Variational Autoencoders: Disentangled Representations and Other Applications

Ademir Tomaz
Escola de Matemática Aplicada
Fundação Getúlio Vargas
Rio de Janeiro, RJ
B41259@fgv.edu.br

Carlos Souza
Escola de Matemática Aplicada
Fundação Getúlio Vargas
Rio de Janeiro, RJ
B39056@fgv.edu.br

Juliana Carvalho
Escola de Matemática Aplicada
Fundação Getúlio Vargas
Rio de Janeiro, RJ
B41260@fgv.edu.br

Raphael Levy
Escola de Matemática Aplicada
Fundação Getúlio Vargas
Rio de Janeiro, RJ
B41257@fgv.edu.br

Abstract

This paper provides a concise overview of disentangled representations, a fundamental concept in machine learning, with a focus on Variational Autoencoders (VAEs). Disentangled representations aim to capture the underlying factors of variation in the data, where different dimensions in the learned representation correspond to distinct generative factors. This enables a better understanding and manipulation of the data. In the context of VAEs, this disentanglement is particularly crucial as VAEs encode input data into a lower-dimensional latent space to capture these essential factors. In addition to providing a robust method for data compression, VAEs also pave the way for improved interpretability and enhanced performance in subsequent tasks.

This paper also presents a comparative experiment between two types of VAEs: Conditional Variational Autoencoder (CVAE) and β-VAE. The objective of the experiment is to evaluate the efficiency of these models in terms of disentanglement and data reconstruction capability. The results obtained with both models will be compared to determine which one is more suitable for the proposed task. The experiment will be conducted and evaluated using a relevant and appropriate dataset for comparative analysis.

The significance of this research lies in understanding and appropriately selecting VAE models for specific applications, as well as advancing the field of disentangled representations in machine learning overall.

Keywords: Disentangled Representations, Variational Autoencoders, CVAE, beta-VAE, Machine Learning, Data Compression, Interpretability, Generative Models.

1 Introdução:

De forma simplificada, a **Disentangled Representation** (Representações Desembaraçadas) é uma técnica de aprendizado não supervisionado que separa cada característica em variáveis restritas e as codifica em dimensões distintas(independentes). Isso permite capturar informações sensíveis e úteis, além de compreender as relações causais entre as variáveis. Por exemplo, na imagem abaixo, é possível observar como a alteração de dimensões específicas afeta a imagem original, o que ajuda a entender as variações específicas.
Por outro lado, os **CVAEs** (Conditional Variational Autoencoders) são definidos como um tipo especial de VAE (Variational Autoencoder) que serão utilizados em nossos experimentos. A principal diferença reside no fato de que informações de rótulo (label) são injetadas tanto na fase de codificação quanto na de decodificação. Isso se faz necessário, pois utilizando apenas o VAE, não temos controle sobre o tipo de dado que ele irá gerar. Por exemplo, ao utilizar o conjunto de dados MNIST e tentar gerar imagens fornecendo $Z \sim \mathcal{N}(0, 1)$ como entrada para o decodificador, o VAE pode produzir dígitos aleatórios. Embora as imagens possam ser bem geradas se o autoencoder estiver bem treinado, não podemos direcionar explicitamente o VAE para gerar uma imagem de um dígito específico. É aí que entra o CVAE: fornecendo a entrada X, que é o rótulo da imagem, desejamos obter como saída Y, que é a imagem gerada. Portanto, dado a observação y, o valor z é extraído de uma distribuição a priori $P_\theta(z|y)$ e x é gerado a partir da distribuição $P_\theta(x|y, z)$. No VAE convencional, a priori é definida como $P_\theta(z)$ e a saída é gerada por $P_\theta(x|z)$.
Dessa forma, o codificador (encoder) tenta aprender $q_\phi(z|x, y)$, o que equivale a aprender representações ocultas dos dados X ou codificar X na representação oculta condicionada a y. Enquanto isso, o decodificador (decoder) tenta aprender $P_\theta(X|z, y)$, decodificando a representação oculta para o espaço de entrada, condicionado por y.

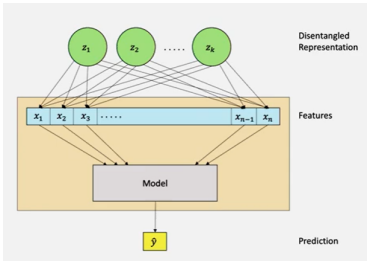

Figura 1: Disentangled representation

2 Fundamentação Teórica

2.1 Descrição de Disentangled Representations:

O "desemaranhamento"funciona por meio de **Representações Latentes** . [1]

Agora, tomando as definições necessárias, retiradas de [3]:

$\mathcal{X}$: variáveis observadas

$\mathcal{Z}$: representações latentes

$\mathcal{Y}$: domínio de saídas do modelo

O aprendizado do modelo é dado por uma função $f : \mathcal{X} \to \mathcal{Y}$, sendo que f pode ser separado em dois componentes:

$$f : E_\phi \circ D_\phi,$$

onde $E_\phi : \mathcal{X} \to \mathcal{Z}$ é o **ENCODER** e $D_\phi : \mathcal{Z} \to \mathcal{Y}$ é o **DECODER**, mapeando os dados para uma representação latente intermediária e a representação para os outputs, respectivamente.

Assim, o objetivo desse aprendizado é aprender uma boa representação, que terá equivariâncias, invariâncias e simetrias para diferentes alterações. Considere as operações:

1. **Simetria:** uma simetria Ω é uma transformação que mantém aspectos dos dados de entrada, como a categoria de um objeto por exemplo, que não se altera quando deslocamos esse objeto.
2. **Equivariância:** um mapeamento E_ϕ é equivariante com respeito a Ω se existe uma transformação $\omega \in \Omega$ de um input $X \in \mathcal{X}$ que afeta o output $Z \in \mathcal{Z}$ da mesma maneira, ou seja, existe um mapeamento $M_\omega : \mathbb{R}^d \to \mathbb{R}^d$ tal que $E_\phi(M_\omega \circ X) = M_\omega \circ E_\phi(X) \; \forall \, \omega \in \Omega$.
3. **Invariância:** é um caso especial da equivariância, onde M é a identidade, e $E_\phi(M_\omega \circ X) = E_\phi(X) \; \forall \, \omega \in \Omega$

Para explicações mais profundas vá até ao apêndice 8.1.

A representação $\mathcal{Z}$ será "desemaranhada"se puder ser decomposta em $Z = Z_1 \times \ldots \times Z_n$, onde uma transformação ω aplicada a Z_i resulta em uma transformação equivalente no domínio $\mathcal{X}$, mantendo os outros aspectos de Z (ou seja, $Z_j, j \neq i$) inalterados. Essa definição deve satisfazer:

[1] São dados comprimidos após serem passados pelo encoder, ou seja, transformando uma informação complexa em uma versão "mais simples", facilitando seu processamento e análise.

1. **Modularidade:** cada dimensão latente codifica apenas um fator gerador, sendo afetada apenas pelas mudanças nesse fator e não por outros. Cada código latente está associado a um único fator gerador e é separável dos outros.

2. **Compacidade:** cada fator gerador é codificado por uma única dimensão latente. Uma dimensionalidade excessivamente alta do espaço latente pode levar à codificação de informações ruidosas ou redundantes.

3. **Explicitude:** todos os valores dos fatores geradores podem ser decodificados da representação por meio de uma transformação linear. Isso implica que a representação desemaranhada captura informações completas sobre os fatores geradores e que essas informações podem ser decodificadas linearmente.

Explicações mais profundas estão no apêndice 8.2.

3 Metodologia

3.1 CVAE $\times$ βVAE

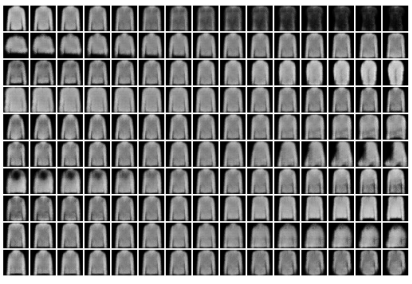

Figura 2: β-VAE_h

Para execução dos experimentos vamos fazer comparações computacionais (tempo de processamento), e da qualidade de diferentes implementações de VAEs, (no caso usaremos: CVAE e βVAE) Utilizando a bases de dados:"Fashion MNIST"o qual consiste em um conjunto de imagens de roupas

Figura 3: Dados de amostra originais

Esses conjuntos de dados são eficientes para avaliar modelos de aprendizado de máquina devido à sua simplicidade e, ao mesmo tempo, possibilidade de geração de modelos complexos. Mas antes disso vamos fazer uma breve introdução sobre VAEs:

3.2 Variational Autoencoders (VAE)

VAEs são baseados em métodos variacionais da estatística Bayesiana. Em vez de mapear o input para um vetor fixo, eles mapeiam para uma distribuição parametrizada por θ (p_θ). Ao contrário dos autoencoders tradicionais, os VAEs possuem dois vetores, um para a média e outro para o desvio padrão.

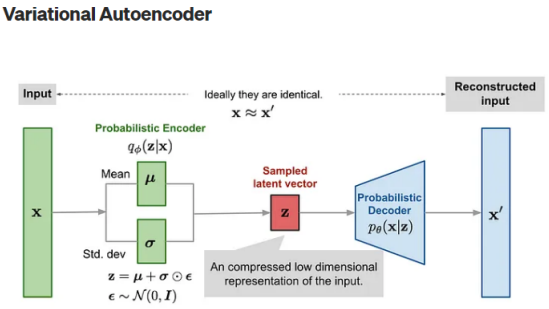

Figura 4: Variational Autoencoder

O objetivo dos VAEs é maximizar a probabilidade de gerar dados reais. Ou seja, o autoencoder visa aprender a log-verossimilhança marginal do dados em tal processo generativo. Isso pode ser formalizado como:

$$max_{\phi,\theta} \mathbb{E} x \sim q_\phi(z|x) \log p_\theta(x|z)$$

onde θ e ϕ são os parâmetros da codificação e da decodificação, respectivamente. Aplicando a

[2]ELBO: Evidence Lower BOund é uma função de custo utilizada em autoencoders variacionais (VAEs) que serve como um limitante inferior na log-verossimilhança dos dados.

ELBO[2], a equação acima pode ser escrita como:

$$\log p_\theta(x|z) \geq$$
$$\mathcal{L}(\theta, \phi, x, z) = \mathbb{E}z \sim q_\phi(z|x) \log p_\theta(x|z)] - D_{KL}(q_\phi(z|x)||p_\theta(z))$$

OBS: Para a otimização dessa equação, na prática, fazemos algumas suposições. Como as distribuições prior $p_\theta(z)$ e posterior $q_\phi(z|x)$

são parametrizadas como Gaussianas com uma matriz de covariância diagonal; $p(z)$ é tipicamente definida como a Gaussiana $\mathcal{N}(0, 1)$. Dessa forma é possível aplicar o 'truque de reparametrização' para estimar os gradientes do limite inferior com relação aos parâmetros ϕ, onde cada variável aleatória $z_i \sim q_\phi(z_i|x) = N(\mu_i, \sigma_i)$ é parametrizada como uma transformação diferenciável de uma variável de ruído: $\epsilon \sim \mathcal{N}(0, 1)$: $z_i = \mu_i + \sigma_i.\epsilon$

3.3 Conditional Variational Autoencoders (CVAE)

O CVAE tem uma pequena diferença para o VAE padrão, em que nele injetamos a informação da label tanto na fase de codificação quanto na de decodificação. Isso pode ser necessário pois, usando apenas o VAE, nós não temos controle sobre que tipo de dado ele irá gerar. Por exemplo, usando o MNIST e tentando gerar imagens passado $Z \sim \mathcal{N}(0, 1)$ para o decoder, ele pode produzir dígitos aleatórios. As imagens podem ser bem geradas se o autoencoder for bem treinado, mas não podemos ordenar que o VAE gere uma imagem de um dígito específico. Aí entra o CVAE: passando como entrada Y, a label da imagem, queremos como saída X, que será a imagem. Sendo assim, dada a observação y, z é extraído de uma distribuição a priori $P_\theta(z|y)$, e x é gerado da distribuição $P_\theta(x|y, z)$. No VAE simples, a priori é $P_\theta(z)$ e a saída é gerada por $P_\theta(x|z)$.

Assim, temos que o encoder tenta aprender $q_\phi(z|x, y)$, que equivale a aprender representações ocultas dos dados X, ou codificar X na representação oculta condicionada y; enquanto que o decoder tenta aprender $P_\theta(X|z, y)$, que decodifica a representação oculta para o espaço de entrada condicionado por y.

Tal modelo é denotado por:

$$\mathcal{L}(\theta, \phi, x, z, c, \beta) = \mathbb{E}q\phi(z|x, c)[\log p_\theta(x|z, c)] - D_{KL}(q_\phi(x|z, c)||p(z|c))$$

Onde c é a informação condicional, isto é, tanto a distribuição codificada $q(z|x, c)$ quanto a distribuição decodificada $p(x|z, c)$ são condicionadas à informação c.

Nossa implementação de modelo encontra-se em [18].

3.4 β-Variational Autoencoders (β-VAE)

Em nossos experimentos com VAE, reproduzimos os modelos e suas respectivas funções de perdas β-VAE_h, proposto por Higgins et al. [16], e β-VAE_b, proposto por Burgess et al [15]. Para isso, nos baseamos na implementação disponível em [19].

Utilizamos a seguinte arquitetura de um VAE para reproduzir os experimentos:

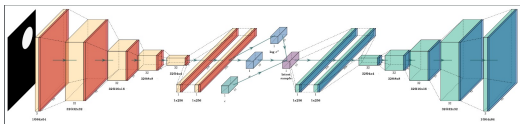

Figura 5: Representação visual da arquitetura do VAE proposto por Burgess. Note que o encoder e o decoder são espelhados.

Esses modelos tratam-se de modificações da estrutura do VAE, onde queremos diminuir a D_{KL} entre as distribuições prior escolhida P_θ e a latente q_ϕ, aproximando os resultados dos dados reais para isso introduzimos um hiperparâmetro β ajustável que equilibra a capacidade latente.

Vamos começar pelo Modelo β-VAE$_h$:

3.5 β-VAE$_h$

O β-VAE_h é um framework proposto em [16] para descoberta automática de representações latentes fatoradas interpretáveis a partir de dados de imagem brutos de uma maneira não supervisionada.

A abordagem é uma modificação da estrutura do autoencoder variacional (VAE), onde se introduz um hiperparâmetro β ajustável que equilibra a capacidade latente do canal e as restrições de independência com precisão de reconstrução:

$$\mathcal{L}(\theta, \phi, x, z, \beta) = \mathbb{E}_{q_\phi(z|x)}[\log p_\theta(x|z)] - \beta D_{KL}(q_\phi(x|z)||p(z))(*)$$

Derivação:

$$\max_{\phi,\theta} \mathbb{E}_{x\sim\mathbb{D}}[\mathbb{E}x \sim q_\phi(z|x) \log p_\theta(x|z)]$$

sujeito a:

$$D_{KL}(q_\phi(z|x)||p_\theta(z)) < \epsilon$$

Com $\epsilon > 0$ e arbitrariamente pequeno.
Para maximizar esse VAE basta usar o multiplicador de Lagrange (como no Apêndice:)

$$\mathcal{L}(\theta, \phi, x, z, \beta) = \mathbb{E}z \sim q_\phi(z|x) \log p_\theta(x|z)] - \beta(D_{KL}(q_\phi(z|x)||p_\theta(z)) - \epsilon)$$

$$\geq \mathbb{E}z \sim q_\phi(z|x) \log p_\theta(x|z)] - \\ - \beta(D_{KL}(q_\phi(z|x)||p_\theta(z)$$

ou seja:

$$\mathcal{L}(\theta, \phi, x, z, \beta) = \mathbb{E}z \sim q_\phi(z|x) \log p_\theta(x|z)] \\ - \beta(D_{KL}(q_\phi(z|x)||p_\theta(z))$$

Nessa equação o termo $\mathbb{E}x \sim q_\phi(z|x) \log p_\theta(x|z)$ recebe o nome de **termo de reconstrução** [3].

O hiperparâmetro β pondera o quanto D_{KL} está presente na função de perda e, consequentemente, na representação desemaranhada. Para isso adotamos as seguinte escalas:

Quando:

- $\beta = 1$, temos a forma original do VAE.
- $\beta > 1$, o termo de D_{KL} terá maior peso. Isso resulta em um espaço latente mais independente ("desemaranhado") à custa da qualidade da reconstrução. [16]
- $\beta < 1$, termo de reconstrução ganha mais peso em relação ao termo de D_{KL}. Isso significa que o modelo irá focar mais em reconstruir os dados de entrada corretamente, e menos na regularização do espaço latente. O resultado é que a qualidade da reconstrução será geralmente melhor, mas o espaço latente pode se tornar menos interpretável e menos contínuo.

Em [16] foi demonstrado que o β-VAE com $\beta > 1$ apropriadamente tunado, qualitativamente — por meio da comparação de imagens com os fatores latentes aprendidas pelos modelos — e quantitativamente — por meio da métrica proposta pelo autor que descreveremos na próxima seção — supera o VAE (β = 1).

Nossa implementação do modelo β-VAE$_h$ encontra-se em [17]. Qualitativamente, comparamos visualmente as imagens de desintanglement produzidas com valores $\beta = 1$ e $\beta > 1$.

3.5.1 Métrica de Disentanglement

Esperamos que uma disentangled representation apresente independência e interpretabilidade, características que a métrica de Disentanglement proposta captura (neste caso, a interpretabilidade é medida devido ao uso de um classificador linear simples dos latentes inferidos).

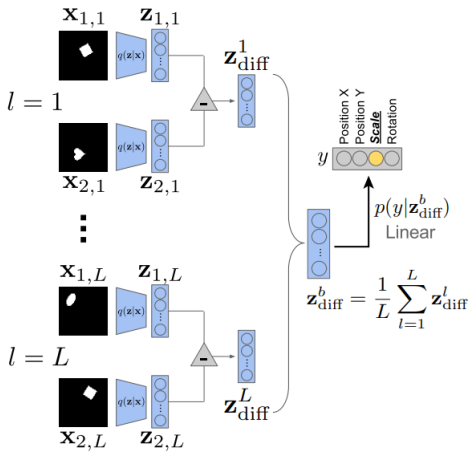

Figura 6: Esquema da métrica de Disentanglement proposta: em um batch de L amostras, cada par de imagens tem um valor fixo para um fator generativo alvo y (aqui y = scale) e difere em todos os outros. Um classificador linear é então treinado para identificar o fator de destino usando a diferença média pareada z^b_{diff} no espaço latente sobre L amostras.

Mais formalmente, partimos de um dataset $D = \{X, V, W\}$, consistindo de imagens $x \in \mathbb{R}^N$ e dois conjuntos de fatores geradores de dados 'ground truth', que são respectivamente fatores condicionalmente independentes $v \in \mathbb{R}^K$, com $\log p(v|x) = \sum_k \log p(v_k|x)$ e fatores condicionalmente dependentes $w \in \mathbb{R}^H$, onde as imagens x são geradas pelo simulador de 'true world' usando os fatores geradores de ground truth correspondentes: $p(x|v, w) = \text{Sim}(v, w)$.

Assim, assumimos que os pontos de dados das imagens são obtidos usando um processo de simulador de ground truth $x \sim \text{Sim}\ (v, w)$, e assumimos também que recebemos rótulos que identificam um subconjunto dos fatores geradores de dados independentes v ∈ V para pelo menos algumas instâncias.

Em seguida, construímos um batch de B vetores z para serem alimentados como entradas para um classificador linear da seguinte forma:

1. Escolha um fator y $\sim$ Unif$[1...K]$ (por exemplo, y = escala na figura 6).
2. Para um lote de L amostras:
   - Amostre dois conjuntos de representações latentes, $v_{1,l}$ e $v_{2,l}$, aplicando $[v_{1,l}]_k = [v_{2,l}]_k$ se $k = y$ (de modo que o valor do fator $k = y$ seja mantido fixo).

[3] Ele garante que, para os termos de entrada x e sua representação latente correspondente z, o modelo deve ser capaz de gerar uma versão reconstruída do dado que seja tão próxima quanto possível.

- Simule a imagem $x_{1,l} \sim$ Sim($v_{1,l}$), e então infira $z_{1,l} = \mu(x_{1,l})$, usando o encoder q(z|x) $\sim$ N (μ(x), σ(x)). Repita o processo para $v_{2,l}$.
- Compute a diferença $z^l_{diff} = |z_{1,l} - z_{2,l}|$, a diferença linear absoluta entre as representações latentes inferidas.

3. Use a média $z^b_{diff} = \frac{1}{L}\sum_{l=1}^{L} z^b_{diff}$ para prever $(y|z^b_{diff})$ e tome a precisão desse preditor como pontuação de métrica de disentanglement.

O objetivo do classificador é prever o índice y do fator gerador que foi mantido fixo para um dado z^b_{diff}. **A precisão desse classificador em vários lotes é usada como nossa pontuação métrica de desemaranhamento.** Escolhemos um classificador linear com baixa dimensão-VC para garantir que ele não tenha capacidade de executar desembaraçamento não linear por si só. Tomamos diferenças de dois vetores latentes inferidos para reduzir o variância nas entradas para o classificador, de modo a garantir que

$[z^b_{diff}]_y < [z^b_{diff}]_{\hat{y}}$ e para reduzir a dependência condicional nas entradas x.

A ideia é que, em média, as diferenças nas representações latentes para o fator gerador fixo (y) são menores do que as diferenças para os outros fatores geradores.

3.6 β-VAE$_b$ — VAE de Burgess com aumento de capacidade controlada:

Proposto por Christopher Burgess et. al [15], é uma extensão do $\beta-$VAE que adiciona um parâmetro de controle C "capacidade de codificação-- onde 0 até um um valor grande o suficiente para produzir reconstruções de boa qualidade – à D_{KL}, o qual o invés minimizar a D_{KL}, como uma VAE tradicional, minimizamos o desvio absoluto da D_{KL}, permitindo um controle mais "fino"sobre a quantidade de informação permitida no espaço latente.

Ao minimizar o desvio absoluto em relação a C, o modelo é incentivado a usar apenas a quantidade certa de capacidade no espaço latente, o que pode ajudar a aprender representações mais desemaranhadas.

Dessa forma, a função de perda assume a seguinte forma:

$$\mathcal{L}(\theta, \phi; x, z, C) = \mathbb{E}_{z\sim q_\phi(z|x)}[\log p_\theta(x|z)] - \gamma|D_{KL}(q_\phi(z|x)||p(z)) - C|$$

O hiperparâmetro γ controla o quanto o modelo é penalizado por desvios de C.
Se γ for grande, o modelo será fortemente penalizado, e se γ for pequeno, o modelo será menos penalizado (basicamente a mesma função do β).
Ao tentar minimizar o desvio absoluto da D_{KL} de um valor fixo C, em vez de minimizá-la diretamente, estamos colocando um "piso"e um "teto"em D_{KL}. Isso "força"o modelo a usar mais dimensões do espaço latente (pois o "piso"impede que a D_{KL} seja zero), ao mesmo tempo que evita que o modelo coloque muita informação em qualquer uma dimensão (pois o "teto"impede que a D_KL seja muito alta).
Observação: assim como β, a escolha tanto C e como o γ depende da modelagem do problema que estamos tentando resolver e dos dados fornecidos. Podemos buscar usando **Grid Search** ou **Cross Validation** [4] ou uma para estimar o melhor valor para C. O modelo proposto por Burgess descreve um procedimento para aumentar C lentamente ao longo do tempo durante o treinamento, começando de um valor pequeno e aumentando até um valor máximo. Isso é conhecido como um "aumento da capacidade"e tem o objetivo de lentamente pressionar o modelo a usar mais do espaço latente à medida que o treinamento progride.

4 Resultados

4.1 Comparação entre β-VAE$_b$ e β-VAE$_h$

Aqui vamos comparar a mesma imagens vericando suas traversais das 10 dimensões latentes utilizando o β-VAE_b e oβ-VAE_h

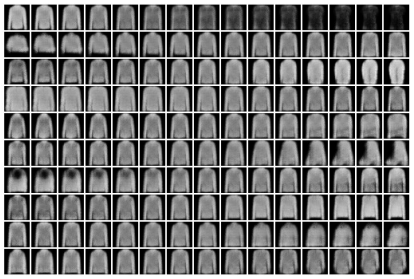

Figura 7: β-VAE_b, com b = 4, C indo de 0 a 25

[4] Ambos são métodos usados no ajuste de modelos de machine learning que consiste em experimentar uma série de valores diferentes, treinar um modelo para cada valor e ver qual funciona melhor de acordo com a métrica de desempenho que foi escolhido.

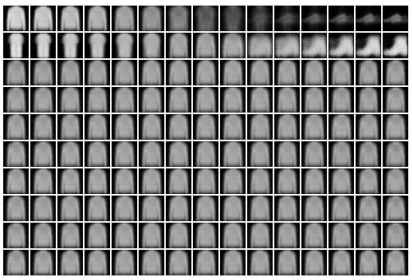

Figura 8: β-VAE_h
Podemos ver aqui que as imagens geradas pelo β-VAE$_b$ são mais nítidas e suáveis.

4.2 2°experimento

Utilizando o VAE proposto por Burgess, fixamos batches de 64 imagens com 32×32 pixels de dimensão, com learning rate de 0.001 e apenas uma epoch, para agilizar a execução do código. A distribuição alvo que utilizamos para as reconstruções das imagens foi a Bernoulli. Paralelamente, fixamos em 10 dimensões latentes (dado que o dataset fashionMNIST possui cerca de 10 sub-variações, escolhemos esse número para tentar distinguir entre as características principais das imagens). Quantificamos a eficácia do método realizando para o β-VAE bem como prever invisíveis instâncias de Y para o conjunto de dados "Fashion MNIST" . Para fornecer comparações justas, usamos arquiteturas de rede neural de codificador e decodificador semelhantes ao realizar comparações entre diferentes valores de β:

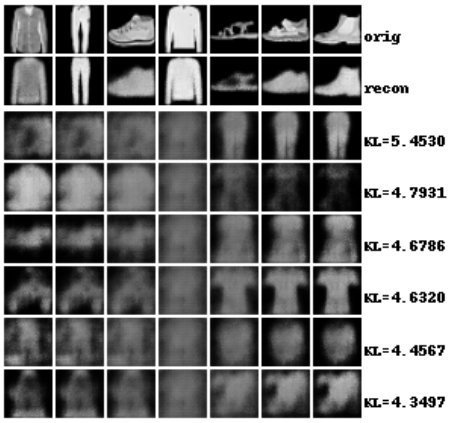

Figura 9: resultado para β=0.5

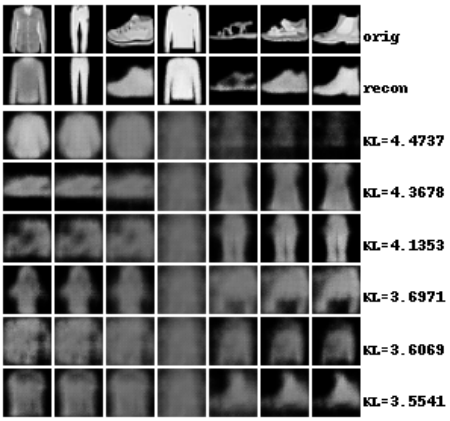

Figura 10: resultado para β=1

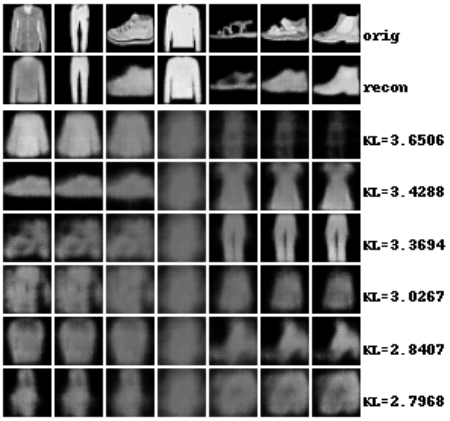

Figura 11: resultado para β=3

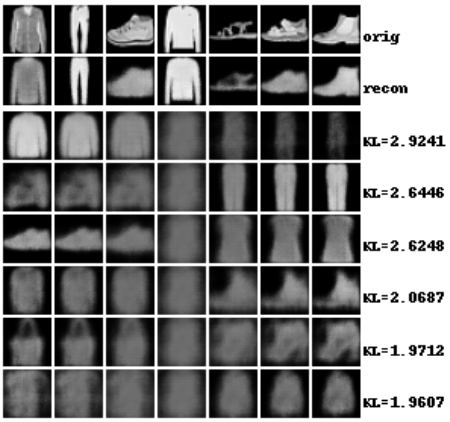

Figura 12: resultado para β=10

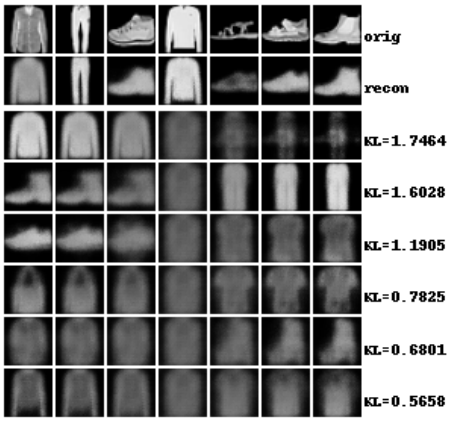

Figura 13: resultado para β=100

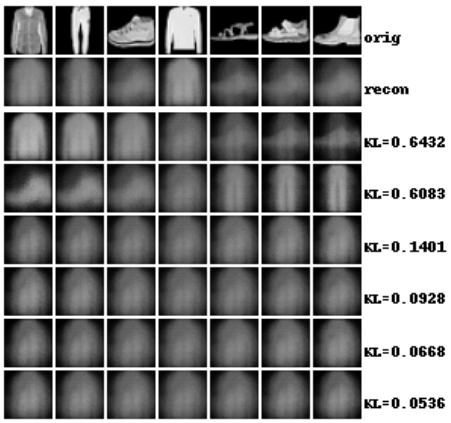

Figura 14: resultado para β=1000

5 Conclusão

Podemos observar, através de testes empíricos, que os valores de $\beta > 1$ obtiveram resultados melhores (forçando a minimização da D_{KL}, ou seja, aproximando a distribuição desejada), até certo ponto. Visualmente, percebe-se que $\beta = 3$ é um parâmetro que captura bem as dimensões das imagens para os exemplos de teste. Ao mesmo tempo, os valores maiores de β acabam por "corromper" o modelo, generalizando ao extremo as imagens obtidas. Assim, os resultados visuais são formas de borrões, falhando em representar quaisquer um dos objetos.

Outro fato importante é que a métrica descrita em 6 funciona apenas para dados já rotulados. Para dados puramente não supervisionados, é necessário procurar uma heurística de inspeção visual ao invés de aplicar a métrica. Por isso não foi possível sua implementação no nosso experimento.

Referências

[1] Oring et al. "Autoencoder Image Interpolation by Shaping the Latent Space."2020.

[2] Berthelot et al. "Understanding and Improving Interpolation in Autoencoders via an Adversarial Regularizer."2018.

[3] Liu, et al. "Learning disentangled representations in the imaging domain."NeurIPS 2022.

[4] Feige, Ilya. "Invariant-equivariant representation learning for multi-class data."2019.

[5] Kainz, Bernhard. "Deep Learning – Equivariance and Invariance."

[6] Taboga, Marco. "Domain shift."2023.

[7] Wang et al. "A Review of Disentangled Representation Learning for Remote Sensing Data."2022.

[8] Higgins et al. "Towards a Definition of Disentangled Representations."2018.

[9] Weng, Lilian. "From Autoencoder to Beta-VAE."2018.

[10] Esmaeili et al. "Structured Disentangled Representations."2019.

[11] Huang et al. "Sufficient and Disentangled Representation Learning."2023.

[12] Higgins et al. "DARLA: Improving Zero-Shot Transfer in Reinforcement Learning."2017.

[13] Kingma, Diederik P., Max Welling. "Auto-Encoding Variational Bayes."2013.

[14] Rahman, Md Ashiqur. "Understanding Conditional Variational Autoencoders."2018.

[15] Burgess et al. "Understanding disentangling in β-VAE."2018.

[16] Higgins, Irina, et al. "beta-vae: Learning basic visual concepts with a constrained variational framework."2017.

[17] Implementação do β-VAE. `https://github.com/CarlSouza/Machine-LEarn/blob/main/join_notebook/Beta_diseentangling_vae.ipynb`

[18] Implementação do CVAE `https://github.com/CarlSouza/Machine-LEarn/blob/main/cvae.ipynb`

[19] Disentangling VAE. Yann Dubois, Alexandros Kastanos, Dave Lines, Bart Melman. March 2019. `http://github.com/YannDubs/disentangling-vae/`

6 Apêndice

.1 Operações

- a equivariância significa que a saída passa pela mesma transformação que é aplicada aos dados de entrada, enquanto que a invariância implica que a saída é a mesma, independentemente das transformações aplicadas aos dados de entrada.
- Os fatores geradores $\mathcal{G}$ são variáveis subjacentes que caracterizam a variação dos dados em $\mathcal{X}$. As representações devem permitir a decomposição dos dados em fatores distintos. Também é necessário considerar as alterações de domínio entre conjuntos de treinamento, validação e teste.

.2 Condições

- A Modularidade garante que cada feature desemaranhada corresponda a um fator gerador e que as features desemaranhadas sejam independentes umas das outras. A Compacidade requer que o modelo mapeie os dados originais de alta dimensão para features de baixa dimensão, funcionando como um compressor de dados.
- A Explicitude é composta por duas partes: completude e informatividade. A completude envolve a representação desemaranhada expressar todas as variáveis dos dados observados, enquanto a informatividade diz respeito às features desemaranhadas serem capazes de recuperar os valores dos fatores generativos por meio do modelo.

.3 ELBO: Evidence Lower BOund

:

$$max_{\phi,\theta}\mathbb{E}x \sim q_\phi(z|x)logp_\theta(x|z)$$

Primeiro, introduzimos uma distribuição $q(z)$ sobre as variáveis latentes z e reescrevemos a log-verossimilhança marginalizada de maneira equivalente usando a D_{KL}

$$logp_\theta(x|z) = D_{KL}[q(z|x)||p(z)] + ELBO$$

onde ELBO será dado pela função de perda (loss function: $\mathcal{L}(\theta, \phi, x, z)$)
Portanto, maximizar $logp(x)$ é equivalente a minimizar $D_{KL}q(z|x)||p(z)$ ou, equivalentemente, maximizar Loss.

Porque $D_{KL}q(z|x)||p(z)$ é não negativa, 0 ocorre quando $q(z) = p(x|z)$ maximizar ELBO é uma aproximação razoável de maximizar $logp(x)$ quando q(z) é restrito a uma família específica de distribuiçõew. Agora, assumindo uma forma específica para $q(z)$ (por exemplo, q(z) = N(μ, σ), nós expressamos como:

$$\log p_\theta(x|z) \quad \geq \quad \mathcal{L}(\theta, \phi, x, z) \quad = \quad \mathbb{E}z \sim q_\phi(z|x)logp_\theta(x|z)] \; - \; (D_{KL}(q_\phi(z|x)||p_\theta(z)$$

.4 Métodos de busca

Grid Search

A busca em grade, ou Grid Search, é um método de hiperajuste de parâmetros. O objetivo é encontrar a combinação ideal de parâmetros que otimiza o desempenho do modelo. Para realizar uma busca em grade, você define uma grade de hiperparâmetros e, em seguida, treina/testa seu modelo para cada combinação de parâmetros na grade para ver qual combinação fornece o melhor desempenho. Por exemplo, se você tem dois hiperparâmetros, digamos taxa de aprendizado e número de camadas em uma rede neural, e você quer testar 3 valores diferentes para cada um, então sua grade terá um total de 3*3 = 9 combinações de hiperparâmetros para testar.

Cross Validation

A validação cruzada, ou Cross Validation, é um procedimento usado para evitar o overfitting ao estimar a habilidade do seu modelo em dados não vistos, ou seja, a sua capacidade de generalização. O método mais comum é o k-fold cross-validation. No k-fold cross-validation, o conjunto de dados é dividido em k subconjuntos distintos. O modelo é treinado em k-1 desses subconjuntos e testado no subconjunto restante. Esse processo é repetido k vezes, com cada subconjunto usado exatamente uma vez como dados de teste. A estimativa final da performance do modelo é a média dos valores computados nos k loops. Isso permite usar todos os dados para treinamento e teste, e fornece uma medida da robustez do modelo.

.5 Resultados

Reconstrução de imagens Plotamos também as reconstruções dos objetos, para os mesmos valores de β:

Figura 15: resultado para β=0.5

Figura 16: resultado para β=1

Figura 17: resultado para β=3

Figura 18: resultado para β=10

Imagens geradas

Abaixo temos as imagens originais, das quais geramos amostras

Figura 19: resultado para β=100

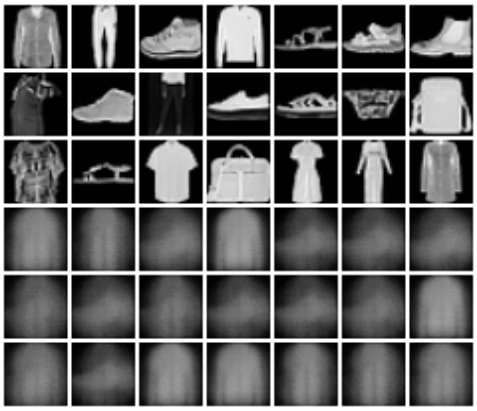

Figura 20: resultado para β=1000

Figura 21: Dados de amostra originais

Agora, para cada valor de β utilizado nos exemplos práticos, plotamos também a amostragem dos dados:

Figura 22: Amostragem para $\beta = 0.5$

Figura 23: Amostragem para $\beta = 1$

Figura 24: Amostragem para $\beta = 3$

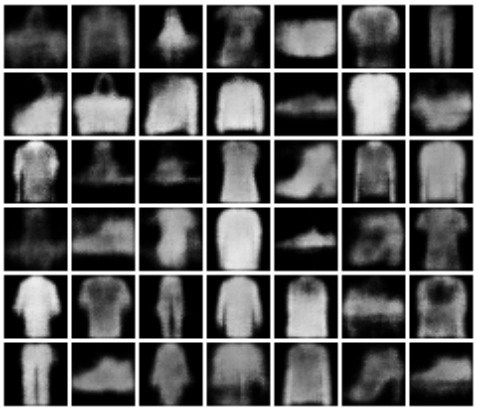

Figura 25: Amostragem para $\beta = 10$

Figura 26: Amostragem para $\beta = 100$

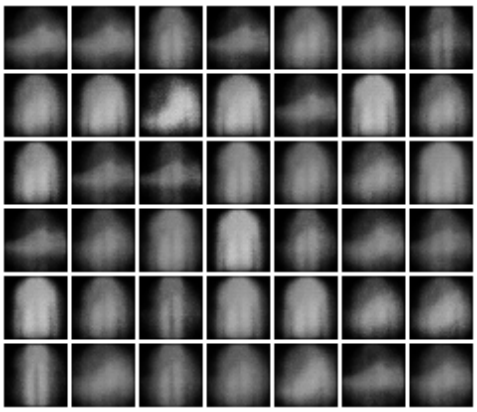

Figura 27: Amostragem para $\beta = 1000$

Imagens Geradas pelo VAE: