

Modelos generativos baseados em fluxo

Aprendizagem de máquina probabilística

Madson Dias

madson.dias@lia.ufc.br

**Mestrado e Doutorado em Ciência da computação
Universidade Federal do Ceará**

Fortaleza, 2023

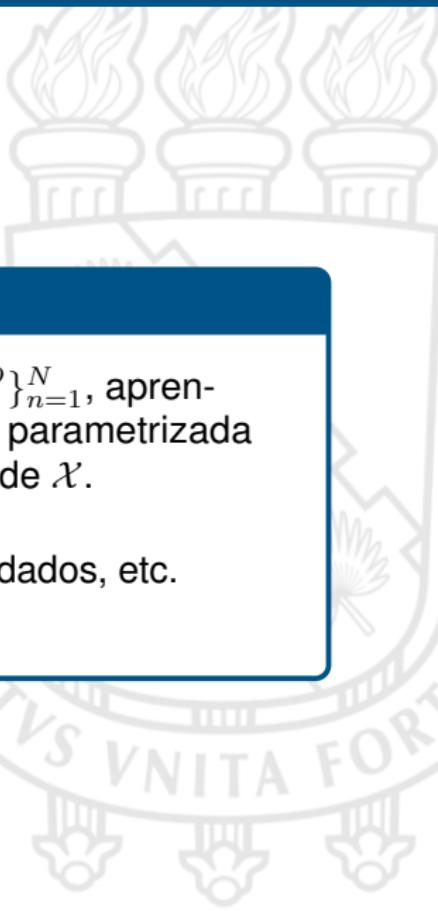
Sumário

- 1** Introdução
- 2** Regra da mudança de variáveis
- 3** Transformações e determinantes
- 4** Normalizing Flows
- 5** Arquiteturas de fluxo
- 6** Outros tópicos



Introdução

Modelo generativos probabilísticos

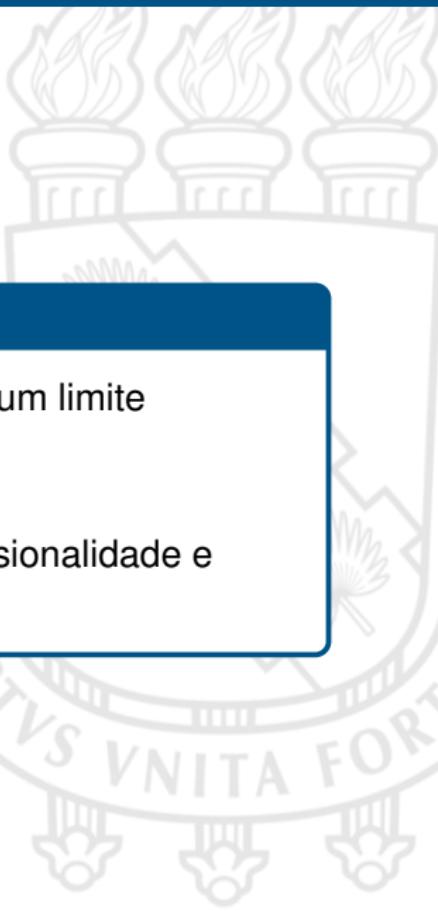


Probabilistic generative model (PGM)

Dado um conjunto de dados $\mathcal{X} = \{x_n \in \mathbb{R}^D\}_{n=1}^N$, aprender uma distribuição de probabilidade $p_X(x)$, parametrizada por θ , sobre uma variável aleatória X a partir de \mathcal{X} .

- Gerar amostras, avaliar novos pontos de dados, etc.
 - Mixture models, GANs, VAEs, etc.

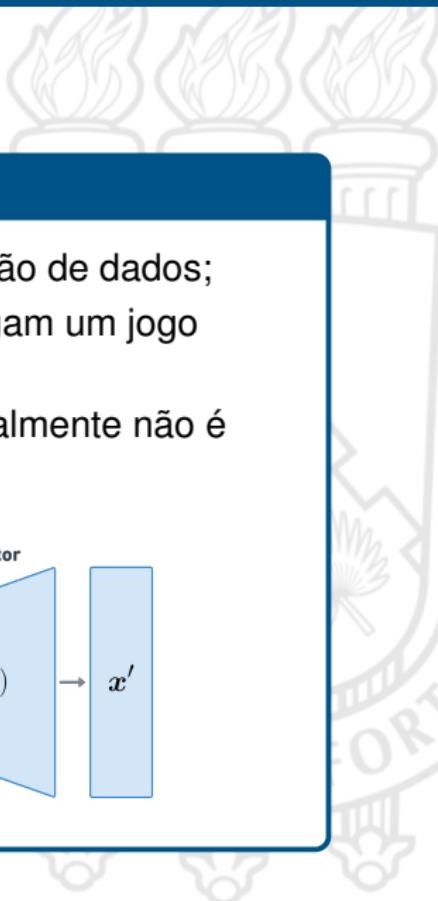
Modelo generativos probabilísticos



(Gaussian) Mixture models

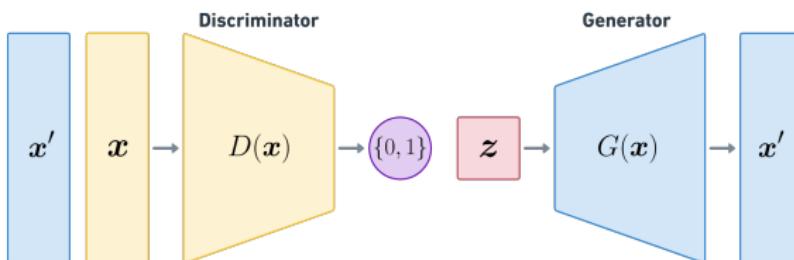
- Treinado via máxima verossimilhança ou um limite variacional dela;
 - Amostrar e avaliar $p_X(x)$ é simples;
 - O desempenho não acompanha a dimensionalidade e expressividade adicional.

Modelo generativos probabilísticos



Generative Adversarial Networks (GAN)

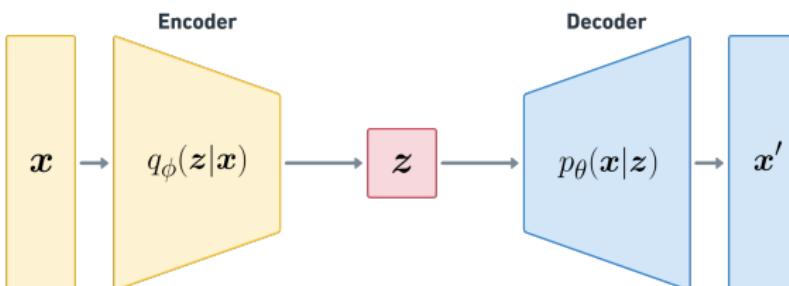
- Solução inteligente para modelar a geração de dados;
 - Dois modelos são treinados enquanto jogam um jogo **min-max**;
 - Amostrar de $p_X(x)$ é simples, avaliar geralmente não é possível.



Modelo generativos probabilísticos

Variational Autoencoders (VAE)

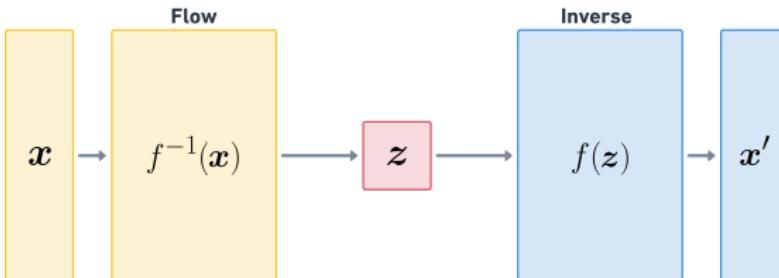
- Aprendizagem conjunta de um codificador e um decodificador;
- Maximiza o **evidence lower bound** (ELBO);
- Amostrar de $p_X(x)$ é simples, a avaliação é aproximada.



Modelo generativos probabilísticos

Normalizing Flows (NF)

- Construído por uma sequência de transformações inversíveis (bijetivas, difeomorfismos);
- Função de custo é o negativo da log-verossimilhança.
- Amostrar e avaliar de $p_X(x)$ são tarefas simples.



Regra da mudança de variáveis

Intuição

Intuição

- Simplificar problemas
- Variáveis originais são substituídas por funções de outras variáveis.
- A intenção é que quando expresso em novas variáveis, o problema se torne mais simples.

Caso univariado

Enunciado

- Dado um conjunto de dados $\mathcal{X} = \{x \in \mathbb{R}\}_{n=1}^N$, quer-se descobrir $p_X(x)$.
- Usando a regra da mudança de variáveis, podemos dizer que $x = f(z)$, com f diferenciável e inversível, tal que $z = f^{-1}(x) = g(x)$ e $z \sim p_Z(z)$, sendo $p_Z(z)$ conhecida.
- Desse modo:

$$\int_{\mathcal{X}} p_X(x) dx = \int_{\mathcal{Z}} p_Z(z) dz = 1.$$

- Temos então:

$$p_X(x) = p_Z(z) \left| \frac{d}{dx} f^{-1}(x) \right|. \quad (1)$$



Caso univariado

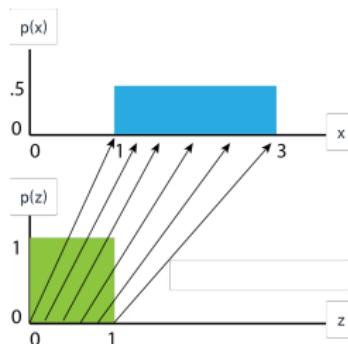
Exemplo

Seja $z \sim \mathcal{U}(0, 1)$ uma variável aleatória e $x = f(z) = 2z + 1$ uma transformação linear da variável de origem z , com inversa dada por

$$f^{-1}(x) = g(x) = \frac{x - 1}{2} = z.$$

Temos então que:

$$p_X(x) = 1 \left| \frac{1}{2} \right| \Leftrightarrow p_X(x) = 0.5.$$



Caso multivariado

Transformações lineares

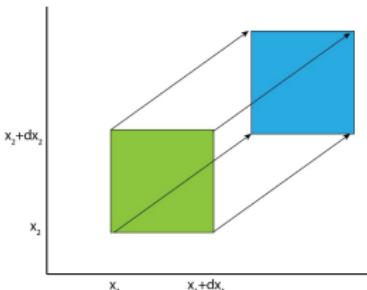
Transformações do tipo:

$$q(\mathbf{x}) = f^{-1}(\mathbf{x}) = \mathbf{A}\mathbf{x} + \mathbf{b}. \quad (2)$$

Transformações de deslocamento não mudam área

Transformação de escala
mudam área:

$$\mathbf{A} = \begin{bmatrix} a & b \\ c & d \end{bmatrix}.$$



UNIVERSIDADE
FEDERAL DO CEARÁ

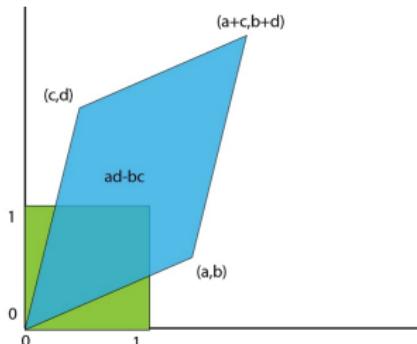
Caso multivariado



Transformações lineares

A nova área é $ad - bc$, que é valor absoluto do determinante da transformação linear.

O determinante é a taxa local e linearizada de mudança de volume de uma transformação.

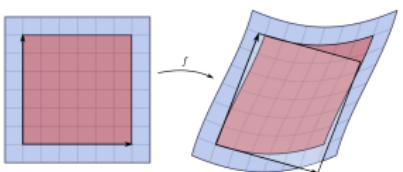


Desse modo, $p_X(x) = p_Z(z) |\det \mathbf{A}|$, para $f^{-1}(x) = \mathbf{A}x + b$.

Caso multivariado



Transformações não lineares



Inúmeros paralelogramos infinitesimais, com cada um representando a quantidade linear de distorção no espaço.

Matematicamente, essa mudança localmente linear no volume é dada pela matriz jacobiana:

$$\mathbf{J} = \begin{bmatrix} \frac{\partial f_1}{\partial x_1} & \cdots & \frac{\partial f_1}{\partial x_D} \\ \vdots & \ddots & \vdots \\ \frac{\partial f_S}{\partial x_1} & \cdots & \frac{\partial f_S}{\partial x_D} \end{bmatrix},$$

em que $f : \mathbb{R}^D \mapsto \mathbb{R}^S$ mapeia um ponto de entrada $\mathbf{x} \in \mathbb{R}^D$ para um vetor de saída $f(\mathbf{x}) \in \mathbb{R}^S$ e $\mathbf{J}_{ij} = \frac{\partial f_i}{\partial x_j}$.



Caso multivariado

Transformações não lineares

Com z e $x \in \mathbb{R}^D$, $x = f_{\theta}(z)$ e $z = f_{\theta}^{-1}(x)$. Temos:

$$p_X(x) = p_Z(z) \left| \det \frac{\partial f_{\theta}^{-1}}{\partial z} \right|, \quad (3)$$

ou ainda^{abc}:

$$\log p_X(x) = \log p_Z(z) - \log \left| \det \frac{\partial f_{\theta}}{\partial x} \right| \quad (4)$$

^aTeorema da função inversa: $\frac{d}{dy} f^{-1}(y) = \left(\frac{d}{dx} f(x) \right)^{-1}$.

^bRegra do determinante da matriz inversa: $\det(\mathbf{A}^{-1}) = [\det \mathbf{A}]^{-1}$.

^cDivisão de logaritmos: $\log(\frac{a}{b}) = \log a - \log b$



Transformações e determinantes



Resumo (parcial)

Método gerador (amostragem)

$$z \sim p_Z(z), \quad (5)$$

$$x = f_{\theta}(z). \quad (6)$$

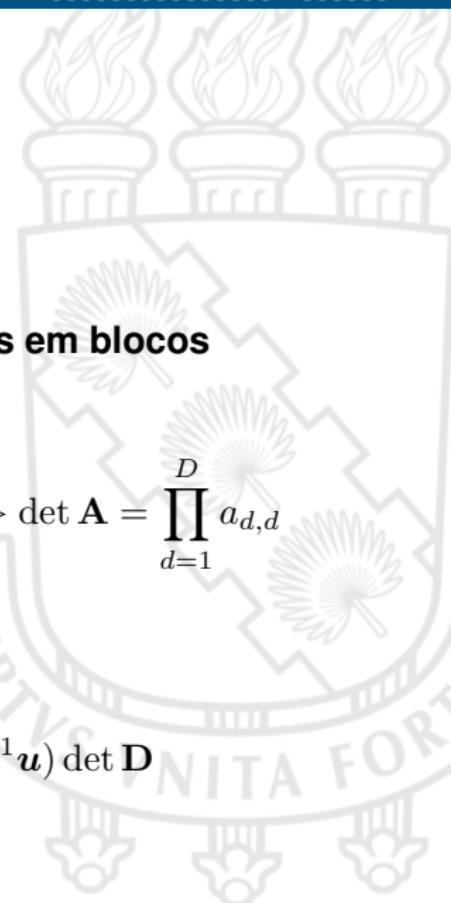
Log-verossimilhança (inferência)

$$\log p_X(x) = \log p_Z(z) - \log \left| \det \frac{\partial f_{\theta}}{\partial z} \right|. \quad (7)$$

Condições

1. Transformações tem que ser inversíveis,
2. O determinante jacobiano tem que ser fácil de computar^[1].

Determinantes tratáveis



Matrizes pequenas

- $\mathcal{O}(D^3)$ não é um problema.

Matrizes diagonais, triangulares, ou diagonais em blocos

$$\mathbf{A} = \begin{bmatrix} a_1 \\ \vdots \\ a_D \end{bmatrix}, \begin{bmatrix} a_{1,1} & & \\ & \ddots & \\ & & a_{D,D} \end{bmatrix} \text{ ou } \begin{bmatrix} B_1 & & \\ & \ddots & \\ & & B_K \end{bmatrix} \Rightarrow \det \mathbf{A} = \prod_{d=1}^D a_{d,d}$$

Lema do determinante da matriz

$$\mathbf{A} = \mathbf{D} + \mathbf{u}\mathbf{v}^\top \Rightarrow \det \mathbf{A} = (1 + \mathbf{v}^\top \mathbf{D}^{-1} \mathbf{u}) \det \mathbf{D}$$



Transformações comuns



Transformações elemento-a-elemento

$$f(\mathbf{z}) = [h(z_1), h(z_2), \dots, h(z_D)]^\top \quad (8)$$

$$\det \frac{\partial f}{\partial \mathbf{z}} = \prod_{d=1}^D h'(z_d) \quad (9)$$

- ✓ Transformações não lineares elemento-a-elemento.
- ✗ Sem mistura de variáveis.

Transformações comuns



Transformações lineares

$$f(\mathbf{z}) = \mathbf{A}\mathbf{z} + \mathbf{b}, \quad (10)$$

$$\det \frac{\partial f}{\partial \mathbf{z}} = \det \mathbf{A} \quad (11)$$

A matriz \mathbf{A} pode ser *diagonal* ($\prod_{d=1}^D a_d$), *triangular* ($\prod_{d=1}^D a_{d,d}$), *permutações* (-1^N), *ortogonal* (-1), *convolução* ou *fatorizações*.

- ✓ Combição afim de variáveis.
- ✗ Poder de representação limitado.



Transformações comuns

Transformações planares

$$f(\mathbf{z}) = \mathbf{z} + \mathbf{u}h(\mathbf{w}^\top \mathbf{z} + b), \quad (12)$$

$$\begin{aligned} \det \frac{\partial f}{\partial \mathbf{z}} &= \det (\mathbf{I}_D + \mathbf{u}\psi(\mathbf{z})^\top) \\ &= 1 + \mathbf{u}^\top \psi(\mathbf{z}) \end{aligned} \quad (13)$$

em que $\psi(\mathbf{z}) = h'(\mathbf{w}^\top \mathbf{z} + b)\mathbf{w}$, os parâmetros $\mathbf{u}, \mathbf{w} \in \mathbb{R}^D$ e $b \in \mathbb{R}$ são livres e $h : \mathbb{R} \rightarrow \mathbb{R}$ é uma não linearidade suave aplicada elemento a elemento.

- ✓ Transformações não lineares.
- ✗ Inversa não existe em certos h e configurações de parâmetros.



Transformações comuns



Transformações radiais

$$f(\mathbf{z}) = \mathbf{z} + \beta h(\alpha, r)(\mathbf{z} - \mathbf{z}^c), \quad (14)$$

$$\det \frac{\partial f}{\partial \mathbf{z}} = \left(1 + \frac{\beta}{h(\alpha, r)} + \frac{\beta r}{h'(\alpha, r)}\right) \left(1 + \frac{\beta}{h(\alpha, r)}\right)^{D-1} \quad (15)$$

em que $r = |\mathbf{z} - \mathbf{z}^c|$, $h(\alpha, r) = (\alpha + r)$ e os parâmetros do mapeamento são $\mathbf{z}^c \in \mathbb{R}^D$, $\alpha \in \mathbb{R}^+$, $\beta \in \mathbb{R}$.

-
- ✓ Transformações não lineares.
 - ✗ Só existe inversa sob certas configurações de parâmetros.

Transformações comuns



Transformações “acopladas” (*coupling*)

$$f(\mathbf{z}) = [\mathbf{z}_{\mathcal{A}}, \ \mathbf{h}(\mathbf{z}_{\mathcal{B}}; \boldsymbol{\sigma}(\mathbf{z}_{\mathcal{A}}))], \quad (16)$$

$$\det \frac{\partial f}{\partial \mathbf{z}} = \det \begin{bmatrix} \mathbf{I}_{|\mathcal{A}|} & \mathbf{0}_{|\mathcal{A}| \times |\mathcal{B}|} \\ \frac{\partial f_{\mathcal{B}}}{\partial \mathbf{z}_{\mathcal{A}}} & \text{diag}(\mathbf{h}'(\mathbf{z}_{\mathcal{B}}; \boldsymbol{\sigma}(\mathbf{z}_{\mathcal{A}}))) \end{bmatrix}. \quad (17)$$

Com $\mathbf{h} : \mathcal{B} \rightarrow \mathcal{B}$, $\boldsymbol{\sigma} : \mathcal{A} \rightarrow \mathcal{B}$, $\mathcal{A} = \{1 : d\}$ e $\mathcal{B} = \{d + 1 : D\}$.

- ✓ Permite transformações não lineares inversíveis.
- ✗ Dependente da função de acoplamento.



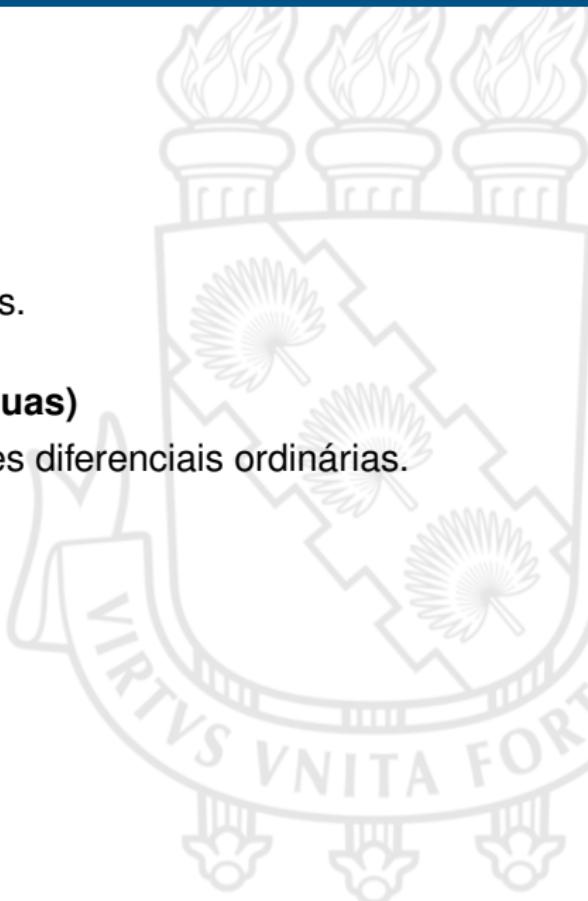
Outras transformações

Transformações residuais

- Utilização de redes neurais residuais.

Transformações infinitesimais (contínuas)

- Dependente da solução de equações diferenciais ordinárias.



Outras transformações

Transformações residuais

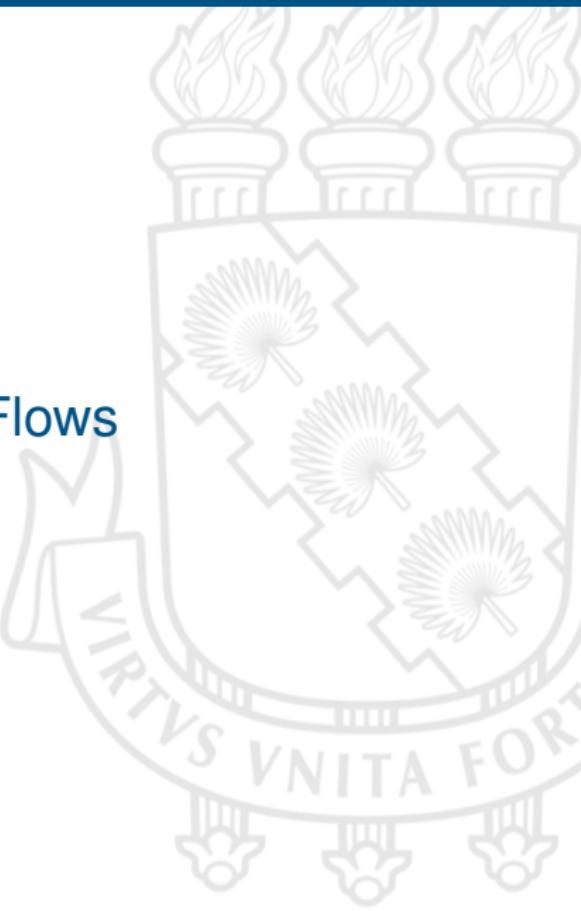
- Utilização de redes neurais residuais.

Transformações infinitesimais (contínuas)

- Dependente da solução de equações diferenciais ordinárias.

*E se ao invés de uma só transformação,
fossem encadeadas várias transformações?*

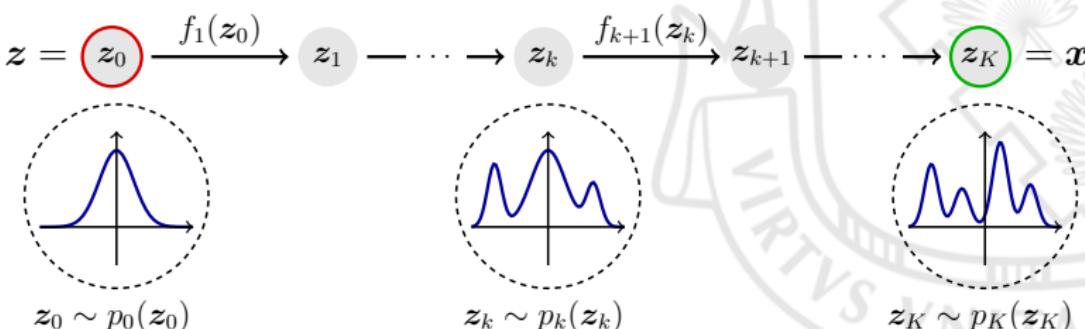
Normalizing Flows



Ideia

Com:

- $\mathbf{x} \triangleq \mathbf{z}_K \sim p_K(\mathbf{z}_K);$
- $\mathbf{z} \triangleq \mathbf{z}_0 \sim p_0(\mathbf{z}_0);$ e
- $\mathbf{z}_k = f_k(\mathbf{z}_{k-1}), \quad \forall k = 1, 2, \dots, K.$



Normalizing flows



Método gerador (amostragem)

$$\mathbf{z} \sim p_Z(\mathbf{z}), \quad (18)$$

$$\mathbf{x} = f_K \circ \dots \circ f_2 \circ f_1(\mathbf{z}). \quad (19)$$

Log-verossimilhança (inferência)

$$\log p(\mathbf{x}) = \log p_0(\mathbf{z}_0) - \sum_{k=1}^K \log \left| \det \frac{\partial \mathbf{z}_k}{\partial \mathbf{z}_{k-1}} \right|, \quad (20)$$

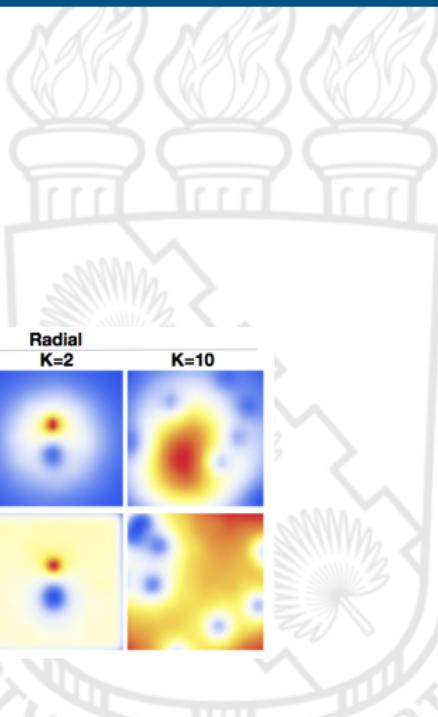
Treinamento (estimação de máxima verossimilhança)

$$L(\mathcal{X}) = -\frac{1}{|\mathcal{X}|} \sum_{\mathbf{x} \in \mathcal{X}} \log p(\mathbf{x}). \quad (21)$$

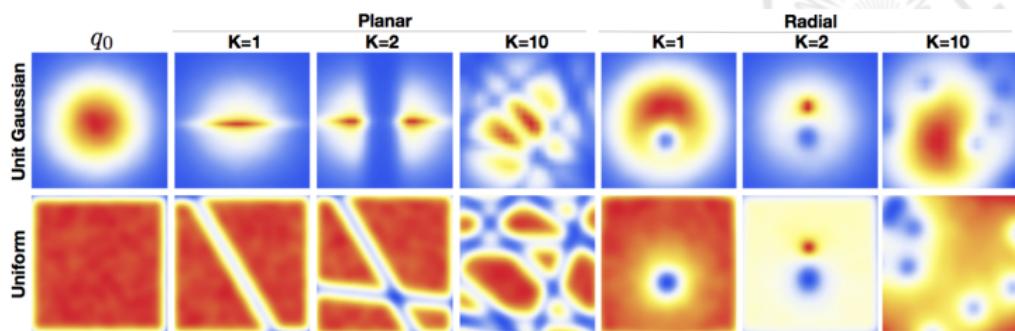
Arquiteturas de fluxo



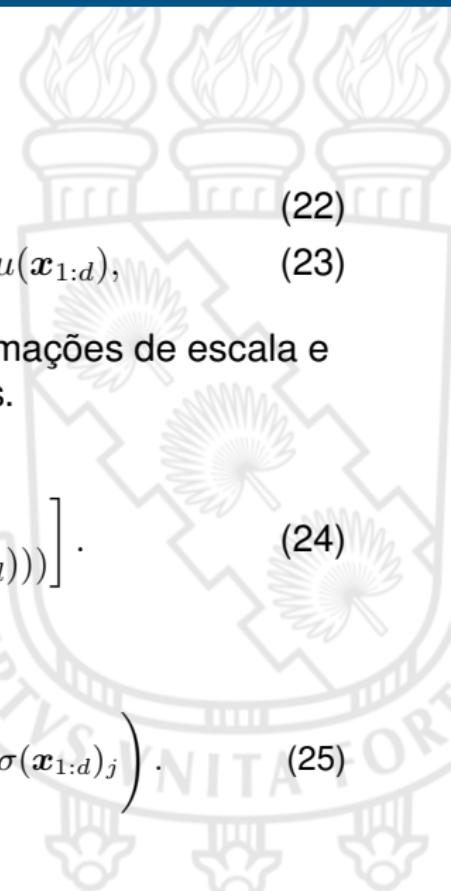
Fluxos simples



Fluxos planares e fluxos radiais



Real-NVP [2]



Transformação

$$\mathbf{y}_{1:d} = \mathbf{x}_{1:d}, \quad (22)$$

$$\mathbf{y}_{d+1:D} = \mathbf{x}_{d+1:D} \odot \exp(\sigma(\mathbf{x}_{1:d})) + \mu(\mathbf{x}_{1:d}), \quad (23)$$

em que $\sigma, \mu : \mathbb{R}^d \rightarrow \mathbb{R}^{D-d}$ representam transformações de escala e deslocamento, parametrizadas por redes neurais.

Matriz Jacobiana

$$\frac{\partial \mathbf{y}}{\partial \mathbf{x}} = \begin{bmatrix} \mathbf{I}_d & \mathbf{0}_{d \times (D-d)} \\ \frac{\partial \mathbf{y}_{d+1:D}}{\partial \mathbf{x}_{1:d}} & \text{diag}(\exp(\sigma(\mathbf{x}_{1:d}))) \end{bmatrix}. \quad (24)$$

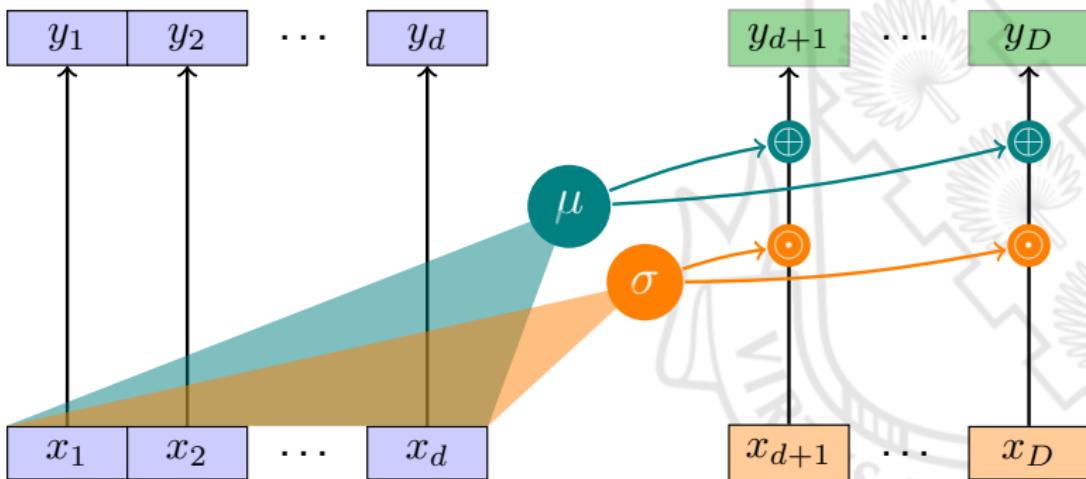
Determinante

$$\det \frac{\partial \mathbf{y}}{\partial \mathbf{x}} = \prod_{j=1}^{D-d} \exp(\sigma(\mathbf{x}_{1:d}))_j = \exp \left(\sum_{j=1}^{D-d} \sigma(\mathbf{x}_{1:d})_j \right). \quad (25)$$



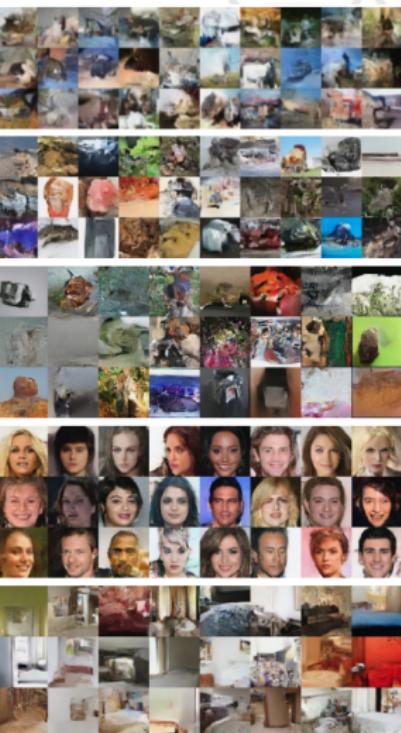
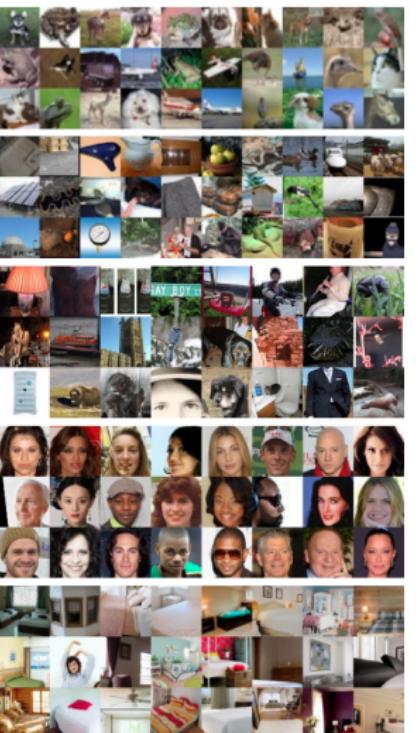
Real-NVP

Coupling layer

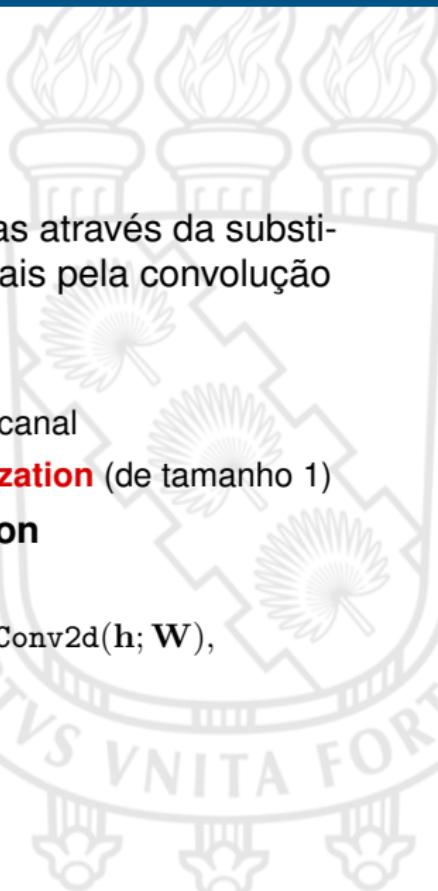


Real-NVP

Exemplos

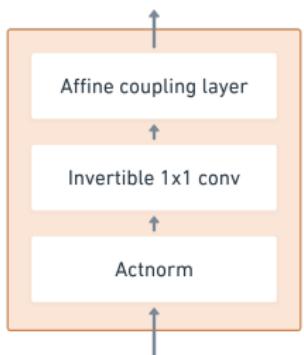
UNIVERSIDADE
FEDERAL DO C...

Glow [3]



Descrição

Estende o NICE e RealNVP, e simplifica estruturas através da substituição da permutação reversa na ordem dos canais pela convolução inversível 1×1 .



Activation normalization

- Transformação afim por canal
- Similar a **batch normalization** (de tamanho 1)

Invertible 1×1 convolution

- Entrada: $\mathbf{h} \in h \times w \times c$
- Transformação: $f(\mathbf{h}) = \text{Conv2d}(\mathbf{h}; \mathbf{W})$,
em que $\mathbf{W} \in c \times c$

Affine coupling layer

- Similar ao RealNVP



[3] Kingma e Dhariwal, "Glow: Generative flow with invertible 1×1 convolutions", 2018

Glow

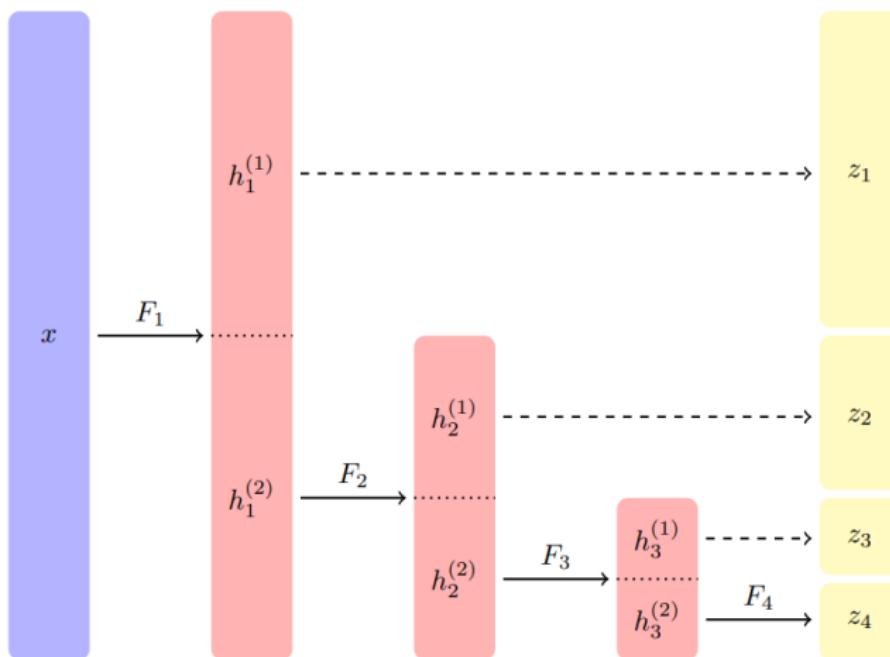
Transformações

Description	Function	Reverse Function	Log-determinant
Actnorm. See Section 3.1.	$\forall i, j : \mathbf{y}_{i,j} = \mathbf{s} \odot \mathbf{x}_{i,j} + \mathbf{b}$	$\forall i, j : \mathbf{x}_{i,j} = (\mathbf{y}_{i,j} - \mathbf{b}) / \mathbf{s}$	$h \cdot w \cdot \text{sum}(\log \mathbf{s})$
Invertible 1×1 convolution. $\mathbf{W} : [c \times c]$. See Section 3.2.	$\forall i, j : \mathbf{y}_{i,j} = \mathbf{W}\mathbf{x}_{i,j}$	$\forall i, j : \mathbf{x}_{i,j} = \mathbf{W}^{-1}\mathbf{y}_{i,j}$	$h \cdot w \cdot \log \det(\mathbf{W}) $ or $h \cdot w \cdot \text{sum}(\log \mathbf{s})$ (see eq. (10))
Affine coupling layer. See Section 3.3 and (Dinh et al., 2014)	$\mathbf{x}_a, \mathbf{x}_b = \text{split}(\mathbf{x})$ $(\log \mathbf{s}, \mathbf{t}) = \text{NN}(\mathbf{x}_b)$ $\mathbf{s} = \exp(\log \mathbf{s})$ $\mathbf{y}_a = \mathbf{s} \odot \mathbf{x}_a + \mathbf{t}$ $\mathbf{y}_b = \mathbf{x}_b$ $\mathbf{y} = \text{concat}(\mathbf{y}_a, \mathbf{y}_b)$	$\mathbf{y}_a, \mathbf{y}_b = \text{split}(\mathbf{y})$ $(\log \mathbf{s}, \mathbf{t}) = \text{NN}(\mathbf{y}_b)$ $\mathbf{s} = \exp(\log \mathbf{s})$ $\mathbf{x}_a = (\mathbf{y}_a - \mathbf{t}) / \mathbf{s}$ $\mathbf{x}_b = \mathbf{y}_b$ $\mathbf{x} = \text{concat}(\mathbf{x}_a, \mathbf{x}_b)$	$\text{sum}(\log(\mathbf{s}))$



Glow

Arquitetura multi-escala



Glow



Exemplos



UNIVERSIDADE
FEDERAL DO CEARÁ

Fluxos autoregressivos

Fluxos auto-regressivos são uma classe de de NF na qual, dada uma amostra de dados sequênciais $\mathbf{x} \in \mathbb{R}^D$, cada dimensão depende somente das dimensões anteriores. Desse modo,

$$\mathbf{y}_d = f(\mathbf{x}_{1:d}) \Rightarrow \det \frac{\partial \mathbf{y}}{\partial \mathbf{x}} = \prod_{d=1}^D \frac{\partial f(\mathbf{x}_{1:d})}{\partial x_d}. \quad (26)$$

Além disso

$$p(\mathbf{x}) = \prod_{d=1}^D p(x_d | x_1, \dots, x_{d-1}) = \prod_{d=1}^D p(x_d | \mathbf{x}_{1:d-1}). \quad (27)$$

Pode-se fazer $p(x_d | \mathbf{x}_{1:d-1}) = \mathcal{N}(h_d(\mathbf{x}_{1:d-1}), (g_d(\mathbf{x}_{1:d-1}))^2)$, com parâmetros computados através de redes neurais.



Masked auto-regressive Flows (MAF, [4])

Utiliza restrições autoregressivas para modelar densidades condicionais parametrizadas como Gaussianas univariadas. A probabilidade condicional de x_d é dada por

$$p(x_d | \mathbf{x}_{1:d-1}) = \mathcal{N}(x_d | \mu_d(\mathbf{x}_{1:d-1}), (\exp(\alpha_d(\mathbf{x}_{1:d-1})))^2), \quad (28)$$

Transformação

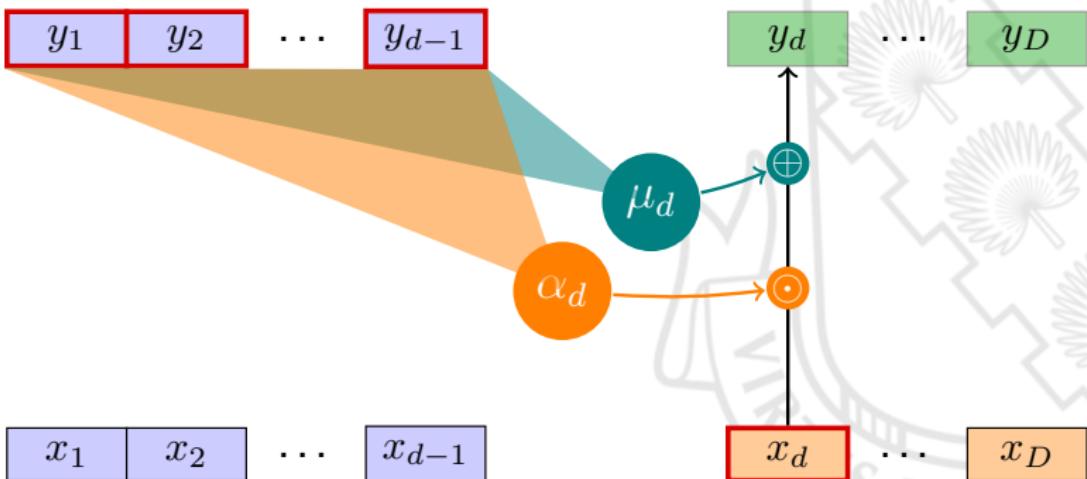
$$y_d = x_d \exp(\alpha_d(\mathbf{y}_{1:d-1})) + \mu_d(\mathbf{y}_{1:d-1}). \quad (29)$$

Jacobiano

$$\det \frac{\partial \mathbf{y}}{\partial \mathbf{x}} = \prod_{d=1}^D \exp(\alpha_d(\mathbf{y}_{1:d-1})) = \exp\left(\sum_{d=1}^D \alpha_d(\mathbf{y}_{1:d-1})\right). \quad (30)$$

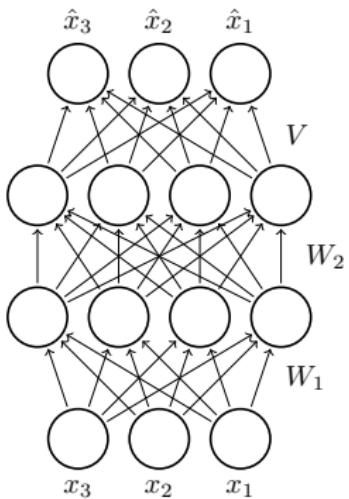
Masked auto-regressive Flows (MAF)

Transformação



Masked auto-regressive Flows (MAF)

Masked Autoencoder for Distribution Estimation (MADE)



autoencoder

x

masks

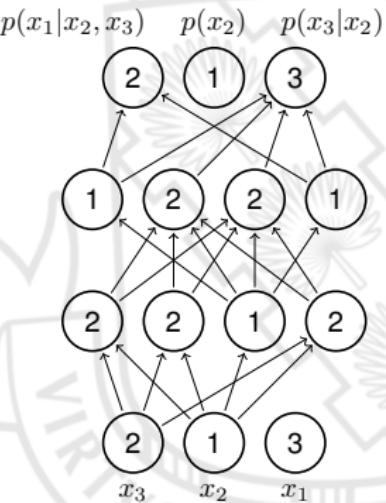
→

MADE

$$M_V = \begin{array}{|c|c|c|c|} \hline & \textcolor{black}{\boxed{}} & \textcolor{black}{\boxed{}} & \textcolor{black}{\boxed{}} \\ \hline \textcolor{white}{\boxed{}} & \textcolor{black}{\boxed{}} & \textcolor{black}{\boxed{}} & \textcolor{white}{\boxed{}} \\ \hline \end{array}$$

$$M_{W_2} =$$

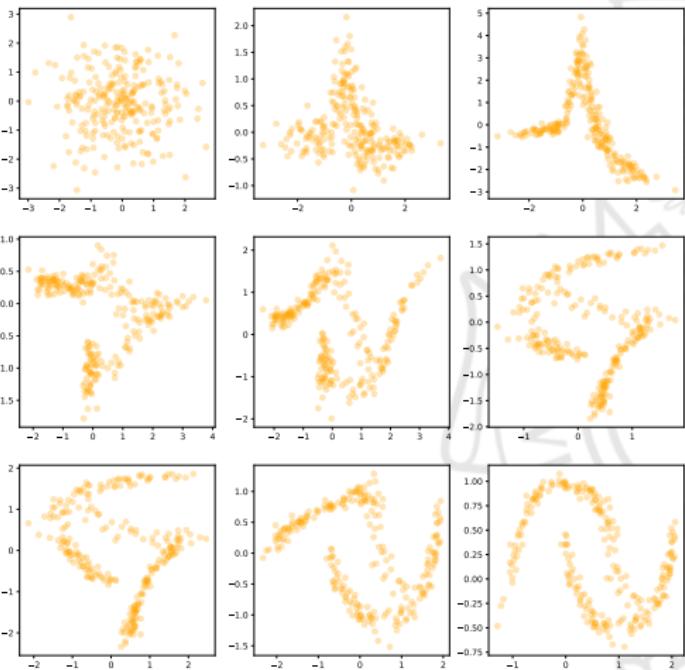
$$M_{W_1} =$$



UNIVERSIDADE
FEDERAL DO CEARÁ

Masked auto-regressive Flows (MAF, [5])

Exemplos



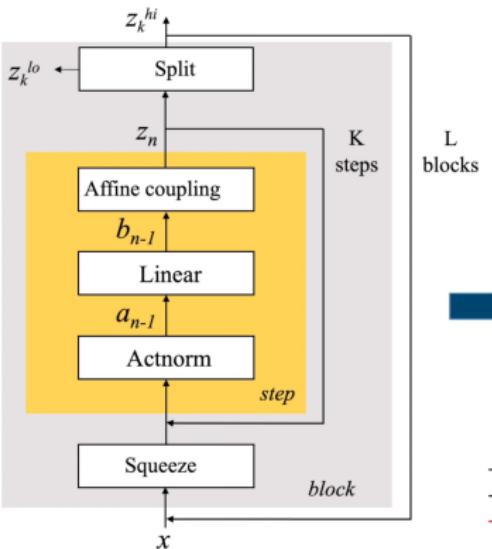
[5] Germain et al., "Made: Masked autoencoder for distribution estimation", 2015



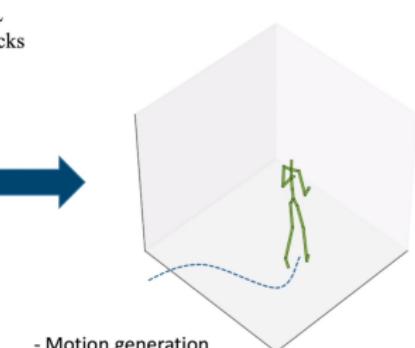
Motion Glow (MoGlow, [6])

Descrição

- Utiliza o Glow para geração de movimentos
- Utiliza LSTM para aprender dependências temporais longas



MoGlow



- Motion generation
- Added control by conditioning $p(x|c)$
- New auto-regressive architecture with RNNs



Introdução
oooooooo

Regra da mudança de variáveis
oooooooooooo

Transformações e determinantes
oooooooooooo

Normalizing Flows
ooo

Arquiteturas de fluxo
oooooooooooooooooooo

Outros tópicos
●○○○○○

Outros tópicos

Outros tópicos

Flow Gaussian Mixture Model (FlowGMM, [7])

- Dados rotulados: $\mathcal{D}_u = \{\mathbf{x}_n\}_{n=1}^N$
- Dados não rotulados: $\mathcal{D}_\ell = \{(\mathbf{x}_m, y_m)\}_{m=1}^M$, tal que $y_m \in \mathcal{C}$
- Distribuição para o espaço latente de \mathcal{D}_ℓ

$$p_{\mathcal{Z}}(z|y=k) = \mathcal{N}(z|\mu_k, \Sigma_k). \quad (31)$$

em que μ_k é a média e Σ_k a matriz de covariância da classe k .

- Distribuição marginal de z é uma mistura de gaussianas:

$$p_{\mathcal{Z}}(z) = \frac{1}{C} \sum_{k=1}^C \mathcal{N}(z|\mu_k, \Sigma_k). \quad (32)$$

em que C é o número de classes.

Outros tópicos

Flow Gaussian Mixture Model (FlowGMM)

- A verossimilhança do conjunto de dados \mathcal{D}_f é

$$p_{\mathcal{X}}(x|y=k) = \mathcal{N}(f_{\theta}(x)|\mu_k, \Sigma_k) \left| \det \left(\frac{\partial f}{\partial x} \right) \right|, \quad (33)$$

e a verossimilhança dos dados não rotulados é

$$p_{\mathcal{X}}(x) = \sum_k p_{\mathcal{X}}(x|y=k)p(y=k). \quad (34)$$

- A função de custo é a verossimilhança conjunta dos dados rotulados e não rotulados:

$$p_{\mathcal{X}}(\mathcal{D}_u, \mathcal{D}_{\ell} | \theta) = \prod_{x_n \in \mathcal{D}_u} p_{\mathcal{X}}(x_n) \prod_{(x_m, y_m) \in \mathcal{D}_{\ell}} p_{\mathcal{X}}(x_m, y_m), \quad (35)$$



Outros tópicos

Latent Outlier Exposure for Anomaly Detection with Contaminated Data (LOE)^[8]

- Treinar um detector de anomalias na presença de contaminação;
- Realiza inferências de anomalia enquanto atualiza parâmetros;
- Combina dois custos que compartilham parâmetros, um deles para dados normais (\mathcal{L}_n^θ) e outro para dados anômalos (\mathcal{L}_a^θ):

$$\mathcal{L}(\theta, \mathbf{y}) = \sum_{n=1}^N (1 - y_n) \mathcal{L}_n^\theta(\mathbf{x}_n) + y_n \mathcal{L}_a^\theta(\mathbf{x}_n), \quad (36)$$

em que $y_n = 1$ para anomalias e $y_n = 0$ para dados normais.

- Modelo é então treinado utilizando *block coordinate descent* para atualização conjunta dos parâmetros e rotulação as anomalias mais prováveis (latentes).



Outros tópicos

Latent Outlier Exposure for Anomaly Detection with Contaminated Data

- Considera uma parcela $\alpha \in [0, 1]$ de anomalias.
- Escore de anomalia para o treinamento:

$$S_n^{train} = \mathcal{L}_n^\theta(\mathbf{x}_n) - \mathcal{L}_a^\theta(\mathbf{x}_n).$$

Algorithm 1 Training process of LOE

Input: Contaminated training set \mathcal{D} (α_0 anomaly rate)
hyperparameter α

Model: Deep anomaly detector with parameters θ
foreach Epoch **do**

foreach Mini-batch \mathcal{M} **do**

 Calculate the anomaly score S_i^{train} for $\mathbf{x}_i \in \mathcal{M}$
 Estimate the label y_i given S_i^{train} and α
 Update the parameters θ by minimizing $\mathcal{L}(\theta, \mathbf{y})$

end

end



Modelos generativos baseados em fluxo

Aprendizagem de máquina probabilística

Madson Dias

madson.dias@lia.ufc.br

**Mestrado e Doutorado em Ciência da computação
Universidade Federal do Ceará**

Fortaleza, 2023