

Efficient Marginalization of Discrete and Structured Latent Variables via Sparsity

Gonçalo Correia Instituto de Telecomunicações, Lisbon

Vlad Niculae Ivl, University of Amsterdam

Wilker Aziz ILLC, University of Amsterdam

André Martins Instituto de Telecomunicações & LUMLIS & Unbabel

Latent Variable Models

Latent variable z can be

Latent Variable Models

Latent variable z can be continuous



Source: Bouges et al., 2013

Latent Variable Models

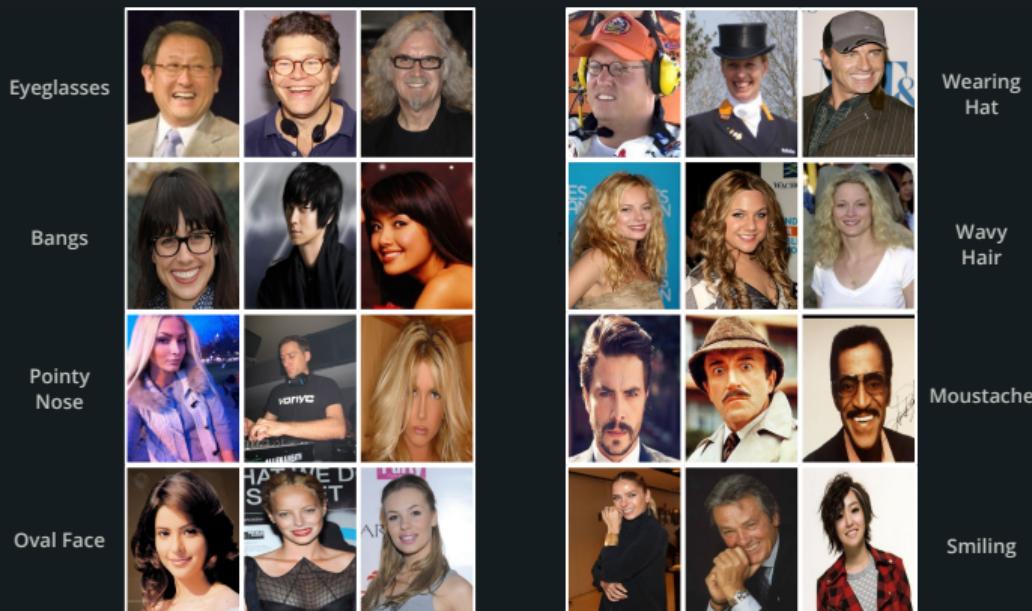
Latent variable z can be **continuous**, **discrete**



Source: valleyeyecareaz.com

Latent Variable Models

Latent variable z can be continuous, discrete, or structured



Source: Liu et al., 2015

Training Discrete or Structured Latent Variable Models

Latent variable z can be

Training Discrete or Structured Latent Variable Models

Latent variable z can be discrete



Training Discrete or Structured Latent Variable Models

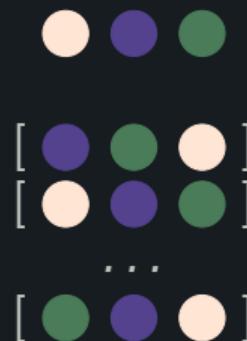
Latent variable z can be **discrete** or **structured**



Training Discrete or Structured Latent Variable Models

Latent variable z can be **discrete** or **structured**

$\pi(z|x, \theta)$: distribution over possible z



Training Discrete or Structured Latent Variable Models

Latent variable z can be **discrete** or **structured**

$\pi(z|x, \theta)$: distribution over possible z



Training Discrete or Structured Latent Variable Models

Latent variable z can be **discrete** or **structured**

$\pi(z|x, \theta)$: distribution over possible z

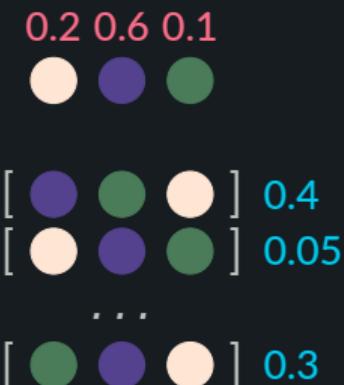


Training Discrete or Structured Latent Variable Models

Latent variable z can be **discrete** or **structured**

$\pi(z|x, \theta)$: distribution over possible z

$\ell(x, z; \theta)$: downstream loss: ELBO, Log-Likelihood, (...)



Training Discrete or Structured Latent Variable Models

Latent variable z can be **discrete** or **structured**

$\pi(z|x, \theta)$: distribution over possible z

$\ell(x, z; \theta)$: downstream loss: ELBO, Log-Likelihood, (...)

To train, we need to compute the following expectation:



Training Discrete or Structured Latent Variable Models

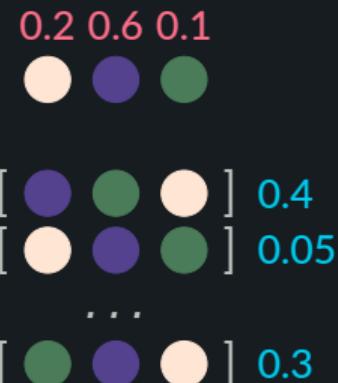
Latent variable z can be **discrete** or **structured**

$\pi(z|x, \theta)$: distribution over possible z

$\ell(x, z; \theta)$: downstream loss: ELBO, Log-Likelihood, (...)

To train, we need to compute the following expectation:

$$\mathcal{L}_x(\theta) = \sum_{z \in \mathcal{Z}} \pi(z|x, \theta) \ell(x, z; \theta)$$

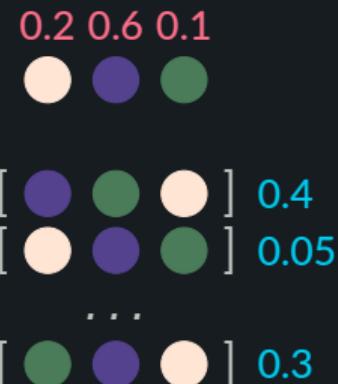


Training Discrete or Structured Latent Variable Models

Latent variable z can be **discrete** or **structured**

$\pi(z|x, \theta)$: distribution over possible z

$\ell(x, z; \theta)$: downstream loss: ELBO, Log-Likelihood, (...)



To train, we need to compute the following expectation:

$$\mathcal{L}_x(\theta) = \sum_{z \in \mathcal{Z}} \pi(z|x, \theta) \ell(x, z; \theta)$$

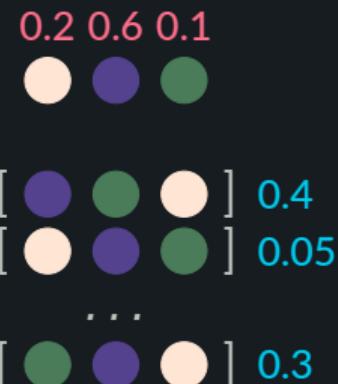
If \mathcal{Z} is **large**, this sum can get very expensive due to $\ell(x, z; \theta)$! 🍺

Training Discrete or Structured Latent Variable Models

Latent variable z can be **discrete** or **structured**

$\pi(z|x, \theta)$: distribution over possible z

$\ell(x, z; \theta)$: downstream loss: ELBO, Log-Likelihood, (...)



To train, we need to compute the following expectation:

$$\mathcal{L}_x(\theta) = \sum_{z \in \mathcal{Z}} \pi(z|x, \theta) \ell(x, z; \theta)$$

If \mathcal{Z} is **combinatorial**, this can be intractable to compute!



Current Solutions

If \mathcal{Z} is large, exact gradient computation is prohibitive

Current Solutions

If \mathcal{Z} is large, exact gradient computation is prohibitive

One option: SFE (aka REINFORCE) → unbiased but high variance

Current Solutions

If \mathcal{Z} is large, exact gradient computation is prohibitive

One option: SFE (aka REINFORCE) → unbiased but high variance

Another option: Gumbel-Softmax → continuous relaxation, biased estimation

Current Solutions

If \mathcal{Z} is large, exact gradient computation is prohibitive

One option: SFE (aka REINFORCE) → unbiased but high variance

Another option: Gumbel-Softmax → continuous relaxation, biased estimation

New option: use sparsity! 

Current Solutions

If \mathcal{Z} is large, exact gradient computation is prohibitive

One option: SFE (aka REINFORCE) → unbiased but high variance

Another option: Gumbel-Softmax → continuous relaxation, biased estimation

New option: use sparsity! 

no need for sampling → no variance

Current Solutions

If \mathcal{Z} is large, exact gradient computation is prohibitive

One option: SFE (aka REINFORCE) → unbiased but high variance

Another option: Gumbel-Softmax → continuous relaxation, biased estimation

New option: use sparsity! 

no need for sampling → no variance

no relaxation into the continuous space

Taking a step back...

Does the expectation over possible z need to be expensive?

Taking a step back...

Does the expectation over possible z need to be expensive?

$$\begin{aligned}\mathcal{L}_x(\theta) &= \sum_{z \in \mathcal{Z}} \pi(z|x, \theta) \ell(x, z; \theta) \\ &= \pi(z_1|x, \theta) \ell(x, z_1; \theta) + \pi(z_2|x, \theta) \ell(x, z_2; \theta) + \dots \\ &\quad + \pi(z_i|x, \theta) \ell(x, z_i; \theta) + \dots + \pi(z_N|x, \theta) \ell(x, z_N; \theta)\end{aligned}$$

Taking a step back...

Does the expectation over possible z need to be expensive?

$$\begin{aligned}\mathcal{L}_x(\theta) &= \sum_{z \in \mathcal{Z}} \pi(z|x, \theta) \ell(x, z; \theta) \\ &= \pi(z_1|x, \theta) \ell(x, z_1; \theta) + \pi(z_2|x, \theta) \ell(x, z_2; \theta) + \dots \\ &\quad + \pi(z_i|x, \theta) \ell(x, z_i; \theta) + \dots + \pi(z_N|x, \theta) \ell(x, z_N; \theta)\end{aligned}$$

Usually we normalize π with softmax $\propto \exp(s) \Rightarrow \pi(z_i|x, \theta) > 0$

Sparse normalizers

We use **sparsemax**, **top- k sparsemax** and **SparseMAP** to allow efficient marginalization

Sparse normalizers

We use **sparsemax**, **top- k sparsemax** and **SparseMAP** to allow efficient marginalization

These functions are able to assign **probabilities of exactly zero!**

Sparse normalizers

We use **sparsemax**, **top- k sparsemax** and **SparseMAP** to allow efficient marginalization

These functions are able to assign **probabilities of exactly zero!**

$$\begin{aligned}\mathcal{L}_x(\theta) &= \sum_{z \in \mathcal{Z}} \pi(z|x, \theta) \ell(x, z; \theta) \\ &= \pi(z_1|x, \theta) \ell(x, z_1; \theta) + \underbrace{\pi(z_2|x, \theta)}_{=0} \ell(x, z_2; \theta) + \dots \\ &\quad + \pi(z_i|x, \theta) \ell(x, z_i; \theta) + \dots + \underbrace{\pi(z_N|x, \theta)}_{=0} \ell(x, z_N; \theta)\end{aligned}$$

Sparse normalizers

We use **sparsemax**, **top- k sparsemax** and **SparseMAP** to allow efficient marginalization

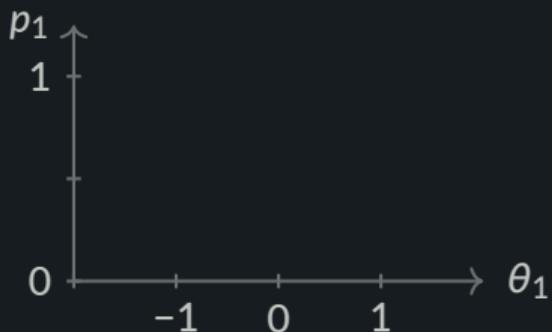
These functions are able to assign **probabilities of exactly zero!**

$$\begin{aligned}
 \mathcal{L}_x(\theta) &= \sum_{z \in \mathcal{Z}} \pi(z|x, \theta) \ell(x, z; \theta) \\
 &= \pi(z_1|x, \theta) \ell(x, z_1; \theta) + \underbrace{\pi(z_2|x, \theta) \ell(x, z_2; \theta)}_{=0} + \dots \\
 &\quad + \pi(z_i|x, \theta) \ell(x, z_i; \theta) + \dots + \underbrace{\pi(z_N|x, \theta) \ell(x, z_N; \theta)}_{=0}
 \end{aligned}$$

No need for computing $\ell(x, z; \theta)$ for all $z \in \mathcal{Z}$!

Discrete, unstructured case: sparsemax

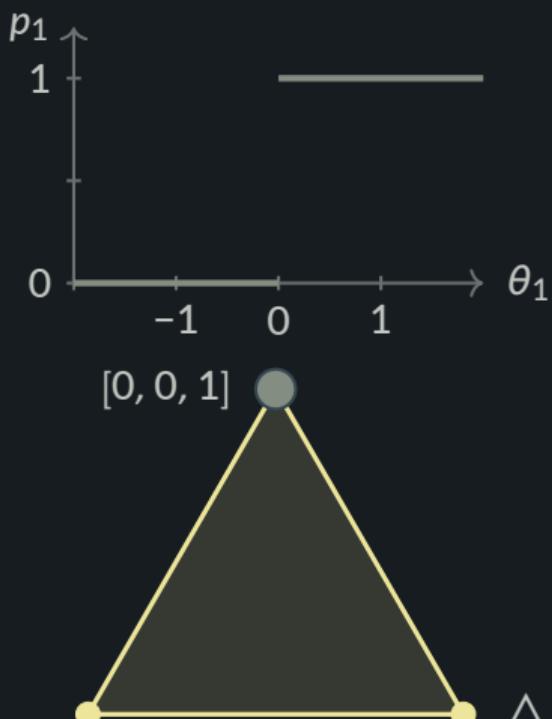
$$\boldsymbol{\pi}_\Omega(\mathbf{s}) = \arg \max_{\mathbf{p} \in \Delta} \mathbf{p}^\top \mathbf{s} - \Omega(\mathbf{p})$$



Discrete, unstructured case: sparsemax

$$\boldsymbol{\pi}_\Omega(\mathbf{s}) = \arg \max_{\mathbf{p} \in \Delta} \mathbf{p}^\top \mathbf{s} - \Omega(\mathbf{p})$$

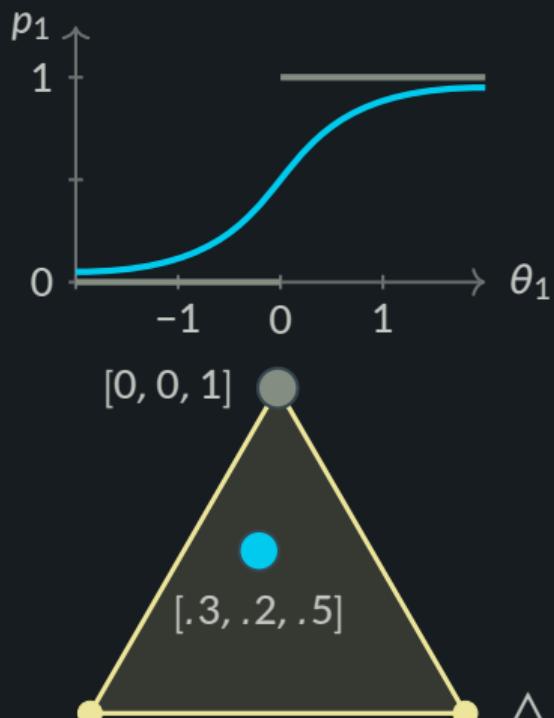
- argmax: $\Omega(\mathbf{p}) = 0$ (*no smoothing*)



Discrete, unstructured case: sparsemax

$$\boldsymbol{\pi}_\Omega(\mathbf{s}) = \arg \max_{\mathbf{p} \in \Delta} \mathbf{p}^\top \mathbf{s} - \Omega(\mathbf{p})$$

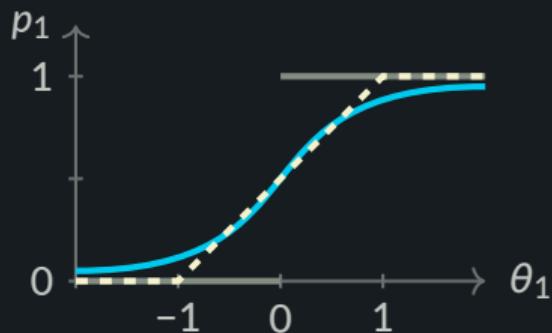
- argmax: $\Omega(\mathbf{p}) = 0$ (*no smoothing*)
- softmax: $\Omega(\mathbf{p}) = \sum_j p_j \log p_j$



Discrete, unstructured case: sparsemax

$$\pi_{\Omega}(s) = \arg \max_{\mathbf{p} \in \Delta} \mathbf{p}^T s - \Omega(\mathbf{p})$$

- argmax: $\Omega(\mathbf{p}) = 0$ (*no smoothing*)
- softmax: $\Omega(\mathbf{p}) = \sum_j p_j \log p_j$
- sparsemax: $\Omega(\mathbf{p}) = 1/2 \|\mathbf{p}\|_2^2$



Semi-Supervised VAE

$$\begin{aligned}\mathcal{L}_x(\theta) &= \sum_{z \in \mathcal{Z}} \pi(z|x) \ell(x, z) \\ &= \mathbb{E}_{z \sim \pi(z|x)} \ell(x, z)\end{aligned}$$

- Semi-Supervised VAE on MNIST: z is one of 10 categories

Semi-Supervised VAE

$$\begin{aligned}\mathcal{L}_x(\theta) &= \sum_{z \in \mathcal{Z}} \pi(z|x) \ell(x, z) \\ &= \mathbb{E}_{z \sim \pi(z|x)} \ell(x, z)\end{aligned}$$

Gaussian VAE

classification network

```
graph TD; A["\mathcal{L}_x(\theta) = \sum_{z \in \mathcal{Z}} \pi(z|x) \ell(x, z)"] --> B["\mathbb{E}_{z \sim \pi(z|x)} \ell(x, z)"]; A --> C["Gaussian VAE"]; B --> D["classification network"]
```

- Semi-Supervised VAE on MNIST: z is one of 10 categories

Semi-Supervised VAE

$$\begin{aligned}\mathcal{L}_x(\theta) &= \sum_{z \in \mathcal{Z}} \pi(z|x) \ell(x, z) \\ &= \mathbb{E}_{z \sim \pi(z|x)} \ell(x, z)\end{aligned}$$

sum over the 10 digits

Gaussian VAE

classification network

The diagram illustrates the loss function for a semi-supervised VAE. It shows two components: a Gaussian VAE loss, represented by the term "sum over the 10 digits" with an arrow pointing to the summation part of the equation, and a classification network loss, represented by the term "classification network" with an arrow pointing to the expectation term. The final equation is the sum of these two components.

- Semi-Supervised VAE on MNIST: z is one of 10 categories

Semi-Supervised VAE

$$\begin{aligned}\mathcal{L}_x(\theta) &= \sum_{z \in \mathcal{Z}} \pi(z|x) \ell(x, z) \\ &= \mathbb{E}_{z \sim \pi(z|x)} \ell(x, z)\end{aligned}$$

sum over the 10 digits

Gaussian VAE

classification network

The diagram illustrates the loss function for a semi-supervised Variational Autoencoder (VAE). The total loss $\mathcal{L}_x(\theta)$ is the sum of two components: the Gaussian VAE loss (sum over the 10 digits) and the classification network loss (sum over latent variables z). The Gaussian VAE loss is represented by the term $\sum_{z \in \mathcal{Z}} \pi(z|x) \ell(x, z)$, and the classification network loss is represented by the term $\mathbb{E}_{z \sim \pi(z|x)} \ell(x, z)$. Arrows from the text labels 'sum over the 10 digits' and 'Gaussian VAE' point to the first term, while an arrow from the label 'classification network' points to the second term.

- Semi-Supervised VAE on MNIST: z is one of 10 categories
- Train this with 10% labeled data

Semi-Supervised VAE

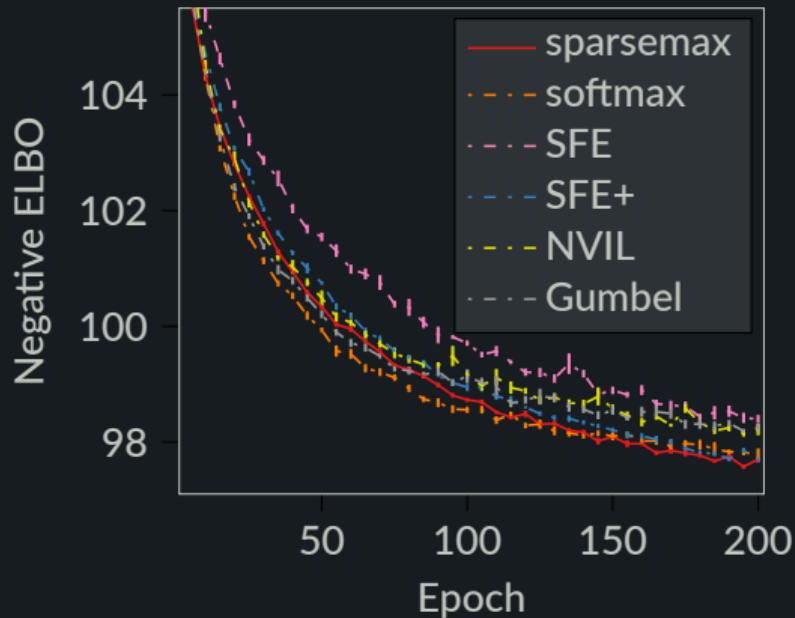
Method	Accuracy (%)	Dec. calls
<i>Monte Carlo</i>		
SFE	$94.75 \pm .002$	1
SFE+	$96.53 \pm .001$	2
NVIL	$96.01 \pm .002$	1
Gumbel	$95.46 \pm .001$	1
<i>Marginalization</i>		
Dense	$96.93 \pm .001$	10

Semi-Supervised VAE

Method	Accuracy (%)	Dec. calls
<i>Monte Carlo</i>		
SFE	$94.75 \pm .002$	1
SFE+	$96.53 \pm .001$	2
NVIL	$96.01 \pm .002$	1
Gumbel	$95.46 \pm .001$	1
<i>Marginalization</i>		
Dense	$96.93 \pm .001$	10
Sparse	$96.87 \pm .001$	1.01 ± 0.01

Semi-Supervised VAE

Method	Accuracy (%)	Dec. calls
<i>Monte Carlo</i>		
SFE	$94.75 \pm .002$	1
SFE+	$96.53 \pm .001$	2
NVIL	$96.01 \pm .002$	1
Gumbel	$95.46 \pm .001$	1
<i>Marginalization</i>		
Dense	$96.93 \pm .001$	10
Sparse	$96.87 \pm .001$	$1.01 \pm .001$



Emergent communication

$$\begin{aligned}\mathcal{L}_x(\theta) &= \sum_{z \in \mathcal{Z}} \pi(z|x) \ell(x, z) \\ &= \mathbb{E}_{z \sim \pi(z|x)} \ell(x, z)\end{aligned}$$

Emergent communication

$$\begin{aligned}\mathcal{L}_x(\theta) &= \sum_{z \in \mathcal{Z}} \pi(z|x) \ell(x, z) \\ &= \mathbb{E}_{z \sim \pi(z|x)} \ell(x, z)\end{aligned}$$

The diagram consists of two curved arrows. One arrow originates from the word "sender" at the top right and points to the term $\pi(z|x)$ in the first equation. Another arrow originates from the word "receiver" at the bottom right and points to the term $\ell(x, z)$ in the second equation.

- receiver picks image from a set \mathcal{V} based on message

Emergent communication

The diagram shows the loss function $\mathcal{L}_x(\theta)$ as a sum over all possible messages $z \in \mathcal{Z}$. The term $\pi(z|x)$ is highlighted in red. The equation is split into two parts: the first part shows the sum over messages, and the second part shows the expectation over messages. Arrows point from the text "sum over all possible messages in the vocabulary" to the summation symbol and from the text "sender" and "receiver" to the expectation symbol and the message term $\pi(z|x)$.

$$\mathcal{L}_x(\theta) = \sum_{z \in \mathcal{Z}} \pi(z|x) \ell(x, z)$$
$$= \mathbb{E}_{z \sim \pi(z|x)} \ell(x, z)$$

sum over
all possible messages
in the vocabulary

sender

receiver

- receiver picks image from a set \mathcal{V} based on message
- images come from ImageNet

Emergent Communication

... but make it harder: $|\mathcal{Z}| = 256$, $|\mathcal{V}| = 16$

Method	success (%)	Dec. calls
<i>Monte Carlo</i>		
SFE	33.05 ± 2.84	1
SFE+	44.32 ± 2.72	2
NVIL	37.04 ± 1.61	1
Gumbel	23.51 ± 16.19	1
ST Gumbel	27.42 ± 13.36	1
<i>Marginalization</i>		

Emergent Communication

... but make it harder: $|\mathcal{Z}| = 256$, $|\mathcal{V}| = 16$

Method	success (%)	Dec. calls
<i>Monte Carlo</i>		
SFE	33.05 ± 2.84	1
SFE+	44.32 ± 2.72	2
NVIL	37.04 ± 1.61	1
Gumbel	23.51 ± 16.19	1
ST Gumbel	27.42 ± 13.36	1
<i>Marginalization</i>		
Dense	93.37 ± 0.42	256

Emergent Communication

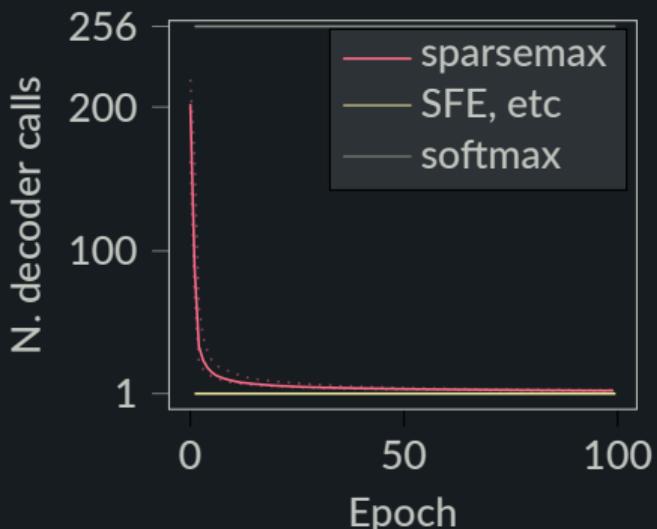
... but make it harder: $|\mathcal{Z}| = 256$, $|\mathcal{V}| = 16$

Method	success (%)	Dec. calls
<i>Monte Carlo</i>		
SFE	33.05 ± 2.84	1
SFE+	44.32 ± 2.72	2
NVIL	37.04 ± 1.61	1
Gumbel	23.51 ± 16.19	1
ST Gumbel	27.42 ± 13.36	1
<i>Marginalization</i>		
Dense	93.37 ± 0.42	256
Sparse	93.35 ± 0.50	3.13 ± 0.48

Emergent Communication

... but make it harder: $|\mathcal{Z}| = 256$, $|\mathcal{V}| = 16$

Method	success (%)	Dec. calls
<i>Monte Carlo</i>		
SFE	33.05 ± 2.84	1
SFE+	44.32 ± 2.72	2
NVIL	37.04 ± 1.61	1
Gumbel	23.51 ± 16.19	1
ST Gumbel	27.42 ± 13.36	1
<i>Marginalization</i>		
Dense	93.37 ± 0.42	256
Sparse	93.35 ± 0.50	3.13 ± 0.48



Limitations

- Mostly (and eventually) very sparse.
But fully dense worst case.
- For the same reason, sparsemax cannot handle structured z .

Limitations

- Mostly (and eventually) very sparse.
But fully dense worst case.
- For the same reason, sparsemax cannot handle structured z.

One solution: top-k sparsemax

$$k\text{-sparsemax}(s) = \arg \min_{p \in \Delta, \|p\|_0 \leq k} \|p - s\|_2^2$$

Limitations

- Mostly (and eventually) very sparse.
But fully dense worst case.
- For the same reason, sparsemax cannot handle structured z .

One solution: **top-k sparsemax**

$$k\text{-sparsemax}(s) = \arg \min_{\mathbf{p} \in \Delta, \|\mathbf{p}\|_0 \leq k} \|\mathbf{p} - s\|_2^2$$

- Non-convex but easy: sparsemax over the k highest scores (Kyrillidis et al., 2013).
- Top-k oracle available for some structured problems.
- Certificate: if at least one of the top-k z gets $p(z) = 0$, **k -sparsemax = sparsemax!**
thus, biased early on, but it goes away.

 Δ  \mathcal{M}

$$\begin{aligned}\mathcal{M} &:= \text{conv} \left\{ \mathbf{a}_z : z \in \mathcal{Z} \right\} \\ &= \left\{ \mathbf{A}p : p \in \Delta \right\} \\ &= \left\{ \mathbb{E}_{Z \sim p} \mathbf{a}_Z : p \in \Delta \right\}\end{aligned}$$



Δ



\mathcal{M}

- $\text{argmax}_{p \in \Delta} p^T s$



Δ



\mathcal{M}

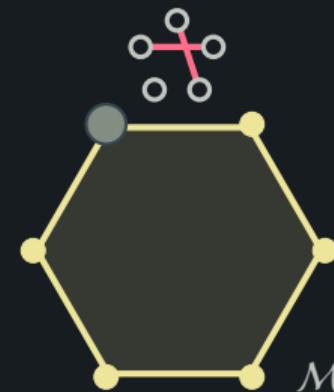
-

$$\text{argmax}_{p \in \Delta} p^T s$$

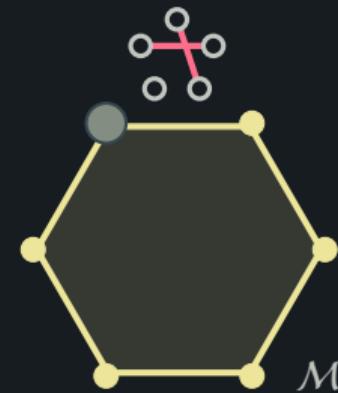


$$\text{MAP}_{\mu \in \mathcal{M}} \arg \max \mu^T t$$

-



- **argmax** $\arg \max_{\boldsymbol{p} \in \Delta} \boldsymbol{p}^T \boldsymbol{s}$
- **MAP** $\arg \max_{\boldsymbol{\mu} \in \mathcal{M}} \boldsymbol{\mu}^T \boldsymbol{t}$
- **softmax** $\arg \max_{\boldsymbol{p} \in \Delta} \boldsymbol{p}^T \boldsymbol{s} + H(\boldsymbol{p})$



- **argmax** $\arg \max_{\boldsymbol{p} \in \Delta} \boldsymbol{p}^T \boldsymbol{s}$
- **softmax** $\arg \max_{\boldsymbol{p} \in \Delta} \boldsymbol{p}^T \boldsymbol{s} + H(\boldsymbol{p})$
- **MAP** $\arg \max_{\boldsymbol{\mu} \in \mathcal{M}} \boldsymbol{\mu}^T \boldsymbol{t}$
- **marginals** $\arg \max_{\boldsymbol{\mu} \in \mathcal{M}} \boldsymbol{\mu}^T \boldsymbol{t} + \tilde{H}(\boldsymbol{\mu})$



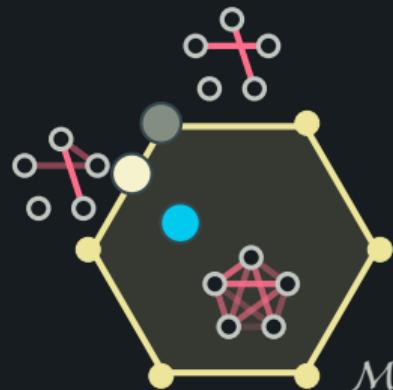
- **argmax** $\arg \max_{\boldsymbol{p} \in \Delta} \boldsymbol{p}^T \boldsymbol{s}$
- **softmax** $\arg \max_{\boldsymbol{p} \in \Delta} \boldsymbol{p}^T \boldsymbol{s} + H(\boldsymbol{p})$
- **sparsemax** $\arg \max_{\boldsymbol{p} \in \Delta} \boldsymbol{p}^T \boldsymbol{s} - 1/2 \|\boldsymbol{p}\|^2$
- **MAP** $\arg \max_{\boldsymbol{\mu} \in \mathcal{M}} \boldsymbol{\mu}^T \boldsymbol{t}$
- **marginals** $\arg \max_{\boldsymbol{\mu} \in \mathcal{M}} \boldsymbol{\mu}^T \boldsymbol{t} + \tilde{H}(\boldsymbol{\mu})$



- **argmax** $\arg \max_{\boldsymbol{p} \in \Delta} \boldsymbol{p}^T \boldsymbol{s}$
- **softmax** $\arg \max_{\boldsymbol{p} \in \Delta} \boldsymbol{p}^T \boldsymbol{s} + H(\boldsymbol{p})$
- **sparsemax** $\arg \max_{\boldsymbol{p} \in \Delta} \boldsymbol{p}^T \boldsymbol{s} - 1/2 \|\boldsymbol{p}\|^2$



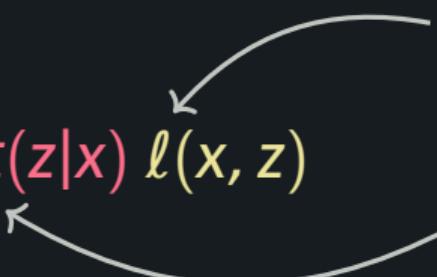
- **MAP** $\arg \max_{\boldsymbol{\mu} \in \mathcal{M}} \boldsymbol{\mu}^T \boldsymbol{t}$
- **marginals** $\arg \max_{\boldsymbol{\mu} \in \mathcal{M}} \boldsymbol{\mu}^T \boldsymbol{t} + \tilde{H}(\boldsymbol{\mu})$
- **SparseMAP** $\arg \max_{\boldsymbol{\mu} \in \mathcal{M}} \boldsymbol{\mu}^T \boldsymbol{t} - 1/2 \|\boldsymbol{\mu}\|^2$



Bit-vector VAE

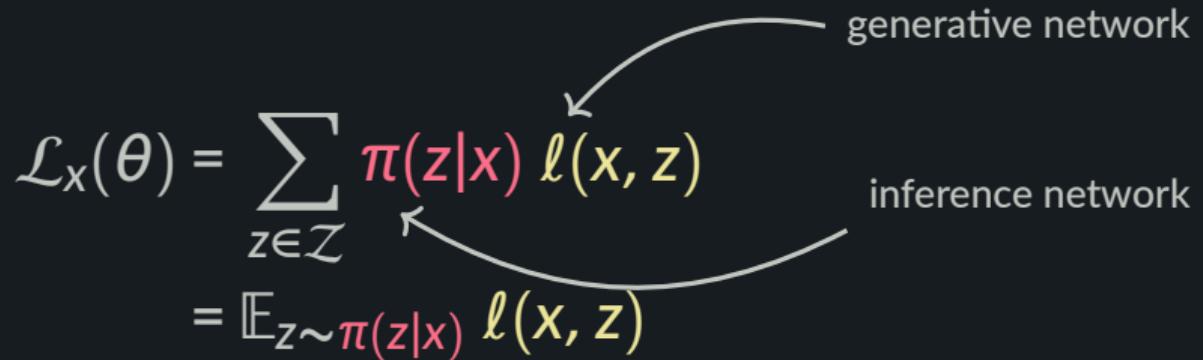
$$\begin{aligned}\mathcal{L}_x(\theta) &= \sum_{z \in \mathcal{Z}} \pi(z|x) \ell(x, z) \\ &= \mathbb{E}_{z \sim \pi(z|x)} \ell(x, z)\end{aligned}$$

Bit-vector VAE

$$\begin{aligned}\mathcal{L}_x(\theta) &= \sum_{z \in \mathcal{Z}} \pi(z|x) \ell(x, z) \\ &= \mathbb{E}_{z \sim \pi(z|x)} \ell(x, z)\end{aligned}$$


- VAE where z is a collection of D bits

Bit-vector VAE

$$\begin{aligned}\mathcal{L}_x(\theta) &= \sum_{z \in \mathcal{Z}} \pi(z|x) \ell(x, z) \\ &= \mathbb{E}_{z \sim \pi(z|x)} \ell(x, z)\end{aligned}$$


The diagram illustrates the components of the Bit-vector VAE loss function. Two curved arrows originate from the right side of the slide. One arrow points from the text "generative network" to the term $\ell(x, z)$. The other arrow points from the text "inference network" to the term $\pi(z|x)$.

- VAE where z is a collection of D bits
- Minimize the negative ELBO

Bit-vector VAE

sum over
an exponentially large
set of structures

$$\begin{aligned}\mathcal{L}_x(\theta) &= \sum_{z \in \mathcal{Z}} \pi(z|x) \ell(x, z) \\ &= \mathbb{E}_{z \sim \pi(z|x)} \ell(x, z)\end{aligned}$$

generative network

inference network

The diagram illustrates the Bit-vector VAE loss function. It starts with the expression $\mathcal{L}_x(\theta)$ on the left, which is annotated with a curved arrow pointing to it from the text "sum over an exponentially large set of structures". This expression is then shown to be equivalent to a sum over all possible latent variable configurations $z \in \mathcal{Z}$, where each term is the product of the probability density $\pi(z|x)$ and the reconstruction loss $\ell(x, z)$. This sum is further simplified to an expectation $\mathbb{E}_{z \sim \pi(z|x)} \ell(x, z)$. The right side of the equation is annotated with two curved arrows: one pointing to the term $\pi(z|x)$ labeled "inference network", and another pointing to the term $\ell(x, z)$ labeled "generative network".

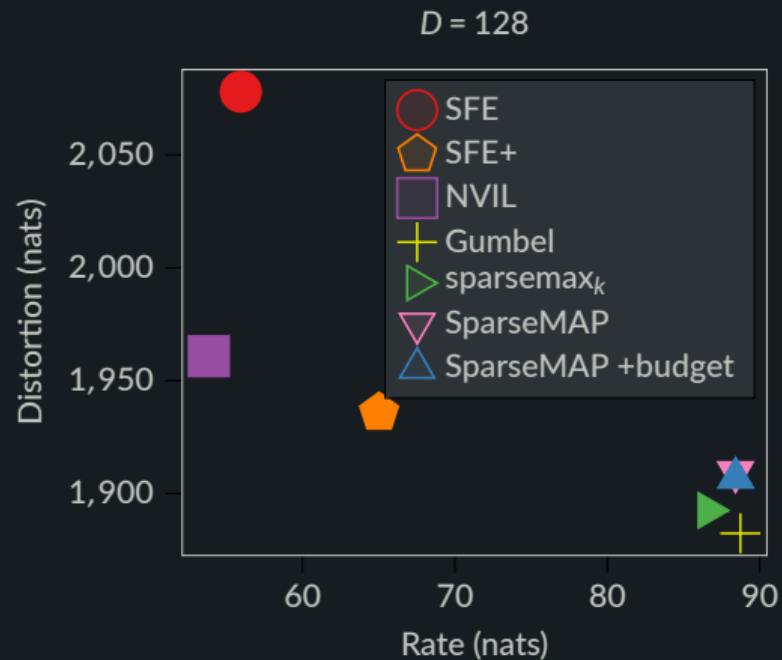
- VAE where z is a collection of D bits
- Minimize the negative ELBO

Bit-vector VAE

Method	$D = 32$	$D = 128$
<i>Monte Carlo</i>		
SFE	3.74	3.77
SFE+	3.61	3.59
NVIL	3.65	3.60
Gumbel	3.57	3.49
<i>Marginalization</i>		
Top- k sparsemax	3.62	3.61
SparseMAP	3.72	3.67
SparseMAP (w/ budget)	3.64	3.66

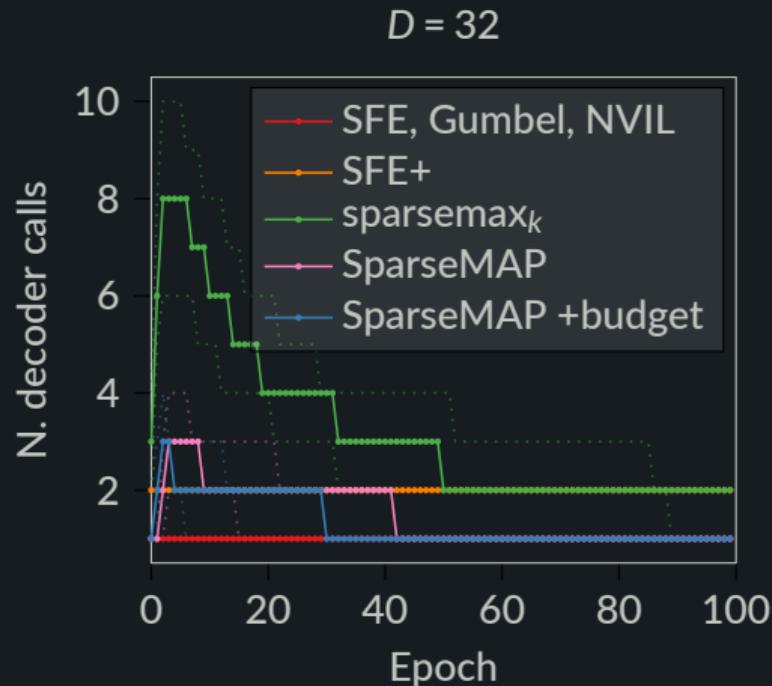
Bit-vector VAE

Method	$D = 32$	$D = 128$
<i>Monte Carlo</i>		
SFE	3.74	3.77
SFE+	3.61	3.59
NVIL	3.65	3.60
Gumbel	3.57	3.49
<i>Marginalization</i>		
Top-k sparsemax	3.62	3.61
SparseMAP	3.72	3.67
SparseMAP (w/ budget)	3.64	3.66



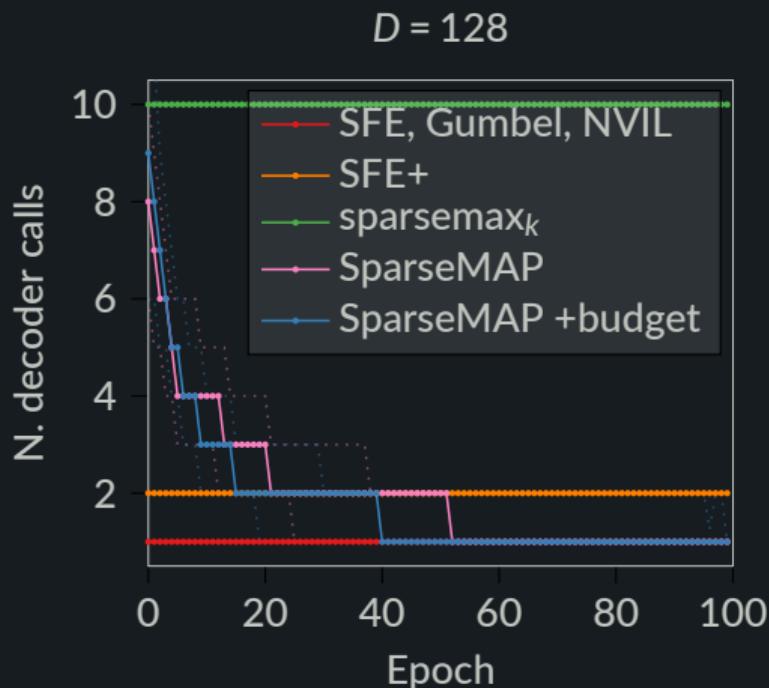
Bit-vector VAE

Method	$D = 32$	$D = 128$
<i>Monte Carlo</i>		
SFE	3.74	3.77
SFE+	3.61	3.59
NVIL	3.65	3.60
Gumbel	3.57	3.49
<i>Marginalization</i>		
Top- k sparsemax	3.62	3.61
SparseMAP	3.72	3.67
SparseMAP (w/ budget)	3.64	3.66



Bit-vector VAE

Method	$D = 32$	$D = 128$
<i>Monte Carlo</i>		
SFE	3.74	3.77
SFE+	3.61	3.59
NVIL	3.65	3.60
Gumbel	3.57	3.49
<i>Marginalization</i>		
Top- k sparsemax	3.62	3.61
SparseMAP	3.72	3.67
SparseMAP (w/ budget)	3.64	3.66



Key Takeaways

We introduce a new method
to train latent variable models.

Key Takeaways

We introduce a new method
to train latent variable models.

discrete and structured

0.2 0.6 0.1


[] 0.4
[] 0.05
...
[] 0.3

Key Takeaways

We introduce a new method
to train latent variable models.

discrete and structured

0.2	0.6	0.1		
[]	0.4
[]	0.05
...				
[]	0.3

deterministic, yet efficient

$$\begin{aligned}\mathcal{L}_x(\theta) = & \pi(z_1|x, \theta) \ell(x, z_1; \theta) \\ & + \underbrace{\pi(z_2|x, \theta) \ell(x, z_2; \theta)}_{=0} \\ & + \dots + \pi(z_i|x, \theta) \ell(x, z_i; \theta) \\ & + \dots + \underbrace{\pi(z_N|x, \theta) \ell(x, z_N; \theta)}_{=0}\end{aligned}$$

Key Takeaways

We introduce a new method
to train latent variable models.

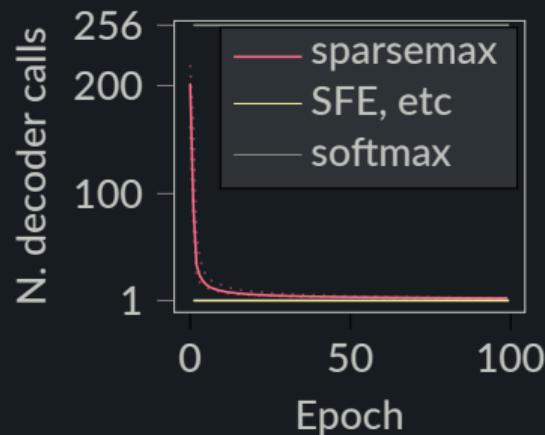
discrete and structured



deterministic, yet efficient

$$\begin{aligned}\mathcal{L}_x(\theta) &= \pi(z_1|x, \theta) \ell(x, z_1; \theta) \\ &\quad + \underbrace{\pi(z_2|x, \theta) \ell(x, z_2; \theta)}_{=0} \\ &\quad + \dots + \pi(z_i|x, \theta) \ell(x, z_i; \theta) \\ &\quad + \dots + \underbrace{\pi(z_N|x, \theta) \ell(x, z_N; \theta)}_{=0}\end{aligned}$$

adaptive, as needed



References I

-  Bouges, Pierre, Thierry Chateau, Christophe Blanc, and Gaëlle Loosli (Dec. 2013). "Handling missing weak classifiers in boosted cascade: application to multiview and occluded face detection". In: *EURASIP Journal on Image and Video Processing* 2013, p. 55. DOI: [10.1186/1687-5281-2013-55](https://doi.org/10.1186/1687-5281-2013-55).
-  Correia, Gonçalo M., Vlad Niculae, Wilker Aziz, and André FT Martins (2020). "Efficient marginalization of discrete and structured latent variables via sparsity". In: *Proc. NeurIPS*.
-  Kyriolidis, Anastasios, Stephen Becker, Volkan Cevher, and Christoph Koch (2013). "Sparse projections onto the simplex". In: *Proc. ICML*.
-  Lazaridou, Angeliki, Alexander Peysakhovich, and Marco Baroni (2017). "Multi-agent cooperation and the emergence of (natural) language". In: *Proc. ICLR*.
-  Liu, Ziwei, Ping Luo, Xiaogang Wang, and Xiaoou Tang (Dec. 2015). "Deep Learning Face Attributes in the Wild". In: *Proceedings of International Conference on Computer Vision (ICCV)*.
-  Martins, André FT and Ramón Fernandez Astudillo (2016). "From softmax to sparsemax: A sparse model of attention and multi-label classification". In: *Proc. of ICML*.
-  Niculae, Vlad and Mathieu Blondel (2017). "A regularized framework for sparse and structured neural attention". In: *Proc. of NeurIPS*.
-  Niculae, Vlad, André FT Martins, Mathieu Blondel, and Claire Cardie (2018). "SparseMAP: Differentiable sparse structured inference". In: *Proc. of ICML*.