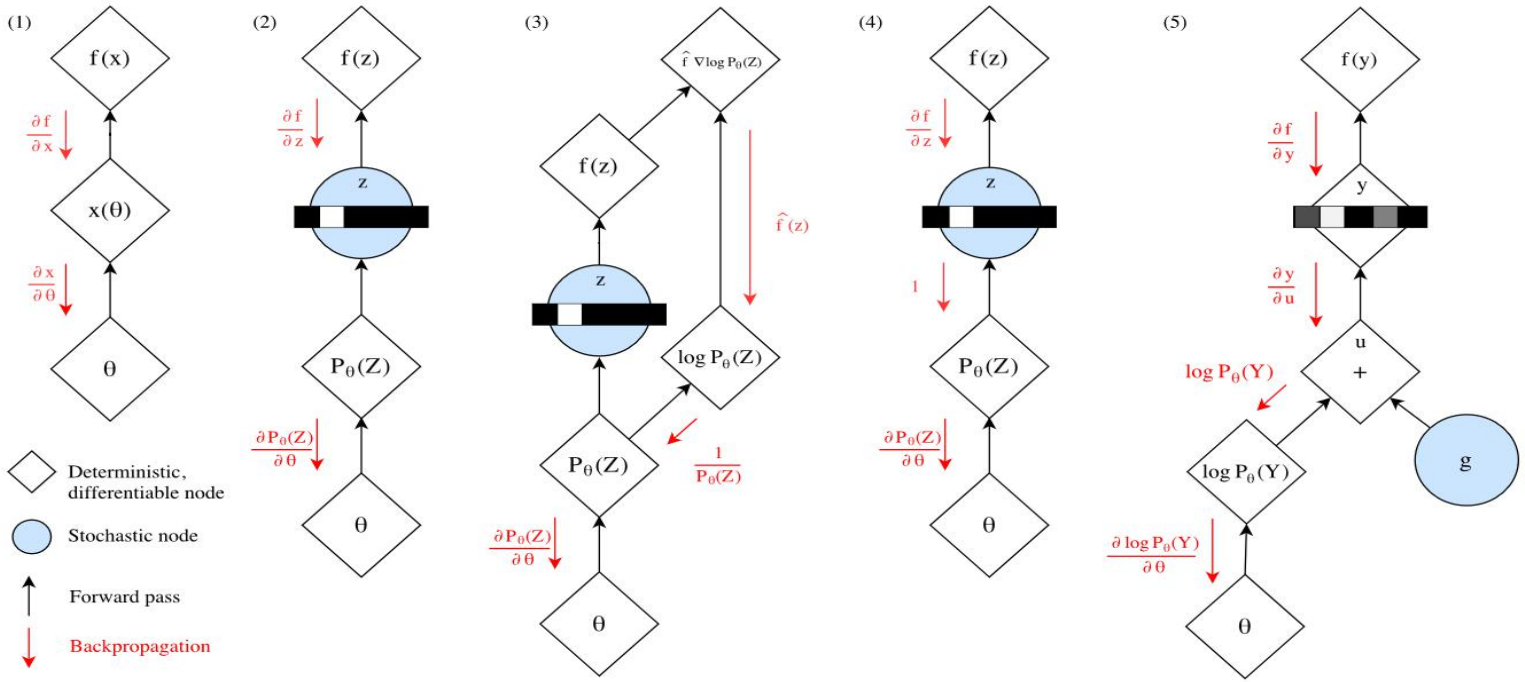


Figure 2: Gradient estimation in stochastic computation graphs. (1) $\nabla_\theta f(x)$ can be computed via backpropagation if $x(\theta)$ is deterministic and differentiable. (2) The presence of stochastic node z precludes backpropagation as the sampler function does not have a well-defined gradient. (3) The score function estimator and its variants (NVIL, DARN, MuProp, VIMCO) obtain an unbiased estimate of $\nabla_\theta f(x)$ by backpropagating along a surrogate loss $\hat{f} \log p_\theta(z)$, where $\hat{f} = f(x) - b$ and b is a baseline for variance reduction. (4) The Straight-Through estimator, developed primarily for Bernoulli variables, approximates $\nabla_\theta z \approx 1$. (5) Gumbel-Softmax is a path derivative estimator for a continuous distribution y that approximates z. Reparameterization allows gradients to flow from $f(y)$ to θ. y can be annealed to one-hot categorical variables over the course of training.

Wrap up

- Sampling is a powerful approach to deal with complex integration, however it will block the gradient path
- Reparametric trick reformulates the distribution as a transform of a basic distribution, so that the distribution itself can be learned.
- Gaussian trick is often used in continuous cases, and Gumbel-softmax can be used in categorical cases.