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

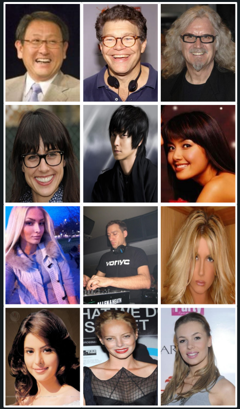
Latent Variable Models

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

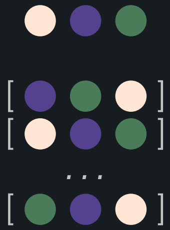
Latent variable z can be

Training Discrete or Structured Latent Variable Models

Latent variable z can be discrete

Training Discrete or Structured Latent Variable Models

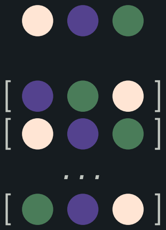
Latent variable z can be discrete or structured



Training Discrete or Structured Latent Variable Models

Latent variable z can be discrete or structured

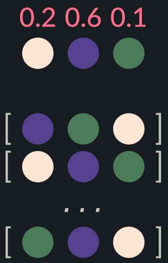
$\pi(z|x, \theta)$: distribution over possible z



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) $\rightarrow$ unbiased but high variance

Current Solutions

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

One option: SFE (aka REINFORCE) $\rightarrow$ unbiased but high variance
Another option: Gumbel-Softmax $\rightarrow$ continuous relaxation, biased estimation

Current Solutions

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

One option: SFE (aka REINFORCE) $\longrightarrow$ unbiased but high variance

Another option: Gumbel-Softmax $\longrightarrow$ 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) $\longrightarrow$ unbiased but high variance

Another option: Gumbel-Softmax $\longrightarrow$ continuous relaxation, biased estimation

New option: use sparsity! 🙌

no need for sampling $\longrightarrow$ no variance

Current Solutions

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

One option: SFE (aka REINFORCE) $\longrightarrow$ unbiased but high variance

Another option: Gumbel-Softmax $\longrightarrow$ continuous relaxation, biased estimation

New option: use sparsity! 🙌

no need for sampling $\longrightarrow$ 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?

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

Taking a step back...

Does the expectation over possible z need to be expensive?

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

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

(Martins and Astudillo, 2016; Niculae, Martins, et al., 2018)

Sparse normalizers

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

(Martins and Astudillo, 2016; Niculae, Martins, et al., 2018)

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**!

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

(Martins and Astudillo, 2016; Niculae, Martins, et al., 2018)

Sparse normalizers

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

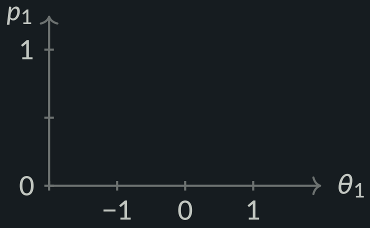
These functions are able to assign **probabilities of exactly zero**!

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

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

Discrete, unstructured case: sparsemax

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

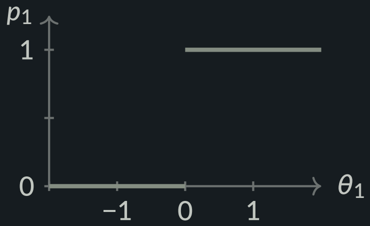


(Niculae and Blondel, 2017)

Discrete, unstructured case: sparsemax

$$\boldsymbol{\pi}_\Omega(s) = \underset{\boldsymbol{p}\in\triangle}{\arg\max}\, \boldsymbol{p}^\top s - \Omega(\boldsymbol{p})$$

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

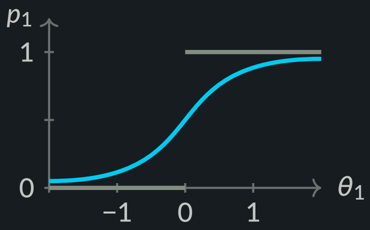


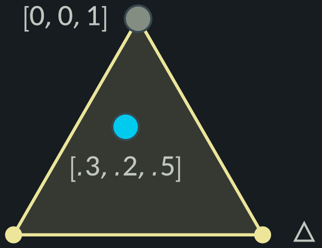
(Niculae and Blondel, 2017)

Discrete, unstructured case: sparsemax

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

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



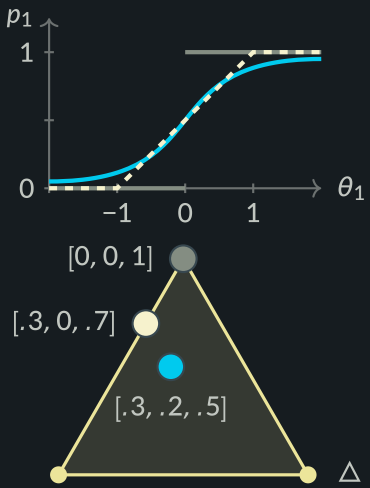


(Niculae and Blondel, 2017)

Discrete, unstructured case: sparsemax

$$\boldsymbol{\pi}_\Omega(s) = \underset{\boldsymbol{p}\in\triangle}{\arg\max}\, \boldsymbol{p}^\top s - \Omega(\boldsymbol{p})$$

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



(Martins and Astudillo, 2016)

Semi-Supervised VAE

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

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

Semi-Supervised VAE

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

$$= \mathbb{E}_{z \sim \pi(z|x)}\, \ell(x, z)$$

Gaussian VAE

classification network

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

Semi-Supervised VAE

sum over
the 10 digits

Gaussian VAE

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

classification network

$$= \mathbb{E}_{z \sim \pi(z|x)}\, \ell(x, z)$$

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

Semi-Supervised VAE

sum over
the 10 digits

Gaussian VAE

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

classification network

$$= \mathbb{E}_{z \sim \pi(z|x)}\, \ell(x, z)$$

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

Semi-Supervised VAE

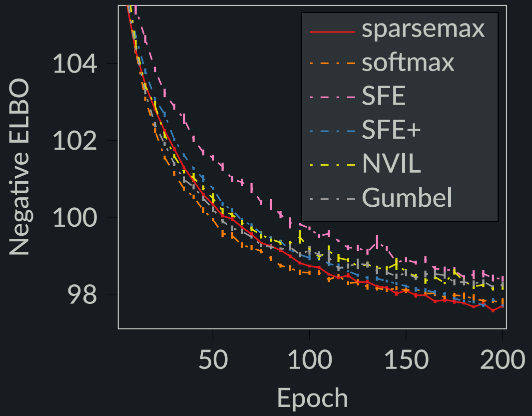
Method	Accuracy (%)	Dec. calls
Monte Carlo		
SFE	94.75$_{\pm .002}$	1
SFE+	96.53$_{\pm .001}$	2
NVIL	96.01$_{\pm .002}$	1
Gumbel	95.46$_{\pm .001}$	1
Marginalization		
Dense	96.93$_{\pm .001}$	10

Semi-Supervised VAE

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

Semi-Supervised VAE

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



Emergent communication

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

Emergent communication

sender

receiver

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

$$= \mathbb{E}_{z \sim \pi(z|x)}\ \ell(x, z)$$

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

Emergent communication

sum over
all possible messages
in the vocabulary

sender

receiver

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

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

(Lazaridou et al., 2017; Correia et al., 2020)

Emergent Communication

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

Method	success (%)	Dec. calls
Monte Carlo		
SFE	33.05 $\pm$2.84	1
SFE+	44.32 $\pm$2.72	2
NVIL	37.04 $\pm$1.61	1
Gumbel	23.51 $\pm$16.19	1
ST Gumbel	27.42 $\pm$13.36	1
Marginalization		

(Lazaridou et al., 2017; Correia et al., 2020)

Emergent Communication

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

Method	success (%)	Dec. calls
Monte Carlo		
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
Marginalization		
Dense	93.37 ±0.42	256

(Lazaridou et al., 2017; Correia et al., 2020)

Emergent Communication

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

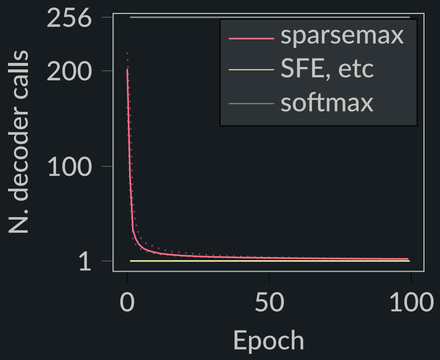
Method	success (%)	Dec. calls
Monte Carlo		
SFE	33.05 ±2.84	1
SFE+	44.32 ±2.72	2
NVIL	37.04 ±1.61	1
Gumbel	23.51 ±16.19	1
ST Gumbel	27.42 ±13.36	1
Marginalization		
Dense	93.37 ±0.42	256
Sparse	93.35 ±0.50	3.13±0.48

(Lazaridou et al., 2017; Correia et al., 2020)

Emergent Communication

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

Method	success (%)	Dec. calls
Monte Carlo		
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
Marginalization		
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

$$k\text{-sparsemax}(s) = \underset{\boldsymbol{p} \in \triangle, \|\boldsymbol{p}\|_0 \leq k}{\arg\min} \|\boldsymbol{p} - s\|_2^2$$

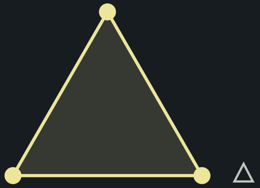
Limitations

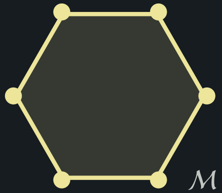
- Mostly (and eventually) very sparse.
  But fully dense worst case.
- For the same reason, sparsemax cannot handle structured z.

One solution: top-k sparsemax

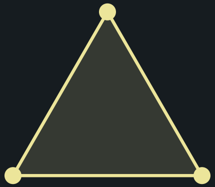
$$k\text{-sparsemax}(s) = \operatorname*{arg\,min}_{\boldsymbol{p}\in\triangle, \|\boldsymbol{p}\|_0 \le k} \|\boldsymbol{p} - s\|_2^2$$

- 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.

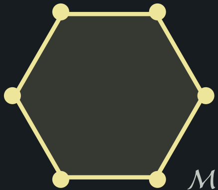

△


$\mathcal{M}$

$$
\begin{aligned}
\mathcal{M} := \operatorname{conv}\left\{\boldsymbol{a}_z : z \in \mathcal{Z}\right\} \\
= \left\{\boldsymbol{A}\boldsymbol{p} : \boldsymbol{p} \in \triangle\right\} \\
= \left\{\mathbb{E}_{Z\sim \boldsymbol{p}}\, \boldsymbol{a}_Z : \boldsymbol{p} \in \triangle\right\}
\end{aligned}
$$

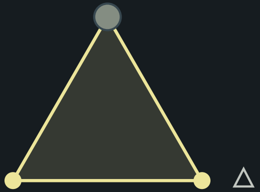


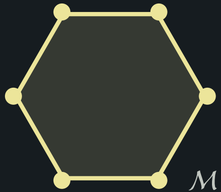
$\triangle$



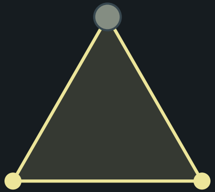
$\mathcal{M}$

- **argmax** $\displaystyle \operatorname*{arg\,max}_{\boldsymbol{p} \in \triangle} \boldsymbol{p}^\top \boldsymbol{s}$

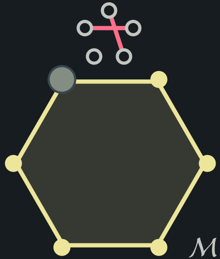




- **argmax** $\arg\max_{\boldsymbol{p} \in \triangle} \boldsymbol{p}^\top \boldsymbol{s}$
- **MAP** $\arg\max_{\boldsymbol{\mu} \in \mathcal{M}} \boldsymbol{\mu}^\top \boldsymbol{t}$



$\triangle$

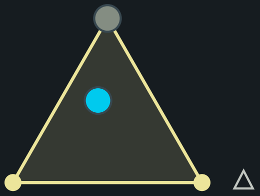


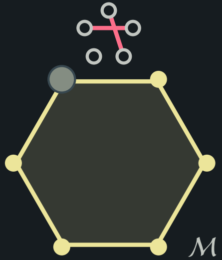
$\mathcal{M}$

argmax $\arg\max_{\boldsymbol{p} \in \triangle} \boldsymbol{p}^\top \boldsymbol{s}$

MAP $\arg\max_{\boldsymbol{\mu} \in \mathcal{M}} \boldsymbol{\mu}^\top \boldsymbol{t}$

softmax $\arg\max_{\boldsymbol{p} \in \triangle} \boldsymbol{p}^\top \boldsymbol{s} + \mathrm{H}(\boldsymbol{p})$



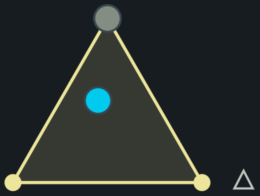


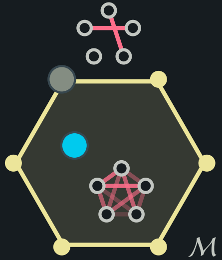
argmax $\arg\max_{\boldsymbol{p}\in\triangle} \boldsymbol{p}^\top \boldsymbol{s}$

softmax $\arg\max_{\boldsymbol{p}\in\triangle} \boldsymbol{p}^\top \boldsymbol{s} + \mathrm{H}(\boldsymbol{p})$

MAP $\arg\max_{\boldsymbol{\mu}\in\mathcal{M}} \boldsymbol{\mu}^\top \boldsymbol{t}$

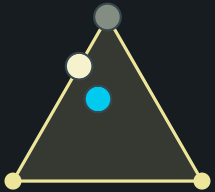
marginals $\arg\max_{\boldsymbol{\mu}\in\mathcal{M}} \boldsymbol{\mu}^\top \boldsymbol{t} + \widetilde{\mathrm{H}}(\boldsymbol{\mu})$



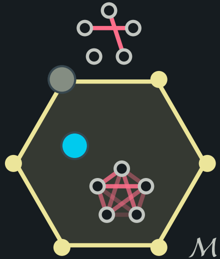


- **argmax** $\arg\max_{\boldsymbol{p}\in\triangle} \boldsymbol{p}^\top \boldsymbol{s}$
- **softmax** $\arg\max_{\boldsymbol{p}\in\triangle} \boldsymbol{p}^\top \boldsymbol{s} + \mathrm{H}(\boldsymbol{p})$
- **sparsemax** $\arg\max_{\boldsymbol{p}\in\triangle} \boldsymbol{p}^\top \boldsymbol{s} - 1/2\|\boldsymbol{p}\|^2$

- **MAP** $\arg\max_{\boldsymbol{\mu}\in\mathcal{M}} \boldsymbol{\mu}^\top \boldsymbol{t}$
- **marginals** $\arg\max_{\boldsymbol{\mu}\in\mathcal{M}} \boldsymbol{\mu}^\top \boldsymbol{t} + \widetilde{\mathrm{H}}(\boldsymbol{\mu})$



$\triangle$



$\mathcal{M}$

(Niculae, Martins, Blondel, and Cardie, 2018)

argmax $\arg\max_{\boldsymbol{p}\in\triangle} \boldsymbol{p}^\top \boldsymbol{s}$

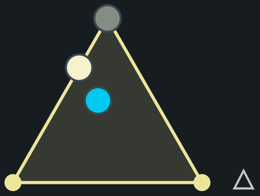
softmax $\arg\max_{\boldsymbol{p}\in\triangle} \boldsymbol{p}^\top \boldsymbol{s} + \mathrm{H}(\boldsymbol{p})$

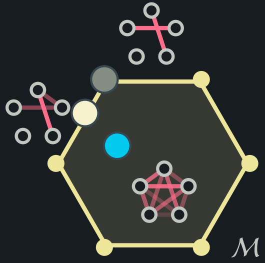
sparsemax $\arg\max_{\boldsymbol{p}\in\triangle} \boldsymbol{p}^\top \boldsymbol{s} - 1/2\|\boldsymbol{p}\|^2$

MAP $\arg\max_{\boldsymbol{\mu}\in\mathcal{M}} \boldsymbol{\mu}^\top \boldsymbol{t}$

marginals $\arg\max_{\boldsymbol{\mu}\in\mathcal{M}} \boldsymbol{\mu}^\top \boldsymbol{t} + \widetilde{\mathrm{H}}(\boldsymbol{\mu})$

SparseMAP $\arg\max_{\boldsymbol{\mu}\in\mathcal{M}} \boldsymbol{\mu}^\top \boldsymbol{t} - 1/2\|\boldsymbol{\mu}\|^2$





Bit-vector VAE

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

Bit-vector VAE

generative network

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

inference network

$$= \mathbb{E}_{z \sim \pi(z|x)}\, \ell(x, z)$$

- VAE where z is a collection of D bits

Bit-vector VAE

generative network

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

inference network

$$= \mathbb{E}_{z \sim \pi(z|x)}\, \ell(x, z)$$

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

Bit-vector VAE

sum over
an exponetially large
set of structures

generative network

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

inference network

$$= \mathbb{E}_{z \sim \pi(z|x)}\, \ell(x, z)$$

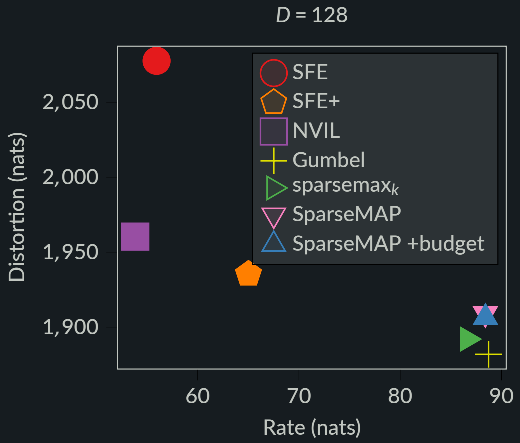
- VAE where z is a collection of D bits
- Minimize the negative ELBO

Bit-vector VAE

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

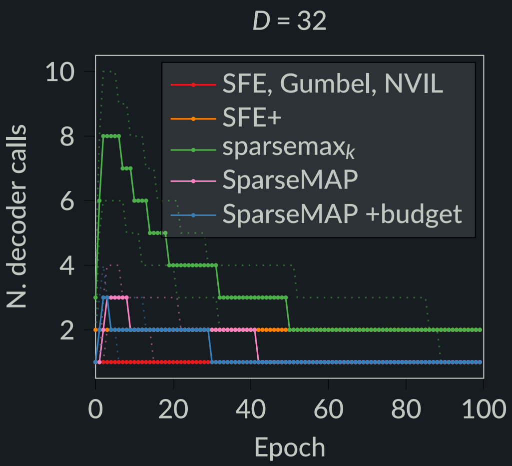
Bit-vector VAE

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



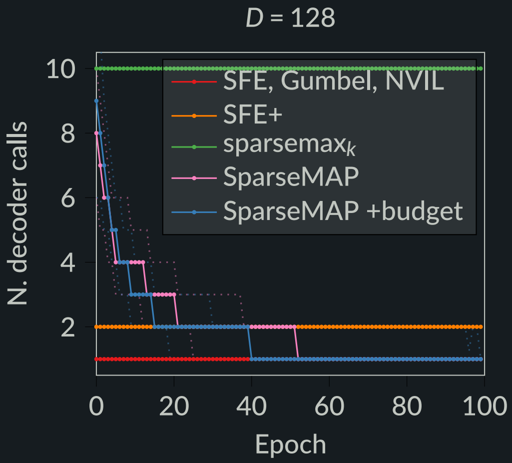
Bit-vector VAE

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



Bit-vector VAE

Method	$D = 32$	$D = 128$
Monte Carlo		
SFE	3.74	3.77
SFE+	3.61	3.59
NVIL	3.65	3.60
Gumbel	3.57	3.49
Marginalization		
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

Key Takeaways

We introduce a new method
to train latent variable models.

discrete and structured

deterministic, yet efficient

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

Key Takeaways

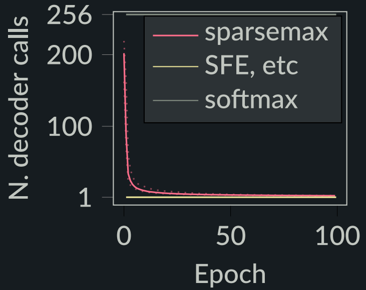
We introduce a new method
to train latent variable models.

discrete and structured

deterministic, yet efficient

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

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.

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*.

Kyrillidis, 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*.