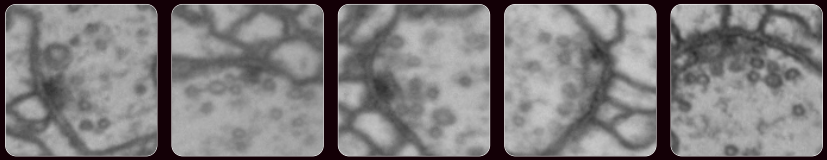
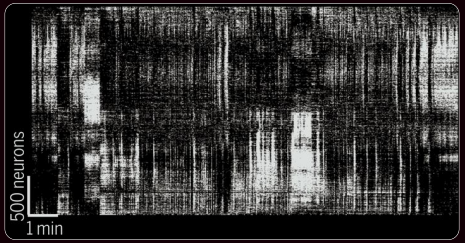
Learning of Disentangled Representations

Jan Funke

Data Analysis Journal Club

May 13, 2021





Neural recordings from: Stringer & Pachitariu et al., "Spontaneous behaviors drive multidimensional, brainwide activity", Science 2019

Formalization

latent variable: z $\in \mathbb{R}^n$

data: x $\in \mathbb{R}^d$

Formalization

latent variable: z $\in \mathbb{R}^n$

data: x $\in \mathbb{R}^d$

$$p(\boldsymbol{x}, \boldsymbol{z}) = p(\boldsymbol{x}|\boldsymbol{z})\, p(\boldsymbol{z})$$

Formalization

latent variable: z $\in \mathbb{R}^n$

data: x $\in \mathbb{R}^d$

$$p(\boldsymbol{x}, \boldsymbol{z}) = p(\boldsymbol{x}|\boldsymbol{z})\, p(\boldsymbol{z})$$

$$p(\boldsymbol{x}) = \int p(\boldsymbol{x}|\boldsymbol{z})\, p(\boldsymbol{z})\, d\boldsymbol{z}$$

Formalization

latent variable: z $\in \mathbb{R}^n$

data: x $\in \mathbb{R}^d$

$$p(\boldsymbol{x}, \boldsymbol{z}) = p(\boldsymbol{x}|\boldsymbol{z})\, p(\boldsymbol{z})$$

$$p(\boldsymbol{x}) = \int p(\boldsymbol{x}|\boldsymbol{z})\, p(\boldsymbol{z})\, d\boldsymbol{z}$$

$$p(\boldsymbol{z}) = \prod_i p(z_i)$$

Formalization

latent variable: $z \in \mathbb{R}^n$

data: $x \in \mathbb{R}^d$

$$p(\boldsymbol{x}, \boldsymbol{z}) = p(\boldsymbol{x}|\boldsymbol{z})\,p(\boldsymbol{z})$$

$$p(\boldsymbol{x}) = \int p(\boldsymbol{x}|\boldsymbol{z})\,p(\boldsymbol{z})\,d\boldsymbol{z}$$

$$p(\boldsymbol{z}) = \prod_i p(z_i)$$

Problem Statement

Given observations $\mathcal{D} = \{\boldsymbol{x}^{(i)}\}_{i=1}^N$ with $\boldsymbol{x}^{(i)} \sim p(\boldsymbol{x})$, we aim to find a model $p_\theta(\boldsymbol{x}, \boldsymbol{z}) = p_\theta(\boldsymbol{x}|\boldsymbol{z})\,p_\theta(\boldsymbol{z})$.

Independent Component Analysis

Linear model:

$$\boldsymbol{x} = \boldsymbol{A}\boldsymbol{z} + \boldsymbol{\epsilon}$$

$$p(\boldsymbol{x}|\boldsymbol{z}) \sim \mathcal{N}(\boldsymbol{A}\boldsymbol{z}, \boldsymbol{\Sigma})$$

Search for $\boldsymbol{W} = \boldsymbol{A}^{-1}$, such that mutual information between components of $\boldsymbol{z} \approx \boldsymbol{W}\boldsymbol{x}$ is minimized.

Independent Component Analysis

Linear model:

$$\boldsymbol{x} = \boldsymbol{A}\boldsymbol{z} + \boldsymbol{\epsilon}$$

$$p(\boldsymbol{x}|\boldsymbol{z}) \sim \mathcal{N}(\boldsymbol{A}\boldsymbol{z}, \boldsymbol{\Sigma})$$

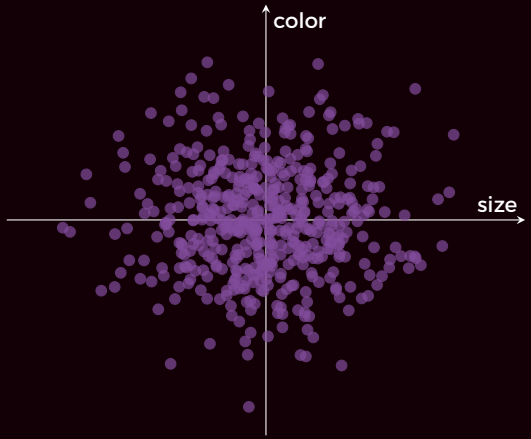
Search for $\boldsymbol{W} = \boldsymbol{A}^{-1}$, such that mutual information between components of $\boldsymbol{z} \approx \boldsymbol{W}\boldsymbol{x}$ is minimized.

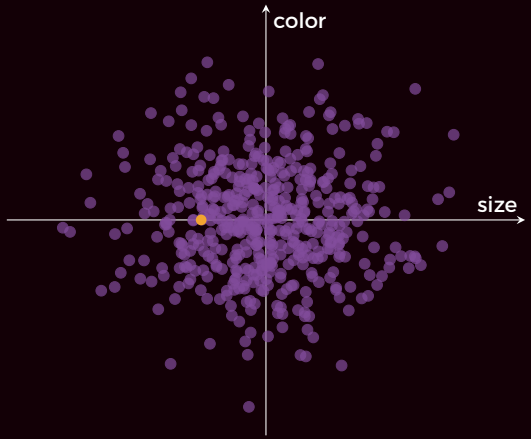
Disentanglement ≈ nonlinear ICA

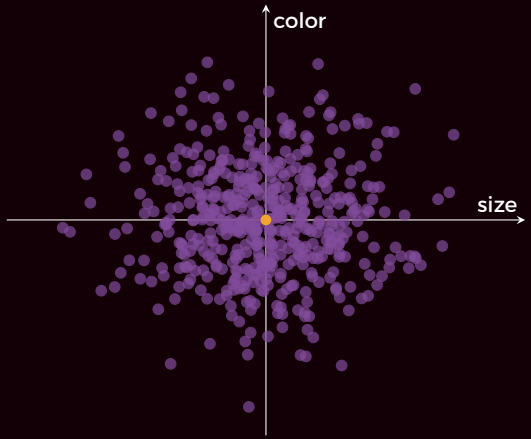
No particular restriction on $p(\boldsymbol{x}|\boldsymbol{z})$:

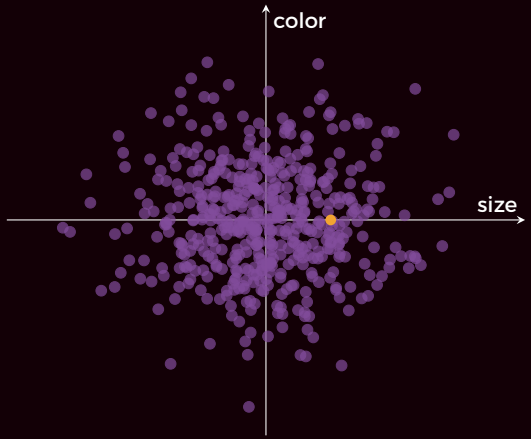
$$\boldsymbol{x} = f(\boldsymbol{z}) + \boldsymbol{\epsilon}$$

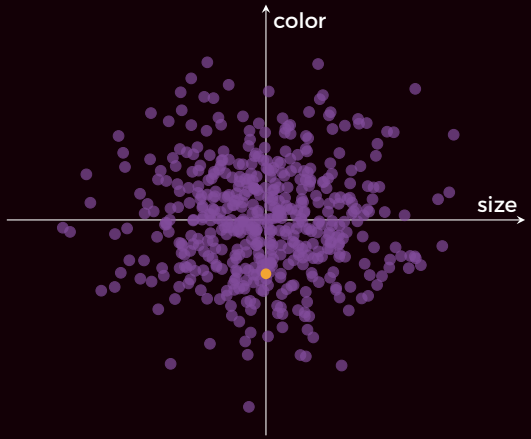
Search for f^{-1}, such that distribution of $\boldsymbol{z} = f^{-1}(\boldsymbol{x})$ factorizes: $p(\boldsymbol{z}) = \prod_i p(z_i)$.

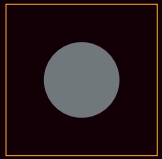

color
size

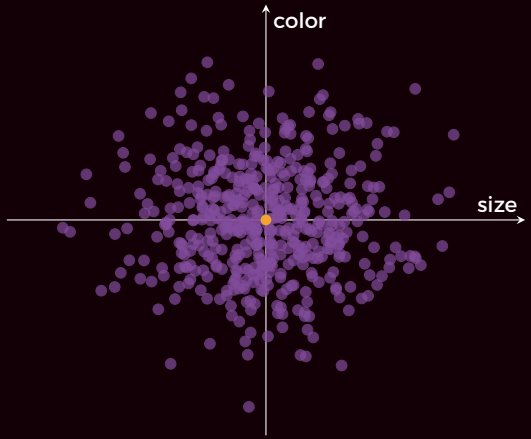

color
size

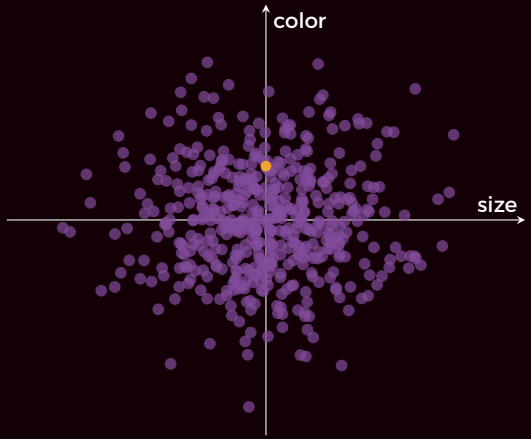

color
size


color
size


color
size




color
size


color
size

Chapter 1: Just do it!

Model Definition

latent variable: z $\in \mathbb{R}^n$

data: x $\in \mathbb{R}^d$

Model Definition

- Model a factorizing prior over latent variables:

$$p_\theta(\boldsymbol{z}) = \prod_i p_\theta(z_i)$$

latent variable: $\boldsymbol{z}$ $\in \mathbb{R}^n$

data: $\boldsymbol{x}$ $\in \mathbb{R}^d$

Model Definition

latent variable: z $\in \mathbb{R}^n$

data: x $\in \mathbb{R}^d$

- Model a factorizing prior over latent variables:

$$p_\theta(z) = \prod_i p_\theta(z_i)$$

- Learn to generate data from latent variables:

$$p_\theta(x|z)$$

such that $p_\theta(x) = \int p_\theta(x|z)p_\theta(z)dz \approx p(x)$

Model Definition

latent variable: z $\in \mathbb{R}^n$

data: x $\in \mathbb{R}^d$

- Model a factorizing prior over latent variables:

$$p_\theta(z) = \prod_i p_\theta(z_i)$$

- Learn to generate data from latent variables:

$$p_\theta(x|z)$$

such that $p_\theta(x) = \int p_\theta(x|z)p_\theta(z)dz \approx p(x)$

- Extra Credit Learn to predict z from x:

$$p_\theta(z|x)$$

Model Definition

latent variable: z $\in \mathbb{R}^n$

data: x $\in \mathbb{R}^d$

- Model a factorizing prior over latent variables:

$$p_\theta(z) = \prod_i p_\theta(z_i)$$

- Learn to generate data from latent variables:

$$p_\theta(x|z)$$

such that $p_\theta(x) = \int p_\theta(x|z)p_\theta(z)dz \approx p(x)$

- Extra Credit Learn to predict z from x:

$$p_\theta(z|x)$$

Variational Autoencoders (VAEs) are well suited for this task.

Variational Autoencoder

Ingredients

- Prior on z:

$$p(\boldsymbol{z}) = \mathcal{N}(\boldsymbol{0}, \boldsymbol{I})$$

- Generator:

$$p_\theta(\boldsymbol{x}|\boldsymbol{z}) = \mathcal{N}(\mu_{\boldsymbol{x}}, \Sigma_{\boldsymbol{x}})$$
$$(\mu_{\boldsymbol{x}}, \Sigma_{\boldsymbol{x}}) = f_\theta(\boldsymbol{z}) \quad \text{(typically a neural network)}$$

- Encoder:

$$q_\theta(\boldsymbol{z}|\boldsymbol{x}) = \mathcal{N}(\mu_{\boldsymbol{z}}, \Sigma_{\boldsymbol{z}})$$
$$(\mu_{\boldsymbol{z}}, \Sigma_{\boldsymbol{z}}) = g_\theta(\boldsymbol{x}) \quad \text{(typically a neural network)}$$
$$q_\theta(\boldsymbol{z}|\boldsymbol{x}) \approx p_\theta(\boldsymbol{z}|\boldsymbol{x})$$

Kingma & Welling, "Auto-Encoding Variational Bayes", https://arxiv.org/abs/1312.6114v10

Variational Autoencoder

Ingredients

- Prior on z:

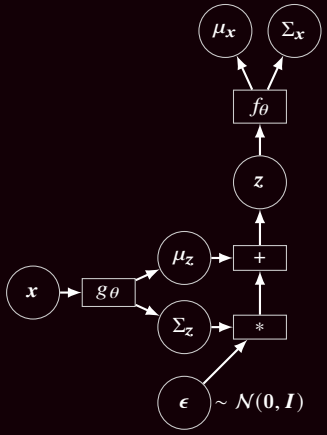
$$p(\boldsymbol{z}) = \mathcal{N}(\boldsymbol{0}, \boldsymbol{I})$$

- Generator:

$$p_\theta(\boldsymbol{x}|\boldsymbol{z}) = \mathcal{N}(\mu_{\boldsymbol{x}}, \Sigma_{\boldsymbol{x}})$$
$$(\mu_{\boldsymbol{x}}, \Sigma_{\boldsymbol{x}}) = f_\theta(\boldsymbol{z}) \quad \text{(typically a neural network)}$$

- Encoder:

$$q_\theta(\boldsymbol{z}|\boldsymbol{x}) = \mathcal{N}(\mu_{\boldsymbol{z}}, \Sigma_{\boldsymbol{z}})$$
$$(\mu_{\boldsymbol{z}}, \Sigma_{\boldsymbol{z}}) = g_\theta(\boldsymbol{x}) \quad \text{(typically a neural network)}$$
$$q_\theta(\boldsymbol{z}|\boldsymbol{x}) \approx p_\theta(\boldsymbol{z}|\boldsymbol{x})$$



Kingma & Welling, "Auto-Encoding Variational Bayes", https://arxiv.org/abs/1312.6114v10

Variational Autoencoder

Ingredients

- Prior on z:

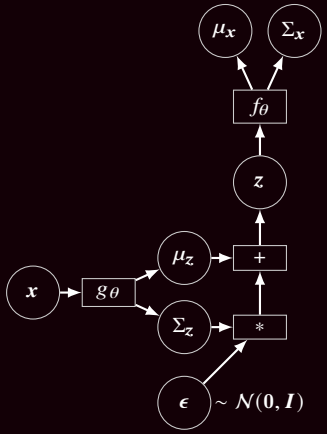
$$p(\boldsymbol{z}) = \mathcal{N}(\boldsymbol{0}, \boldsymbol{I})$$

- Generator:

$$p_\theta(\boldsymbol{x}|\boldsymbol{z}) = \mathcal{N}(\mu_{\boldsymbol{x}}, \Sigma_{\boldsymbol{x}})$$
$$(\mu_{\boldsymbol{x}}, \Sigma_{\boldsymbol{x}}) = f_\theta(\boldsymbol{z}) \quad \text{(typically a neural network)}$$

- Encoder:

$$q_\theta(\boldsymbol{z}|\boldsymbol{x}) = \mathcal{N}(\mu_{\boldsymbol{z}}, \Sigma_{\boldsymbol{z}})$$
$$(\mu_{\boldsymbol{z}}, \Sigma_{\boldsymbol{z}}) = g_\theta(\boldsymbol{x}) \quad \text{(typically a neural network)}$$
$$q_\theta(\boldsymbol{z}|\boldsymbol{x}) \approx p_\theta(\boldsymbol{z}|\boldsymbol{x})$$



Training

$$\mathcal{L}(\theta, \boldsymbol{x}) = \mathbb{E}_{q_\theta(\boldsymbol{z}|\boldsymbol{x})}\left[-\log p_\theta(\boldsymbol{x}|\boldsymbol{z})\right] + D_{\mathsf{KL}}\left(q_\theta(\boldsymbol{z}|\boldsymbol{x}) \,||\, p(\boldsymbol{z})\right) \quad \to \min$$

Kingma & Welling, "Auto-Encoding Variational Bayes", https://arxiv.org/abs/1312.6114v10

Variational Autoencoder

Ingredients

- Prior on z:

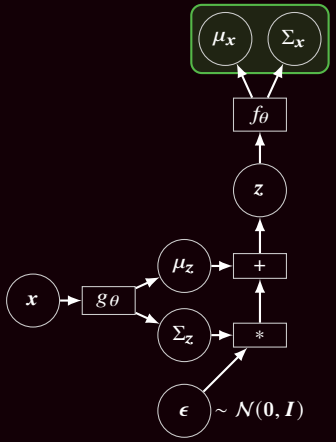
$$p(\boldsymbol{z}) = \mathcal{N}(\mathbf{0}, \boldsymbol{I})$$

- Generator:

$$p_\theta(\boldsymbol{x}|\boldsymbol{z}) = \mathcal{N}(\mu_{\boldsymbol{x}}, \Sigma_{\boldsymbol{x}})$$
$$(\mu_{\boldsymbol{x}}, \Sigma_{\boldsymbol{x}}) = f_\theta(\boldsymbol{z}) \quad \text{(typically a neural network)}$$

- Encoder:

$$q_\theta(\boldsymbol{z}|\boldsymbol{x}) = \mathcal{N}(\mu_{\boldsymbol{z}}, \Sigma_{\boldsymbol{z}})$$
$$(\mu_{\boldsymbol{z}}, \Sigma_{\boldsymbol{z}}) = g_\theta(\boldsymbol{x}) \quad \text{(typically a neural network)}$$
$$q_\theta(\boldsymbol{z}|\boldsymbol{x}) \approx p_\theta(\boldsymbol{z}|\boldsymbol{x})$$



Training

$$\mathcal{L}(\theta, \boldsymbol{x}) = \mathbb{E}_{q_\theta(\boldsymbol{z}|\boldsymbol{x})}\left[-\log p_\theta(\boldsymbol{x}|\boldsymbol{z})\right] + D_{\mathsf{KL}}\left(q_\theta(\boldsymbol{z}|\boldsymbol{x}) \,||\, p(\boldsymbol{z})\right) \quad \to \min$$

Kingma & Welling, "Auto-Encoding Variational Bayes", https://arxiv.org/abs/1312.6114v10

Variational Autoencoder

Ingredients

- Prior on z:

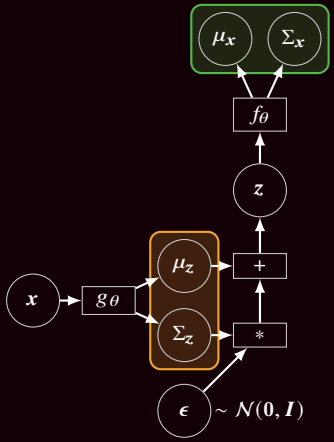
$$p(\boldsymbol{z}) = \mathcal{N}(\boldsymbol{0}, \boldsymbol{I})$$

- Generator:

$$p_\theta(\boldsymbol{x}|\boldsymbol{z}) = \mathcal{N}(\mu_{\boldsymbol{x}}, \Sigma_{\boldsymbol{x}})$$
$$(\mu_{\boldsymbol{x}}, \Sigma_{\boldsymbol{x}}) = f_\theta(\boldsymbol{z}) \quad \text{(typically a neural network)}$$

- Encoder:

$$q_\theta(\boldsymbol{z}|\boldsymbol{x}) = \mathcal{N}(\mu_{\boldsymbol{z}}, \Sigma_{\boldsymbol{z}})$$
$$(\mu_{\boldsymbol{z}}, \Sigma_{\boldsymbol{z}}) = g_\theta(\boldsymbol{x}) \quad \text{(typically a neural network)}$$
$$q_\theta(\boldsymbol{z}|\boldsymbol{x}) \approx p_\theta(\boldsymbol{z}|\boldsymbol{x})$$



Training

$$\mathcal{L}(\theta, \boldsymbol{x}) = \mathbb{E}_{q_\theta(\boldsymbol{z}|\boldsymbol{x})}\left[-\log p_\theta(\boldsymbol{x}|\boldsymbol{z})\right] + D_{\mathsf{KL}}\left(q_\theta(\boldsymbol{z}|\boldsymbol{x}) \,||\, p(\boldsymbol{z})\right) \to \min$$

Kingma & Welling, "Auto-Encoding Variational Bayes", https://arxiv.org/abs/1312.6114v10

β-VAE

$$\mathcal{L}(\theta, \boldsymbol{x}) = \mathbb{E}_{q_\theta(\boldsymbol{z}|\boldsymbol{x})} \left[-\log p_\theta(\boldsymbol{x}|\boldsymbol{z}) \right] + \beta D_{\mathsf{KL}} \left(q_\theta(\boldsymbol{z}|\boldsymbol{x}) \,||\, p(\boldsymbol{z}) \right) \quad \rightarrow \min$$

Higgins et al., "β-VAE: Learning Basic Visual Concepts [...]", ICLR 2017

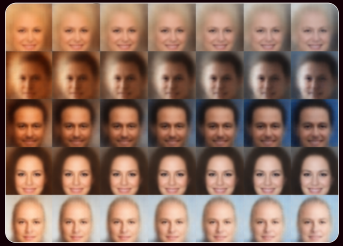
β-VAE

$$\mathcal{L}(\theta, \boldsymbol{x}) = \mathbb{E}_{q_\theta(\mathbf{z}|\boldsymbol{x})}\left[-\log p_\theta(\boldsymbol{x}|\mathbf{z})\right] + \beta D_{\mathsf{KL}}\left(q_\theta(\mathbf{z}|\boldsymbol{x}) \,||\, p(\mathbf{z})\right) \quad \rightarrow \min$$

Examples on CelebA

skin color

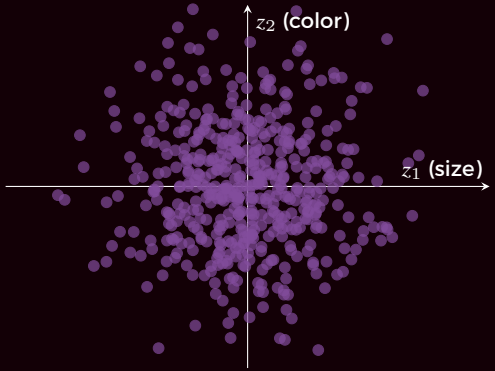
age/gender



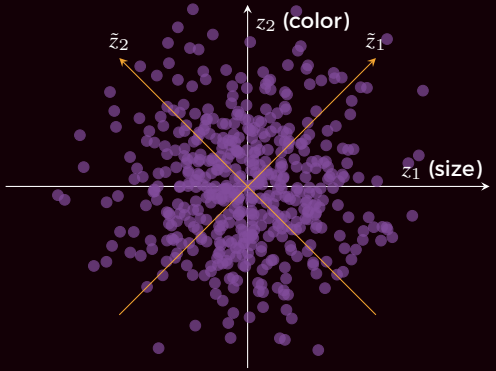
saturation

Higgins et al., "β-VAE: Learning Basic Visual Concepts [...]", ICLR 2017

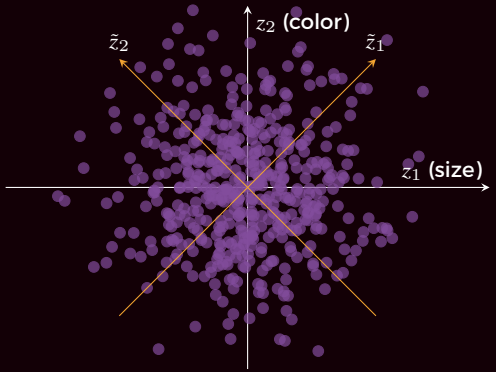
Chapter 2: Bad News



Proofs in Khemakhem et al., "Variational Autoencoders and Nonlinear ICA [...]", AISTATS 2020



Proofs in Khemakhem et al., "Variational Autoencoders and Nonlinear ICA [...]", AISTATS 2020

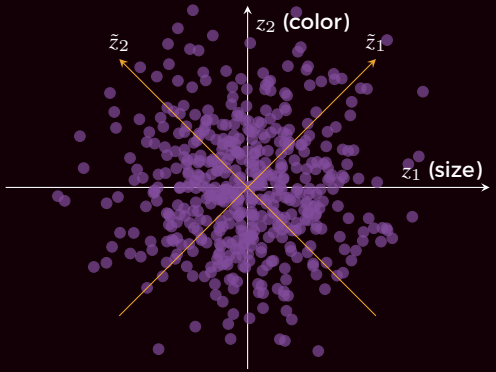


Gaussian Prior

$$p_\theta(\boldsymbol{x}) = \int p_\theta(\boldsymbol{x}|\boldsymbol{z}) p_\theta(\boldsymbol{z}) d\boldsymbol{z}$$

$$\boldsymbol{z} \sim p_\theta(\boldsymbol{z}) = \mathcal{N}(\boldsymbol{0}, \boldsymbol{I})$$

Proofs in Khemakhem et al., "Variational Autoencoders and Nonlinear ICA [...]", AISTATS 2020



Problem

Let $\tilde{z} = Mz$, where M is an orthogonal transformation:

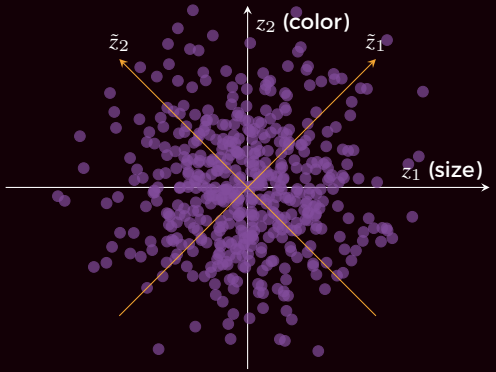
$$
\begin{aligned}
p(\tilde{z}) &= p(M^\top z)\,|\det M| \\
&= \frac{1}{(2\pi)^{d/2}} \exp\left(-\frac{1}{2}||M^\top z||^2\right) \\
&= \frac{1}{(2\pi)^{d/2}} \exp\left(-\frac{1}{2}||z||^2\right) \\
&= p(z)
\end{aligned}
$$

Gaussian Prior

$$p_\theta(\boldsymbol{x}) = \int p_\theta(\boldsymbol{x}|\boldsymbol{z})\, p_\theta(\boldsymbol{z})\, d\boldsymbol{z}$$

$$\boldsymbol{z} \sim p_\theta(\boldsymbol{z}) = \mathcal{N}(\boldsymbol{0}, \boldsymbol{I})$$

Proofs in Khemakhem et al., "Variational Autoencoders and Nonlinear ICA [...]", AISTATS 2020



Problem

Let $\tilde{z} = Mz$, where M is an orthogonal transformation:

$$\begin{aligned} p(\tilde{z}) &= p(M^\top z)\,|\det M| \\ &= \frac{1}{(2\pi)^{d/2}} \exp\left(-\frac{1}{2}||M^\top z||^2\right) \\ &= \frac{1}{(2\pi)^{d/2}} \exp\left(-\frac{1}{2}||z||^2\right) \\ &= p(z) \end{aligned}$$

Gaussian Prior

$$p_\theta(\boldsymbol{x}) = \int p_\theta(\boldsymbol{x}|\boldsymbol{z})\, p_\theta(\boldsymbol{z})\, d\boldsymbol{z}$$

$$\boldsymbol{z} \sim p_\theta(\boldsymbol{z}) = \mathcal{N}(\boldsymbol{0}, \boldsymbol{I})$$

Even Worse...

This applies to any factorial prior:

- transform z_i into uniform distribution via CDF $F_i(z_i)$
- further transform into Gaussian distribution: $\Phi^{-1}(F_i(z_i))$

Proofs in Khemakhem et al., "Variational Autoencoders and Nonlinear ICA [...]", AISTATS 2020

Alternative Proof in Locatello et al.

"[…] unsupervised learning of disentangled representations is fundamentally impossible without inductive biases on both the models and the data."

"[…] well-disentangled models seemingly cannot be identified without supervision."

Locatello et al., "Challenging Common Assumptions in the Unsupervised Learning of Disentangled Representations", ICML 2019

Chapter 3: Semi-Supervision to the Rescue!

Conditional Priors

Replace $p_\theta(z)$ with $p_\theta(z|\boldsymbol{u})$, where $\boldsymbol{u}$ is "an additionally observed variable":

$$p_\theta(\boldsymbol{x}, \boldsymbol{z}|\boldsymbol{u}) = p_\theta(\boldsymbol{x}|\boldsymbol{z})\, p_\theta(\boldsymbol{z}|\boldsymbol{u})$$

Khemakhem et al., "Variational Autoencoders and Nonlinear ICA [...]", AISTATS 2020

Conditional Priors

Replace $p_\theta(z)$ with $p_\theta(z|\boldsymbol{u})$, where $\boldsymbol{u}$ is "an additionally observed variable":

$$p_\theta(\boldsymbol{x}, \boldsymbol{z}|\boldsymbol{u}) = p_\theta(\boldsymbol{x}|\boldsymbol{z})\, p_\theta(\boldsymbol{z}|\boldsymbol{u})$$

In particular, model the prior as an exponential family:

$$p_\theta(\boldsymbol{z}|\boldsymbol{u}) = \prod_i \frac{Q_i(z_i)}{Z_i(\boldsymbol{u})} \exp\left[\sum_{j=1}^{k} T_{i,j}(z_i)\lambda_{i,j}(\boldsymbol{u})\right]$$

where $T_{i,j}$ are (learnable) sufficient statistics (k per component of $\boldsymbol{z}$) and $\lambda_{i,j}$ their (learnable) coefficients.

Khemakhem et al., "Variational Autoencoders and Nonlinear ICA [...]", AISTATS 2020

Identifiability Theorem 1

Under some mild conditions, the parameters θ of the true distribution are identifiable up to:

- a linear invertible transformation A
- point-wise nonlinearities

For $k = 1$:

$$(T_1^*(z_1^*), \ldots, T_n^*(z_n^*)) = A(T_1(z_1), \ldots, T_n(z_n))$$

Khemakhem et al., "Variational Autoencoders and Nonlinear ICA [...]", AISTATS 2020

Identifiability Theorem 1

Under some mild conditions, the parameters θ of the true distribution are identifiable up to:

- a linear invertible transformation A
- point-wise nonlinearities

For $k = 1$:

$$(T_1^*(z_1^*), \ldots, T_n^*(z_n^*)) = A(T_1(z_1), \ldots, T_n(z_n))$$

Identifiability Theorem 2

If further

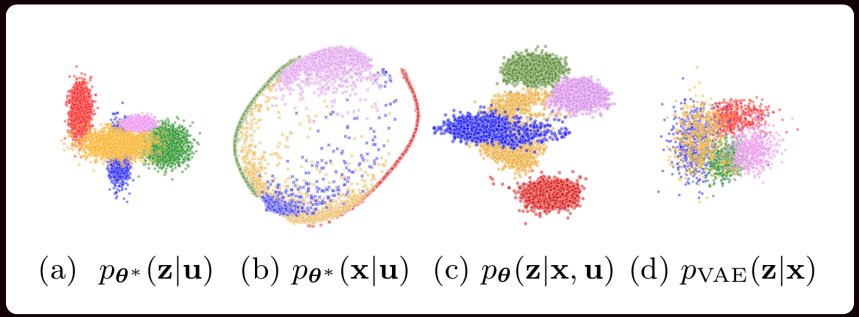
- $T_{i,j}$ are twice differentiable
- f (the generator function) has all second order cross derivatives

then A reduces to a permutation matrix (for $k = 1$ some weaker conditions can be applied).

For $k = 1$:

$$T_i^*(z_i^*) = T_{i'}(z_{i'})$$

Khemakhem et al., "Variational Autoencoders and Nonlinear ICA [...]", AISTATS 2020



(a) $p_{\boldsymbol{\theta}^*}(\mathbf{z}|\mathbf{u})$ (b) $p_{\boldsymbol{\theta}^*}(\mathbf{x}|\mathbf{u})$ (c) $p_{\boldsymbol{\theta}}(\mathbf{z}|\mathbf{x},\mathbf{u})$ (d) $p_{\mathrm{VAE}}(\mathbf{z}|\mathbf{x})$

Khemakhem et al., "Variational Autoencoders and Nonlinear ICA [...]", AISTATS 2020

Discussion

Chapter 1

- VAEs are well suited to learn data distributions with handcrafted latent priors
- β-VAEs (and variants) put extra emphasis on “disentanglement”
- posterior estimate $q_\theta(z|x)$ encodes data, potentially useful for downstream tasks

Discussion

Chapter 1

- VAEs are well suited to learn data distributions with handcrafted latent priors
- β-VAEs (and variants) put extra emphasis on “disentanglement”
- posterior estimate $q_\theta(z|x)$ encodes data, potentially useful for downstream tasks

Chapter 2

- true factors of variation can not be discovered with nonlinear models
- this result does not depend on the design of $p_\theta(z)$
- supervision or inductive biases needed

Discussion

Chapter 1

- VAEs are well suited to learn data distributions with handcrafted latent priors
- β-VAEs (and variants) put extra emphasis on “disentanglement”
- posterior estimate $q_\theta(\boldsymbol{z}|\boldsymbol{x})$ encodes data, potentially useful for downstream tasks

Chapter 2

- true factors of variation can not be discovered with nonlinear models
- this result does not depend on the design of $p_\theta(\boldsymbol{z})$
- supervision or inductive biases needed

Chapter 3

- conditional priors $p_\theta(\boldsymbol{z}|\boldsymbol{u})$ allow identification of factors of variation up to simple transform
- under some conditions, the transform reduces to a permutation plus pointwise nonlinearity
- unclear what the requirements on $\boldsymbol{u}$ are

Conditions of Theorem 1

- The set $\{\boldsymbol{x} \in \mathcal{X} \mid \phi_\epsilon(\boldsymbol{x}) = 0\}$ has measure zero.
- f is injective.
- $T_{i,j}$ are differentiable almost everywhere and $(T_{i,j})_{i \le j \le k}$ are linearly independent.
- There exist $nk + 1$ distinct points $\boldsymbol{u}^0, \ldots, \boldsymbol{u}^{nk}$ such that

$$L = (\lambda(\boldsymbol{u}^1) - \lambda(\boldsymbol{u}^0), \ldots, \lambda(\boldsymbol{u}^{nk} - \lambda(\boldsymbol{u}^0))$$

  is invertible.