Deep Learning (Parte 3)

Índice Geral

1 As tarefas mais comuns em aprendizado de máquina 2
2 Alguns exemplos de mapeamentos multidimensionais 7
3 Aprendizado da representação 13
4 Hipótese das variedades (manifolds) 15
5 Fundamentos de autoencoders 21
5.1 Denoising autoencoders 26
5.2 Pré-treinamento empregando stacked autoencoders 29
5.3 Variational autoencoders 32
5.4 O truque da reparametrização 38
5.5 Disentagled VAE 42
6 Restricted Boltzmann machines (RBMs) 43
6.1 RBMs × (V)AEs 51
7 Alucinação em redes neurais profundas 52
7.1 Neural inpainting 55
8 Referências bibliográficas 60

1 As tarefas mais comuns em aprendizado de máquina

- O conteúdo desta seção não deve ser visto como uma taxonomia fechada, mas apenas como uma amostragem de tarefas que vêm sendo resolvidas com sucesso empregando técnicas de aprendizado de máquina.
- Segundo GOODFELLOW et al. (2016), as tarefas mais comuns em aprendizado de máquina são:
    - ✓ **Classificação**: a qual dentre *k* classes a entrada pertence? Variantes envolvem fornecer a distribuição de probabilidade ao longo das classes.
    - ✓ **Classificação com entradas faltantes**: não havendo a garantia de que todas as variáveis de entradas sejam fornecidas a todo momento, pode-se modelar a distribuição de probabilidade envolvendo todas as variáveis relevantes, de modo a definir a distribuição de probabilidade ao longo das classes. Com isso, é possível empregar a lei da probabilidade total e marginalizar as variáveis ausentes. De fato,

considerando n variáveis de entrada, esta técnica permite obter 2^n diferentes funções de classificação, todas deriváveis de uma única distribuição de probabilidade conjunta. Muitas das demais tarefas de aprendizado de máquina a serem apresentadas a seguir admitem este mesmo tratamento.

Considere duas variáveis aleatórias A e B. Caso se conheça a probabilidade conjunta $P(A=a, B=b)$ para todos os valores possíveis de a e b, pode-se marginalizar a variável A como segue:

$$P(B=b) = \sum_a P(A=a, B=b)$$

- ✓ **Regressão**: a máquina de aprendizado deve predizer um ou mais valores numéricos na saída, dada uma entrada multidimensional. É similar ao problema de classificação, mas difere no tipo de saída.
- ✓ **Transcrição**: a máquina de aprendizado deve observar uma informação não-estruturada e a descrever numa forma estruturada. Por exemplo, a partir da imagem de um texto, deve-se retornar o texto na forma de uma sequência de caracteres. Como um outro exemplo, tem-se o reconhecimento de fala, em que uma máquina de aprendizado recebe na entrada uma onda sonora, a qual deve ser transcrita na saída em formato de texto.
- ✓ **Tradução de máquina**: a entrada já aparece na forma de uma sequência de símbolos em alguma linguagem, sendo que a máquina deve converter esta entrada numa sequência de símbolos de uma outra linguagem, capaz de expressar um conteúdo semântico equivalente.

- ✓ **Saída estruturada**: trata-se de uma ampla classe de aplicações em potencial, onde a saída expressa relações importantes entre seus múltiplos elementos. Transcrição e tradução, descritas acima, se encaixam aqui também. O exemplo mais impressionante é o recebimento de uma imagem e sua rotulação pela geração automática de um texto em linguagem natural que descreve o seu conteúdo. Como outro exemplo não menos impressionante tem-se a segmentação pixel a pixel de imagens, com cada pixel sendo classificado numa categoria específica. O fundamental aqui é que as múltiplas saídas estabeleçam um forte interrelação.
- ✓ **Detecção de anomalia**: varre-se um amplo conjunto de eventos ou objetos e sinaliza-se quando alguns deles são atípicos. Como um exemplo, tem-se detecção de fraude em operações de cartão de crédito, que envolve a modelagem dos hábitos de compra do cliente.

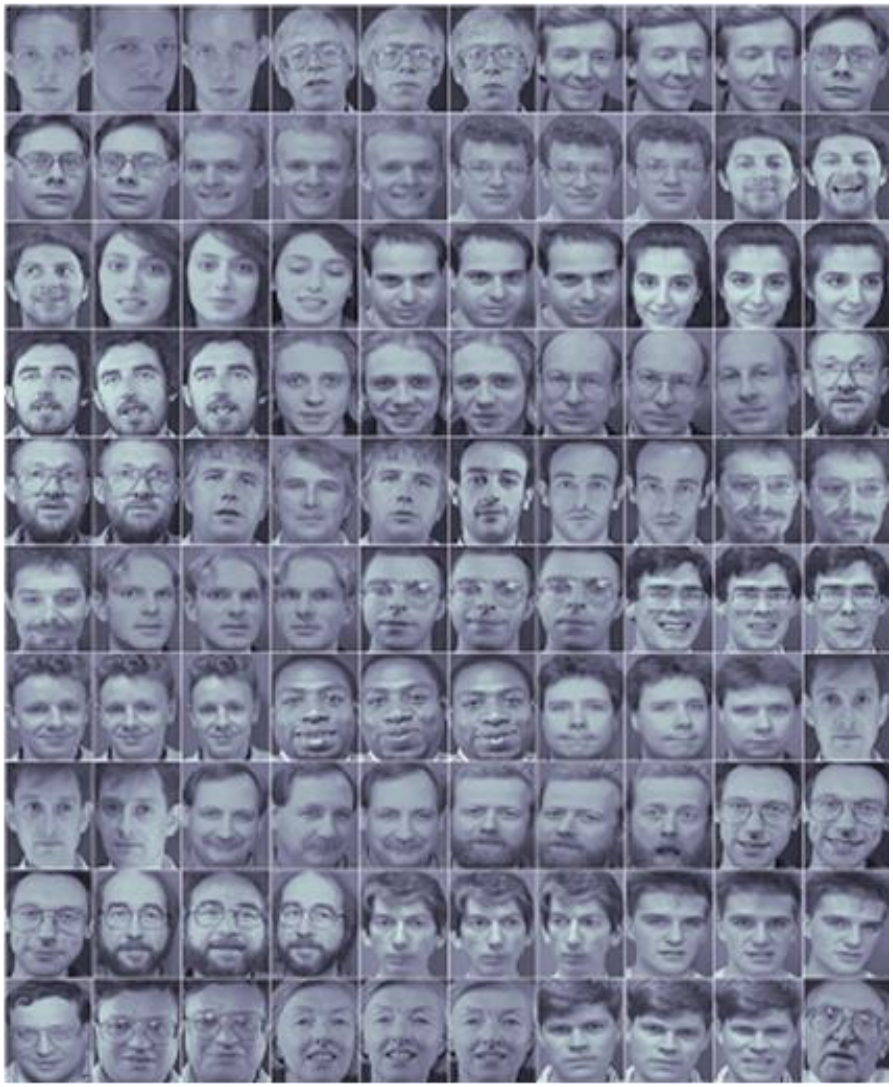
- ✓ **Síntese e amostragem**: a máquina de aprendizado deve gerar novos exemplos similares àqueles do conjunto de treinamento. Uma possibilidade de aplicação está em videogames e outras atividades de mídia em que deve-se gerar uma grande quantidade de conteúdo, por exemplo, em fundo de tela.
- ✓ **Imputação de dados faltantes**: a máquina de aprendizado deve ser capaz de preencher adequadamente os elementos faltantes da entrada.
- ✓ **Eliminação / atenuação de ruído**: a entrada fornecida é uma versão corrompida de um padrão puro, sendo que não se conhece o processo de corrupção do conteúdo original, e o objetivo é reproduzir este padrão puro na saída.
- ✓ **Estimação de densidade ou estimação da função massa de probabilidade**: a máquina de aprendizado deve aprender a estrutura dos dados vistos na fase de treinamento, tomando as variáveis como sendo

contínuas ou discretas. Um exemplo de aplicação está na imputação de dados faltantes. A função distribuição de probabilidade não precisa ser capturada explicitamente, como ocorre em estimação de densidade, pois pode-se recorrer ao conceito de aprendizado da variedade (*manifold*). Além disso, dependendo das dimensões, estimar e operar com estimativas de probabilidade se torna intratável.

2 Alguns exemplos de mapeamentos multidimensionais

- São muitos os problemas da literatura que podem ser resolvidos a partir da síntese de mapeamentos multidimensionais. Seguem alguns exemplos, ainda sem evidenciar a etapa de aprendizado da representação, que pode ser interpretado como a síntese de um mapeamento do espaço original dos dados para um espaço de representação que torna a tarefa de aprendizado mais fácil de ser resolvida.

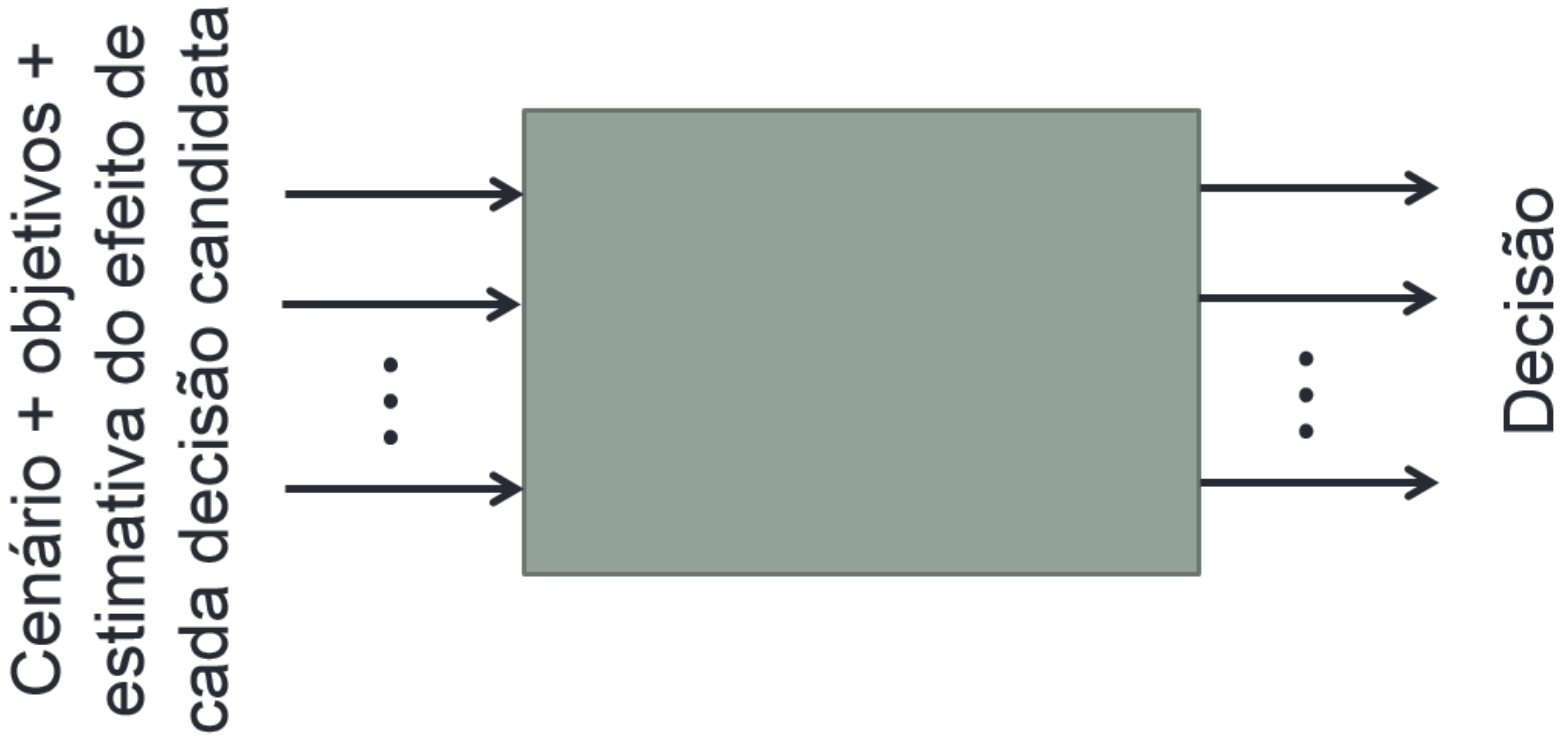
Reconhecimento de padrões / Classificação:




Atributos
Classes

Planejamento:

Tomada de decisão:



Predição:

3 Aprendizado da representação

- O sucesso dos algoritmos de aprendizado de máquina depende fundamentalmente da forma com que os dados são representados (BENGIO et al., 2013).
- Representações diferentes carregam ou escondem diferentes fatores capazes de explicar a distribuição e o comportamento dos dados.
- Embora conhecimento de domínio possa, vez ou outra, estar disponível, é possível também aprender a partir de hipóteses iniciais genéricas (*priors*). Mas o que tem se mostrado mais efetivo é o aprendizado da representação, que vai sintetizar esses *priors* a partir dos dados.
- Classificadores de alto desempenho, por exemplo, requerem a identificação de atributos discriminativos das classes.
- Esta parte do material vai abordar aprendizado de variedade (*manifold*) e *autoencoders*.

- Se a inteligência artificial envolve conceber um bom modelo de mundo, de modo a maximizar o acerto na tomada de decisão, é possível argumentar que isso só pode ser alcançado ao se aprender como identificar e desembaraçar os fatores explicativos fundamentais que se escondem no baixo nível dos dados observados (BENGIO et al., 2013).
- Com base em um aprendizado de representação competente, tende a ficar mais fácil a extração de informação útil ao se sintetizarem classificadores e outros preditores.
- Métodos de *deep learning* são caracterizados pela composição de uma cascata de transformações não-lineares com o propósito de produzir representações dos dados que sejam mais abstratas e mais úteis para a execução da tarefa pretendida.
- Objetivos centrais em aprendizado de representação são: (1) Identificar o que faz uma representação ser melhor que outra; (2) Definir como sintetizar tal representação; (3) Expressar matematicamente os objetivos a serem atingidos.

4 Hipótese das variedades (manifolds)

- Hipótese das variedades (manifolds): a função densidade de probabilidade que gera os dados se concentra em torno de regiões de baixa dimensão (CAYTON, 2005; NARAYANAN & MITTER, 2010).
- Sendo assim, é muito pouco provável que configurações de variáveis do processo definidas segundo uma distribuição uniforme, ou outras distribuições arbitrárias e definidas a priori, sejam capazes de gerar o tipo de dado que se quer modelar.
- Qual é a probabilidade de se gerar uma imagem que aparente ser real pela escolha independente da intensidade dos pixels?
- Qual é a probabilidade de se gerar um texto sintática e semanticamente válido, numa linguagem, pela escolha independente dos caracteres na sequência?
- De fato, dentre todas as distribuições possíveis de pixels em uma imagem, por exemplo, por mais variação que esteja presente na base de imagens, ainda assim a função densidade de probabilidade que explica as distribuições de pixels em imagens de interesse prático tende a residir em espaços de baixa dimensão e volume reduzido.

- Algo equivalente pode ser dito a respeito de dados de texto, vídeo, fala e música.
- O conceito de variedade é mais rigoroso em teorias matemáticas do que em fundamentos de aprendizado de máquina:
    - ✓ Os dados não precisam pertencem estritamente à variedade, podendo apenas estar próximos dela.
    - ✓ A dimensão da variedade não precisa ser a mesma em todo lugar.
    - ✓ A variedade pode residir em espaços contínuos ou discretos.
- O fundamental é mapear onde se concentram as amostras de mais alta probabilidade, e é isso que o aprendizado de variedades busca realizar, seja para distribuições contínuas ou discretas. Logo, o problema é mais de aproximação de uma variedade em um subespaço do que de estimação de densidade de probabilidade, embora ambas as perspectivas sejam válidas em tarefas de aprendizado de máquina.
- Mais especificamente, busca-se desdobrar a variedade a partir de mapeamentos não-lineares, para, em seguida, aproximar esta variedade por um hiperplano de dimensão compatível, ou seja, por uma solução de PCA.

- No exemplo da Figura 1, percebe-se que a distribuição de probabilidade se concentra em um volume desprezível de todo o espaço de configurações admissíveis, indicando que identificar a região em que a variedade se manifesta representa uma informação de alta relevância.

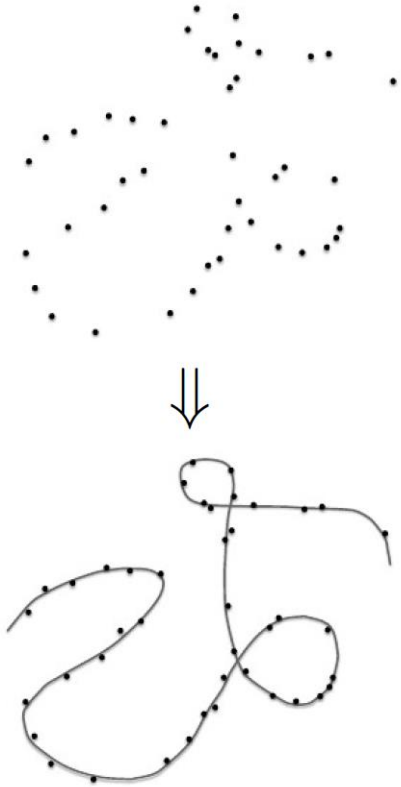


Figura 1 – Dados amostrados de uma distribuição 2D que, de fato, está concentrada em uma variedade unidimensional. Fonte: GOODFELLOW et al. (2016).

- Identificar adequadamente a variedade permite então responder a perguntas como:
    - ✓ A amostra corresponde a uma configuração provável ou não?
    - ✓ A sentença é sintática e semanticamente plausível ou não?
    - ✓ A imagem tem uma aparência real ou não?
- Responder a essas questões, que admitem respostas binárias, por indicar onde as altas probabilidades se concentram, nos diz muito mais acerca das características, por exemplo, de uma linguagem natural ou imagem, do que a informação adicional associada à atribuição de uma probabilidade precisa a cada sequência possível de caracteres ou subconjunto de pixels.
- É sabido que as variedades de interesse em IA geralmente apresentam muitas oscilações e dobras, o que torna atrativo o emprego de arquiteturas profundas de redes neurais para identificar, desbobrar e suavizar cada variedade.
- Análise de componentes principais (PCA) pode ser interpretada como uma técnica capaz de identificar variedades lineares junto aos dados, conforme ilustrado nas figuras a seguir, extraídas de HASTIE et al. (2009).

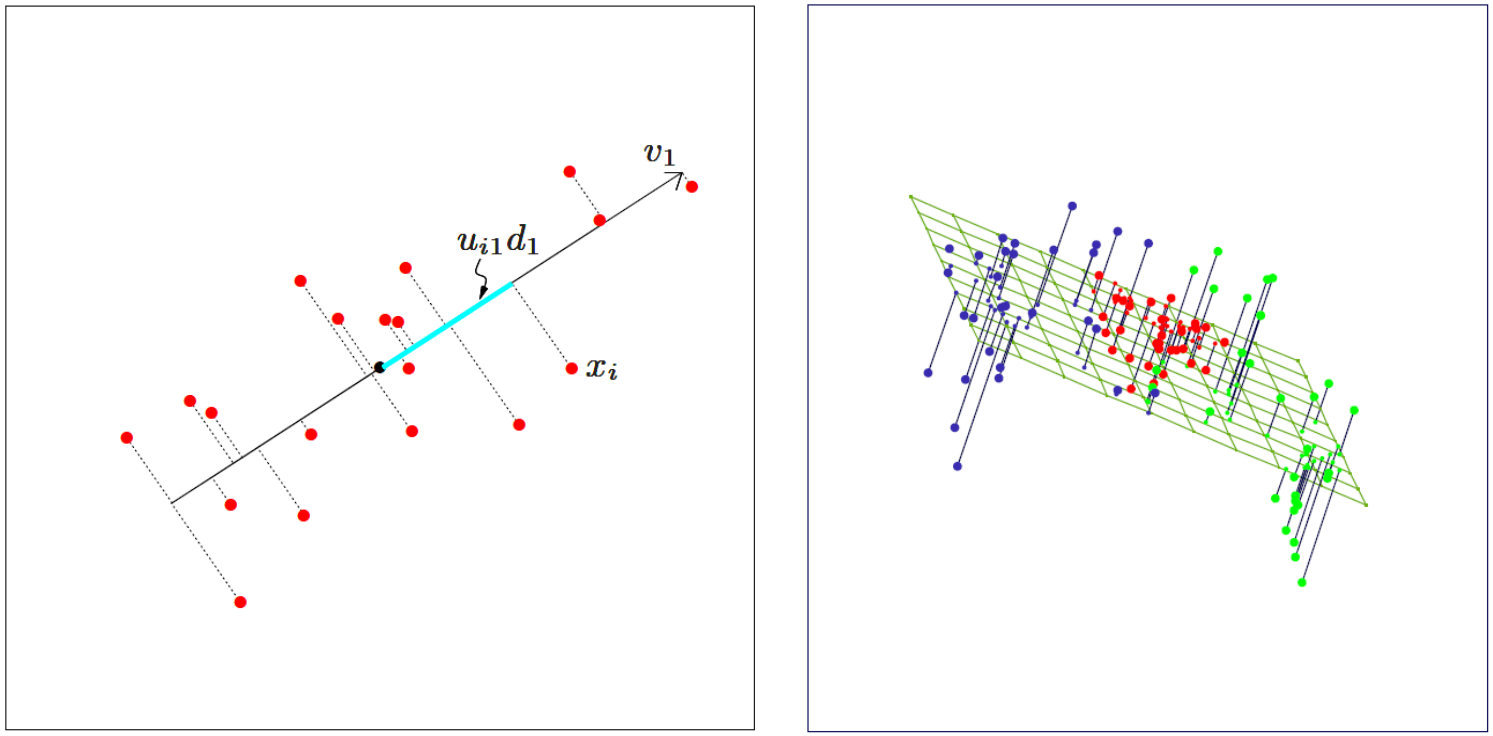


Figura 2 – Variedades lineares obtidas a partir de análise de componentes principais (PCA).

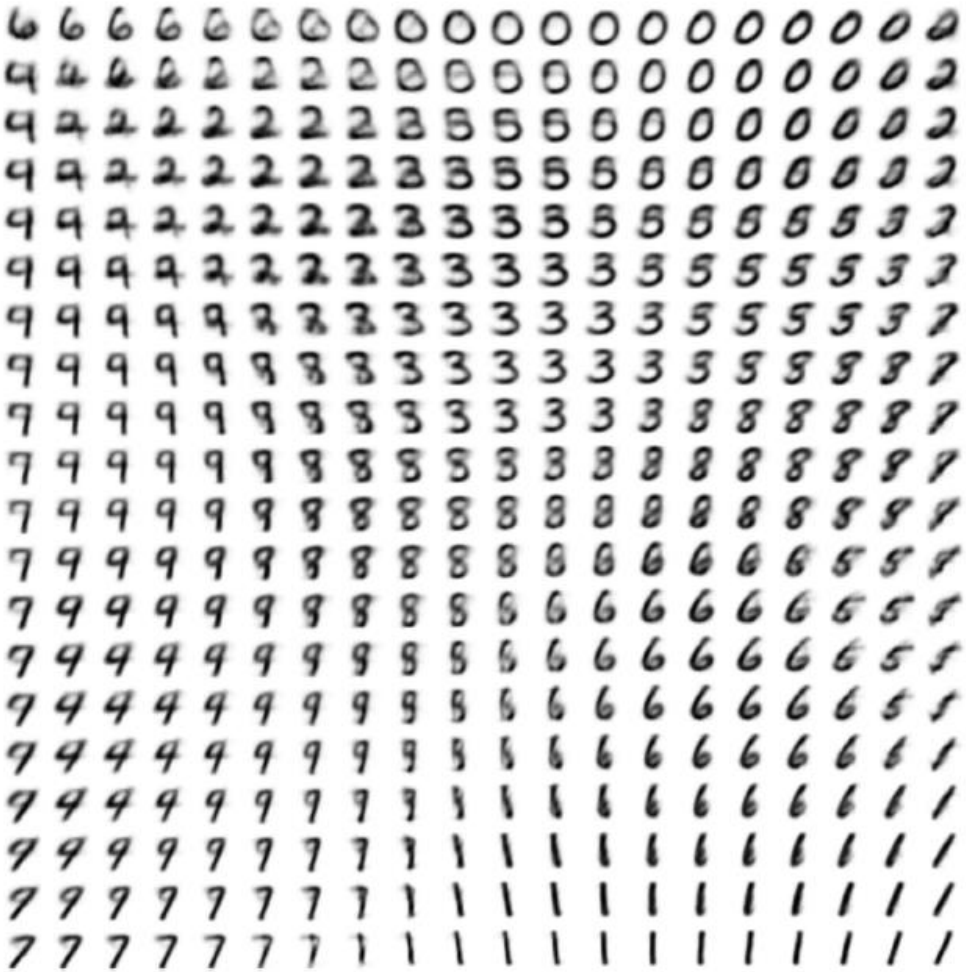


Figura 3 – Manifold aprendido por um autoencoder variacional para a base de dados MNIST.

Fonte: GOODFELLOW et al. (2016).

5 Fundamentos de autoencoders

- Uma das configurações mais poderosas para *deep learning*, usado, por exemplo, em aprendizado da representação e na identificação de variedades (*manifolds*), é o *autoencoder*, que está associado a treinamento não-supervisionado, mas é guiado por treinamento supervisionado.
- O objetivo é codificar eficientemente os dados de entrada em um espaço de representação, ou espaço de variáveis latentes, geralmente com redução de dimensão (gerando assim um gargalo, *bottleneck*), e empregar este código para reconstruir a entrada na saída da rede neural.
- Muitas arquiteturas profundas de alto desempenho foram concebidas como uma cascata de camadas de codificadores sintetizados por autoencoders (o código do autoencoder anterior é tomado como a entrada para o treinamento do autoencoder seguinte), processo denominado de *stacked autoencoders*. Autoencoders têm sido

empregados numa grande diversidade de aplicações bem-sucedidas, como em modelos geracionais, eliminação de ruído em imagens e *inpainting* em imagens.

- Em termos de topologia, um autoenconder tradicional pode ser interpretado como duas redes neurais MLP em cascata, uma responsável pelo codificador e a outra pelo decodificador. O emprego de função de ativação linear conduz o codificador do autoencoder a realizar PCA.

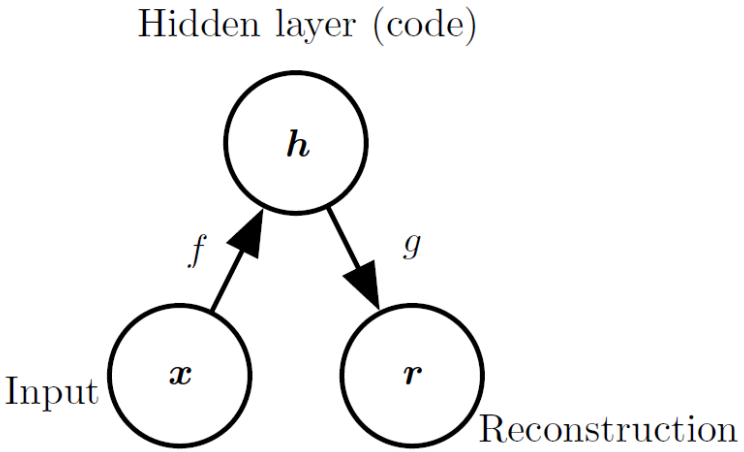


Figura 4 – Sequência de mapeamentos do autoencoder. Fonte: GOODFELLOW et al. (2016).

- Esta sequência de mapeamentos deve ser tal que *r* aproxime *x*.

- Deve-se evitar que $f(\cdot)$ – codificador – e $g(\cdot)$ – decodificador – realizem a função identidade. É até possível que h tenha uma dimensão maior que x, mas, para isso, técnicas de regularização são mandatórias.

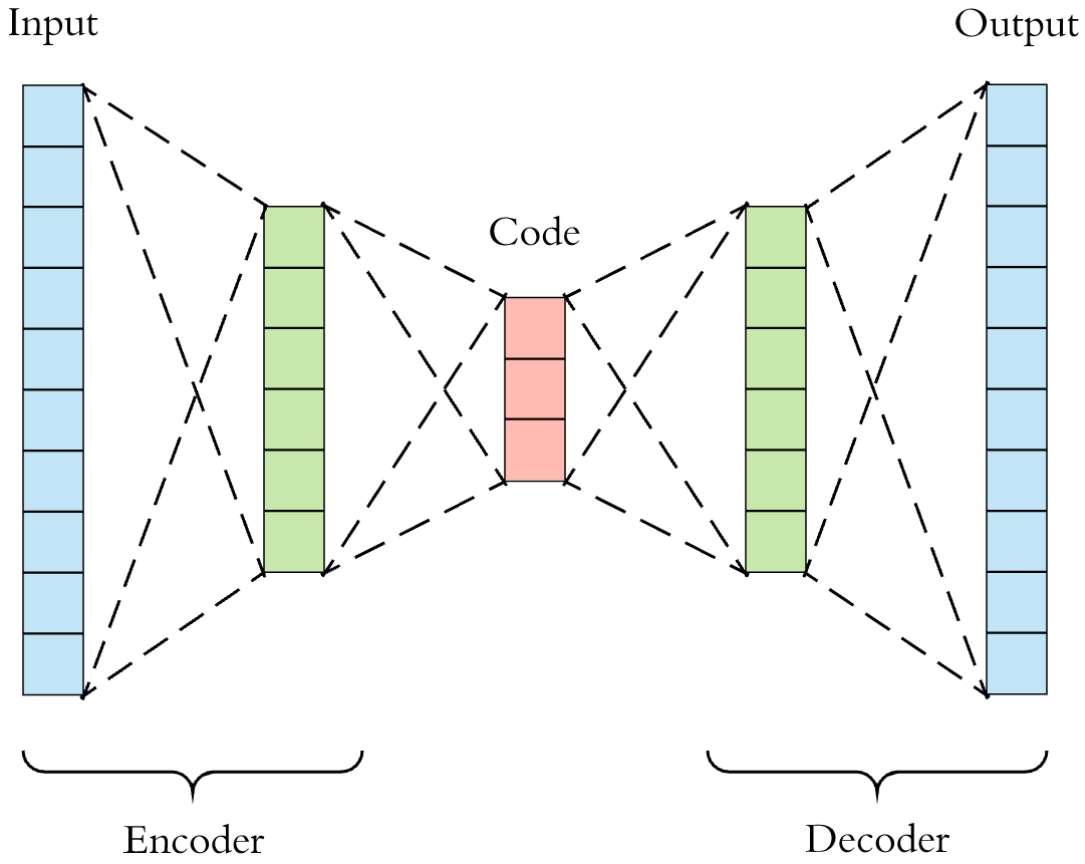


Figura 5 – Configuração tradicional de um autoencoder

- Nada impede o emprego de mais de uma camada intermediária para cada MLP, se for conveniente, ou até a adoção de camadas convolucionais.

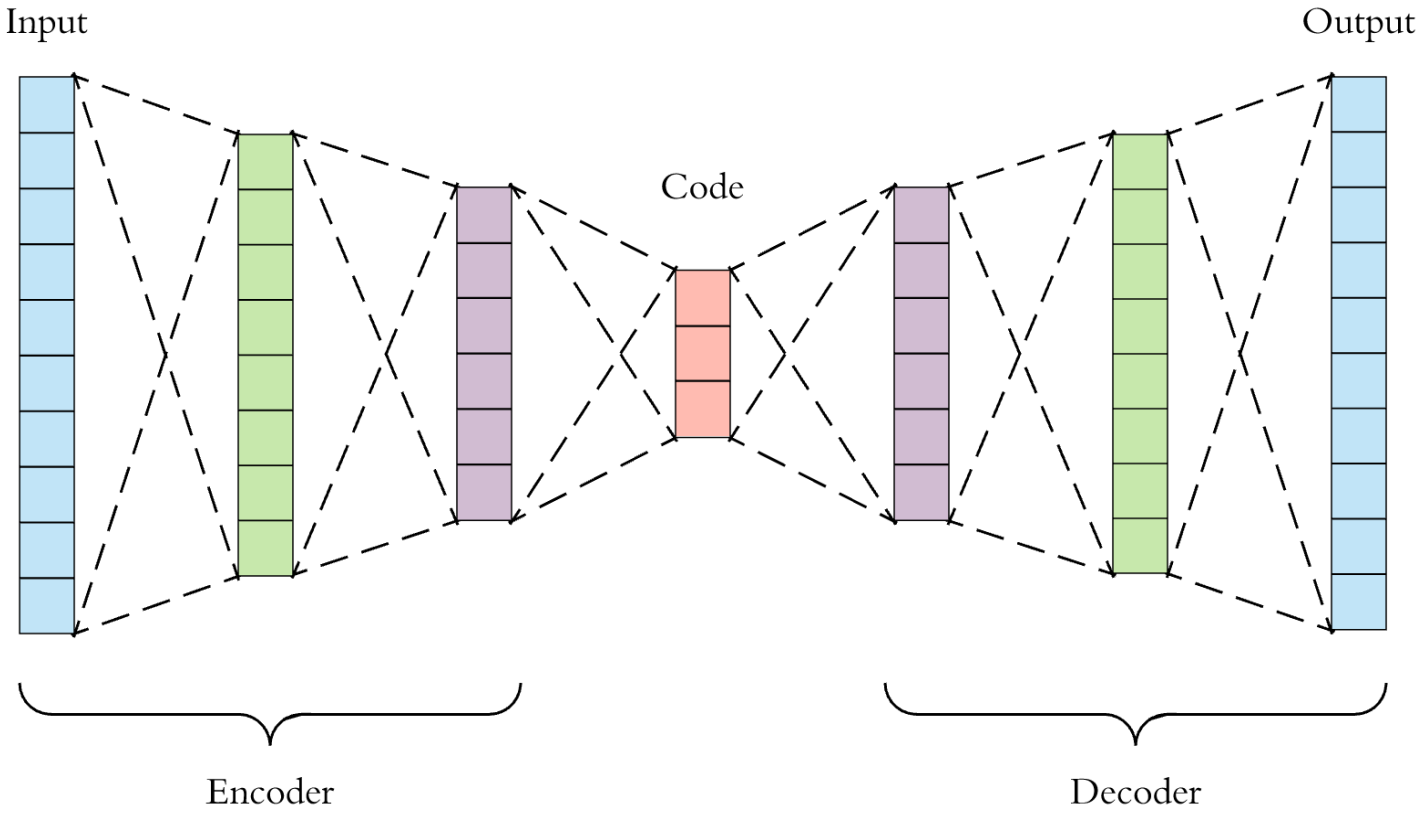


Figura 6 – Autoencoder tradicional, com mais camadas intermediárias

- É fundamental compreender que codificador e decodificador são treinados simultaneamente para reduzir o erro de reconstrução na saída da rede neural.
- Autoencoders convolucionais, como o da Figura 7, podem operar com imagens de resolução maior, pois promovem economia de conexões sinápticas.

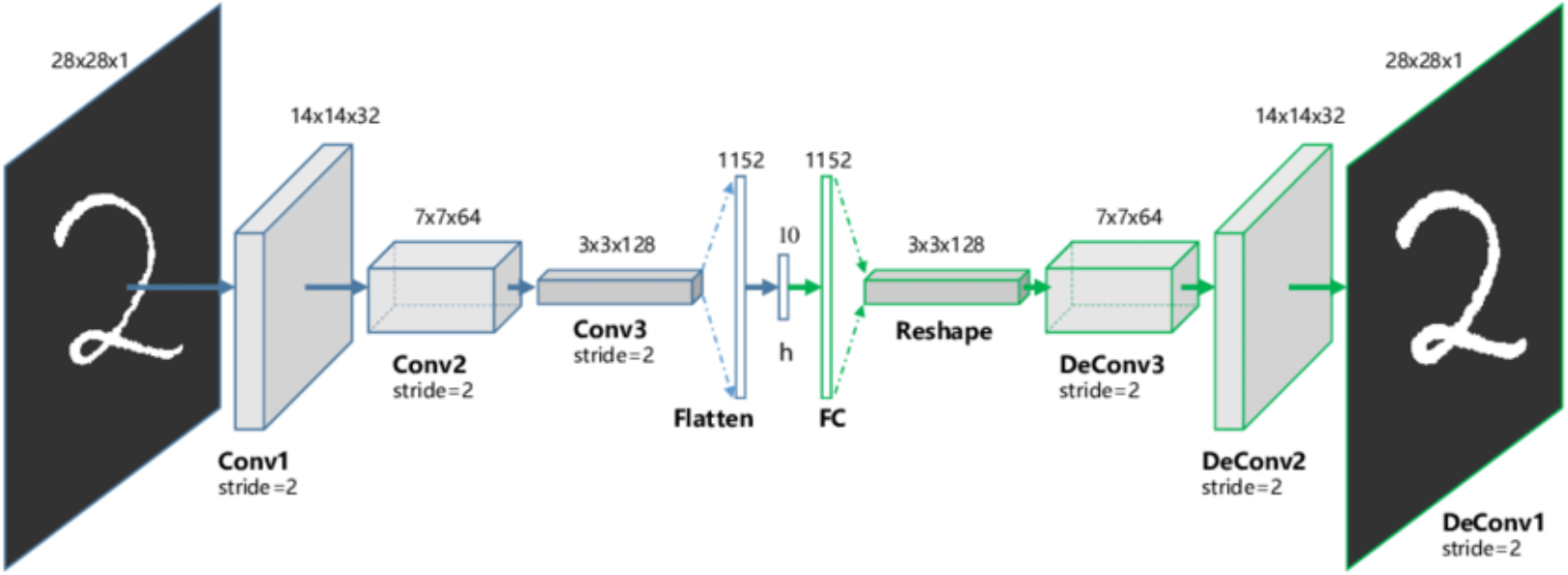


Figura 7 – Exemplo de um autoenconder convolucional. Fonte: GUO, X.; LIU, X.; ZHU, E.; YIN, J. "Deep Clustering with Convolutional Autoencoders", ICONIP, pp. 373-382, 2017.

5.1 Denoising autoencoders

- Objetivo: gerar uma representação latente mais robusta, que captura a essência da variação nos dados de entrada.

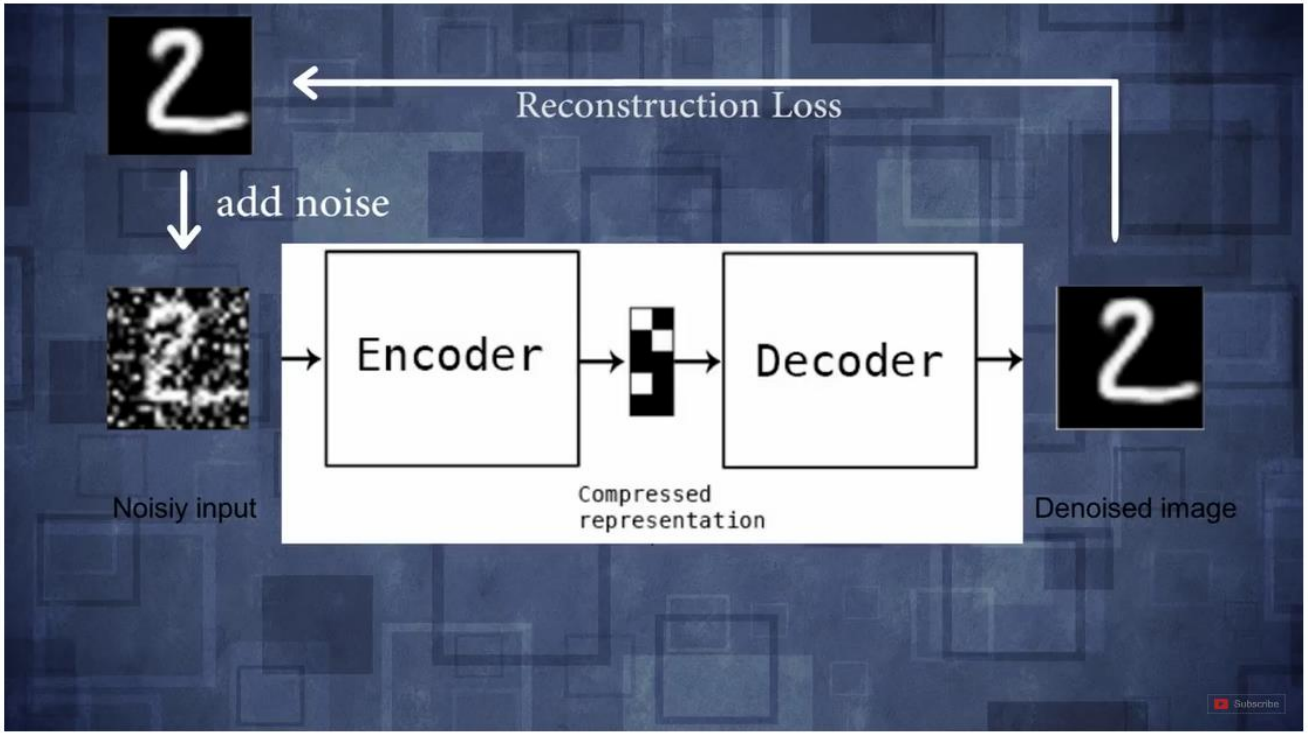


Figura 8 – Princípio de operação do denoising autoencoder.

Fonte: https://www.youtube.com/watch?v=9zKuYvjFFS8

Denoising Autoencoder

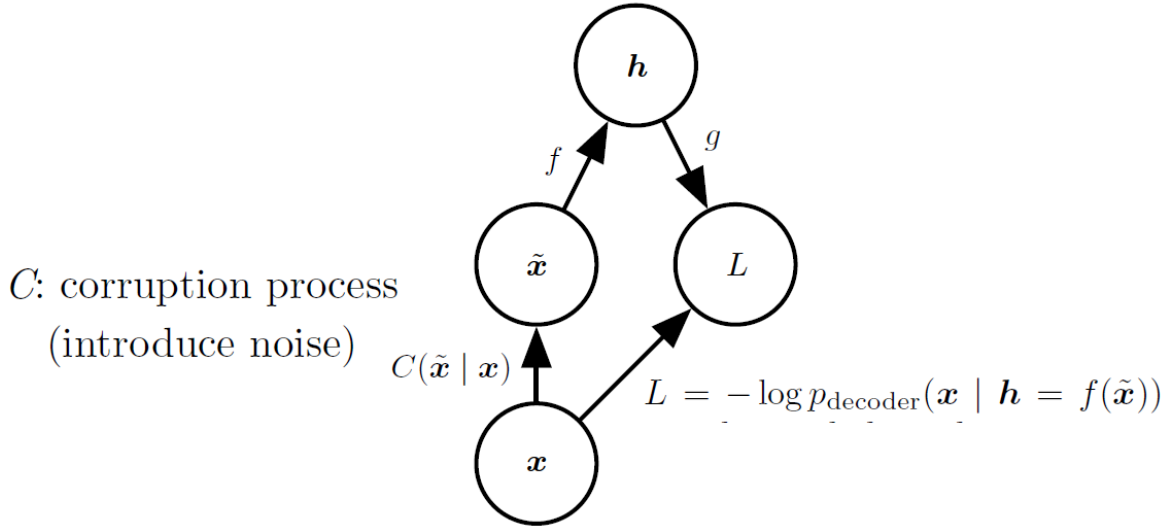


Figura 9 – Princípio de operação do denoising autoencoder com ênfase nos mapeamentos e não função de perda. Fonte: GOODFELLOW et al. (2016).

- São variados os tipos de ruído que podem ser introduzidos, dependendo da aplicação pretendida. Exemplos: gaussiano, sal e pimenta, oclusão, dropout, etc.
- A norma L_1 tende a produzir melhores resultados que a norma L_2, pois esta última acaba gerando imagens um pouco embaçadas, como aquelas da Figura 10.

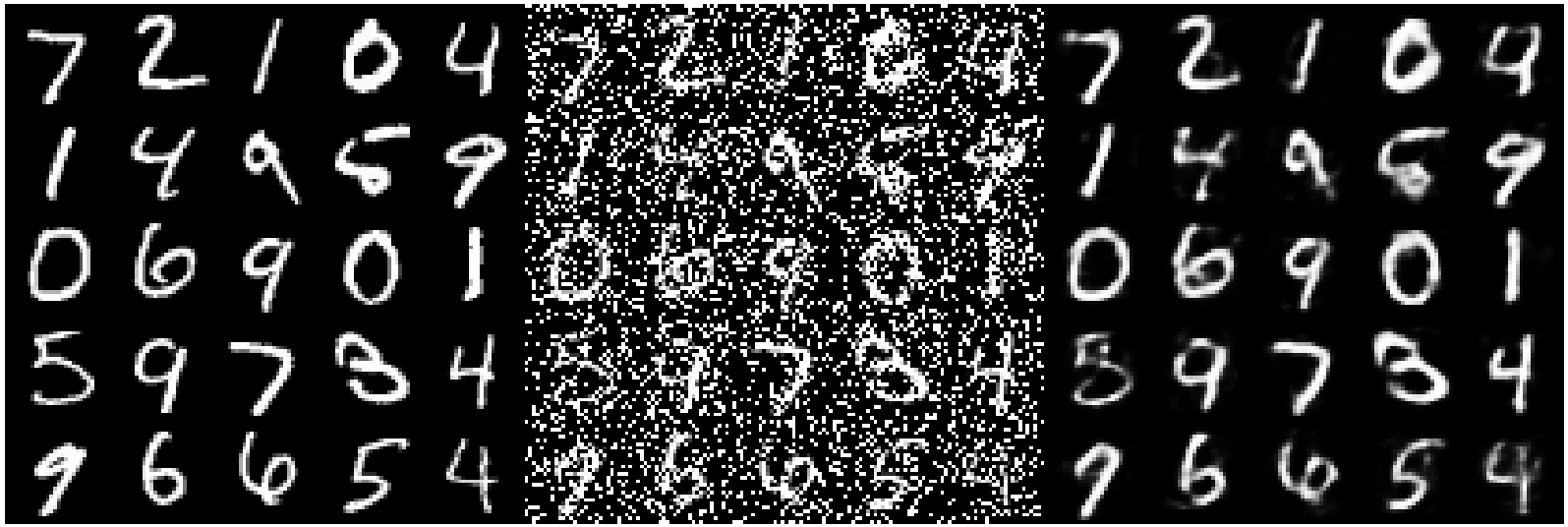


Figura 10 – Resultado da operação de um denoising autoencoder treinado para a base MNIST. À esquerda, dados originais; no centro, entrada ruidosa; à direita, saída do autoencoder. Fonte: https://towardsdatascience.com/denoising-autoencoders-explained-dbb82467fc2

- Para mais detalhes, consultar VINCENT et al. (2010).
- Outras formas de regularização:
  - ✓ [Sparse autoencoders] e [autoencoders with dropout at the hidden layer];
  - ✓ Weight sharing;
  - ✓ Emprego de um decodificador linear (limitação de flexibilidade).

5.2 Pré-treinamento empregando stacked autoencoders

- Apesar do seu elevado potencial de representação, é desafiador treinar uma rede neural profunda a partir de pesos iniciais aleatórios, particularmente pela grande dimensão do problema de otimização associado ao treinamento, pela existência de muitos mínimos locais ruins e pelo grande desafio associado à regularização do mapeamento produzido pela rede neural profunda, dada a reduzida quantidade de dados de entrada-saída, quando comparada com a quantidade de pesos sinápticos ajustáveis (flexibilidade do modelo de aproximação).
- Sendo assim, técnicas que permitam partir de pesos sinápticos de alguma forma mais próximos daqueles que seriam obtidos após um treinamento bem-sucedido são indicadas, sendo denominadas de pré-treinamento.
- Dentre as mais competentes técnicas de pré-treinamento se encontram aquelas baseadas na máquina de Boltzman restrita (HINTON et al., 2006) e nos autoencoders (BENGIO et al., 2006).

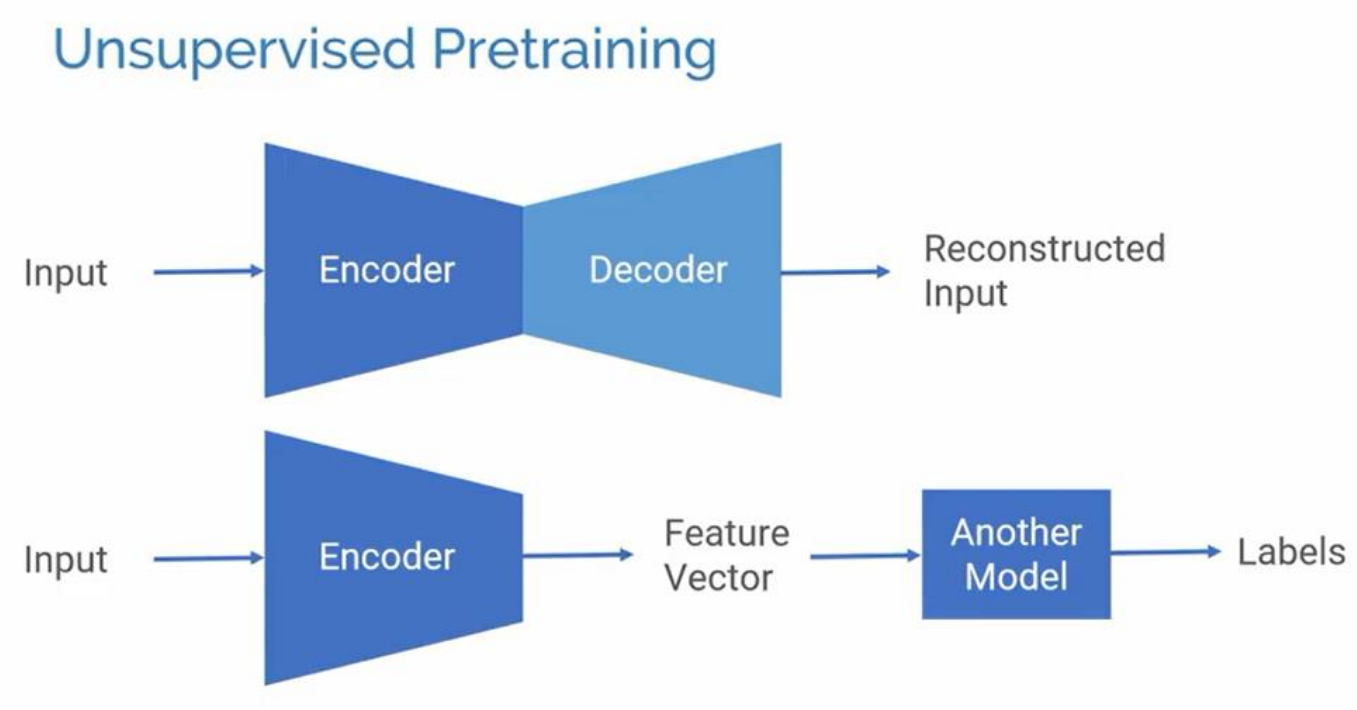


Figura 11 – Pré-treinamento considerando uma camada de codificação. Fonte: https://www.youtube.com/watch?v=P8_W5Wc4zeg

- No caso de autoencoders, cada uma das camadas da rede neural profunda que se entende serem destinadas ao aprendizado da representação devem corresponder ao codificador de um autoencoder treinado para reproduzir a entrada daquela respectiva camada.

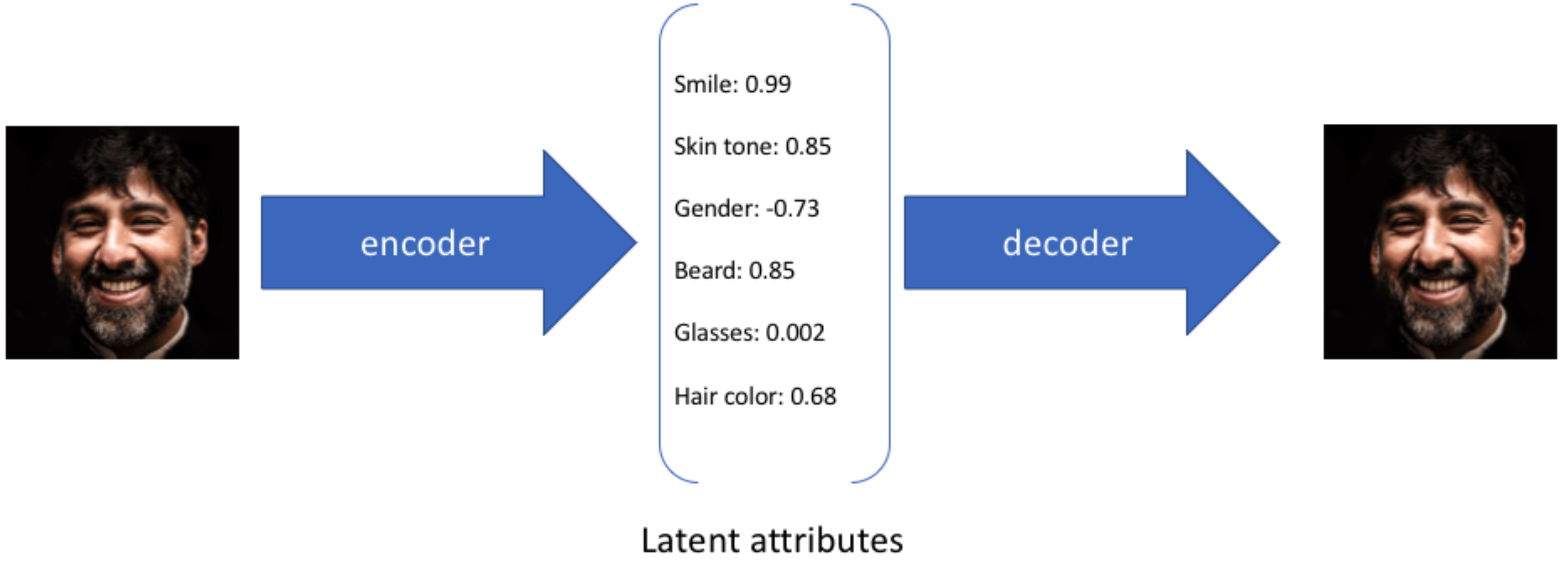
- Sendo assim, a saída do codificador que forma a camada anterior da rede neural é tomada como entrada para o treinamento do autoencoder seguinte, e assim por diante, formando uma cascata de codificadores sintetizados por autoencoders.
- Após montar essas camadas pré-treinadas, promove-se um refinamento pelo treinamento conjunto de toda a rede neural profunda, agora já bem menos suscetível a mínimos locais ruins, por exemplo.
- Dicas práticas de como implementar o pré-treinamento podem ser obtidas em:

  https://machinelearningmastery.com/greedy-layer-wise-pretraining-tutorial/
- No tratamento de imagens, o pré-treinamento camada a camada não é mais comum, tendo sido substituído por técnicas de *transfer learning*, que toma um modelo extrator de características de alto desempenho, treinado em bases de imagens contendo uma quantidade gigantesca de imagens, sendo que apenas a camada de saída é reprojetada para o contexto específico de aplicação.

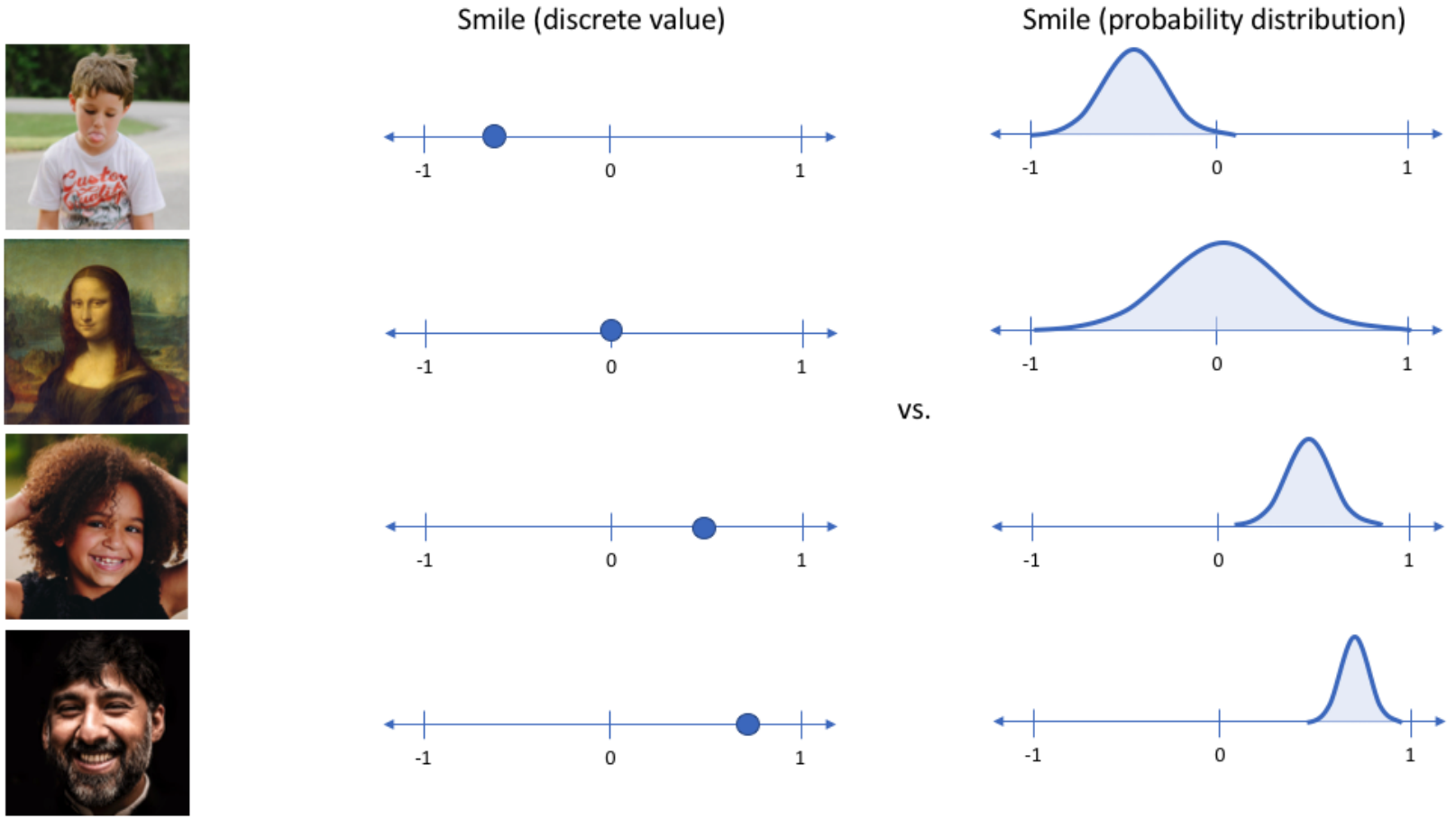
5.3 Variational autoencoders

- Em lugar de mapear a entrada diretamente numa variável latente, o que se busca é mapear a entrada numa distribuição normal para a variável latente, com média e variância.
- As figuras desta seção têm como fonte:

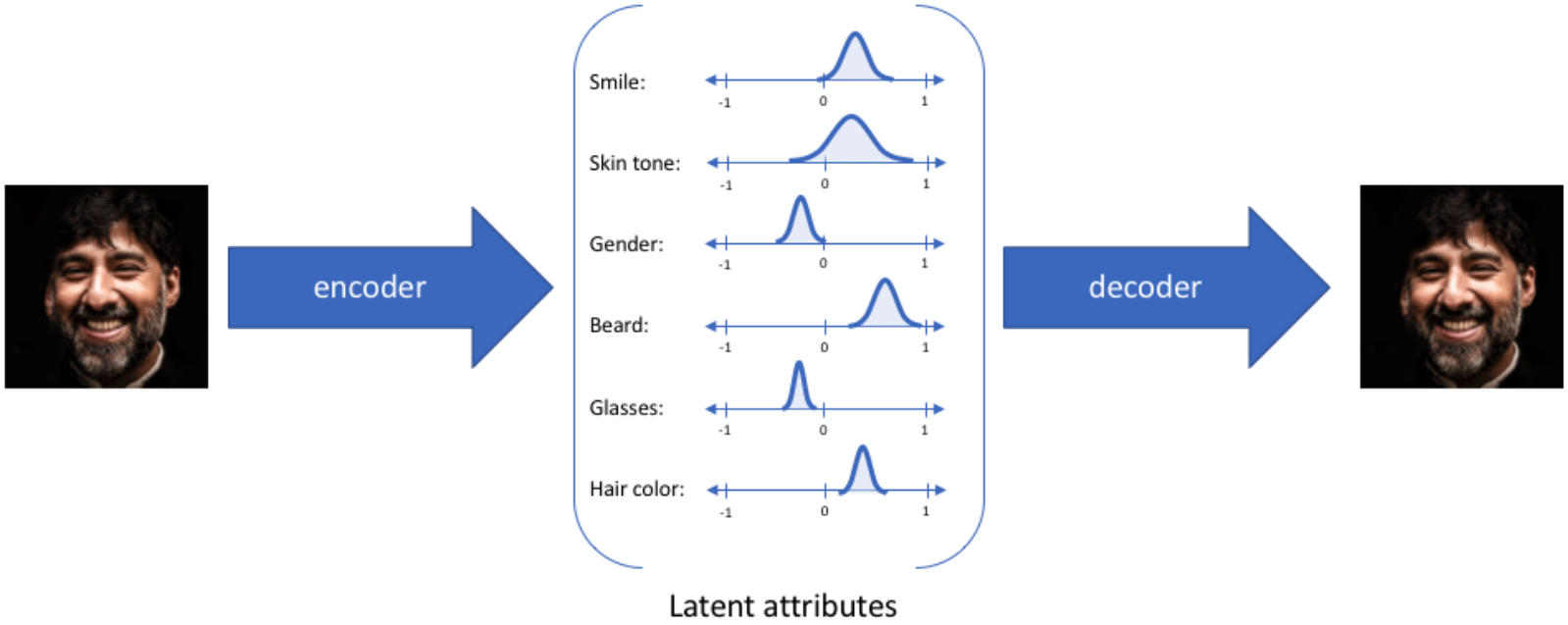
https://www.jeremyjordan.me/variational-autoencoders/



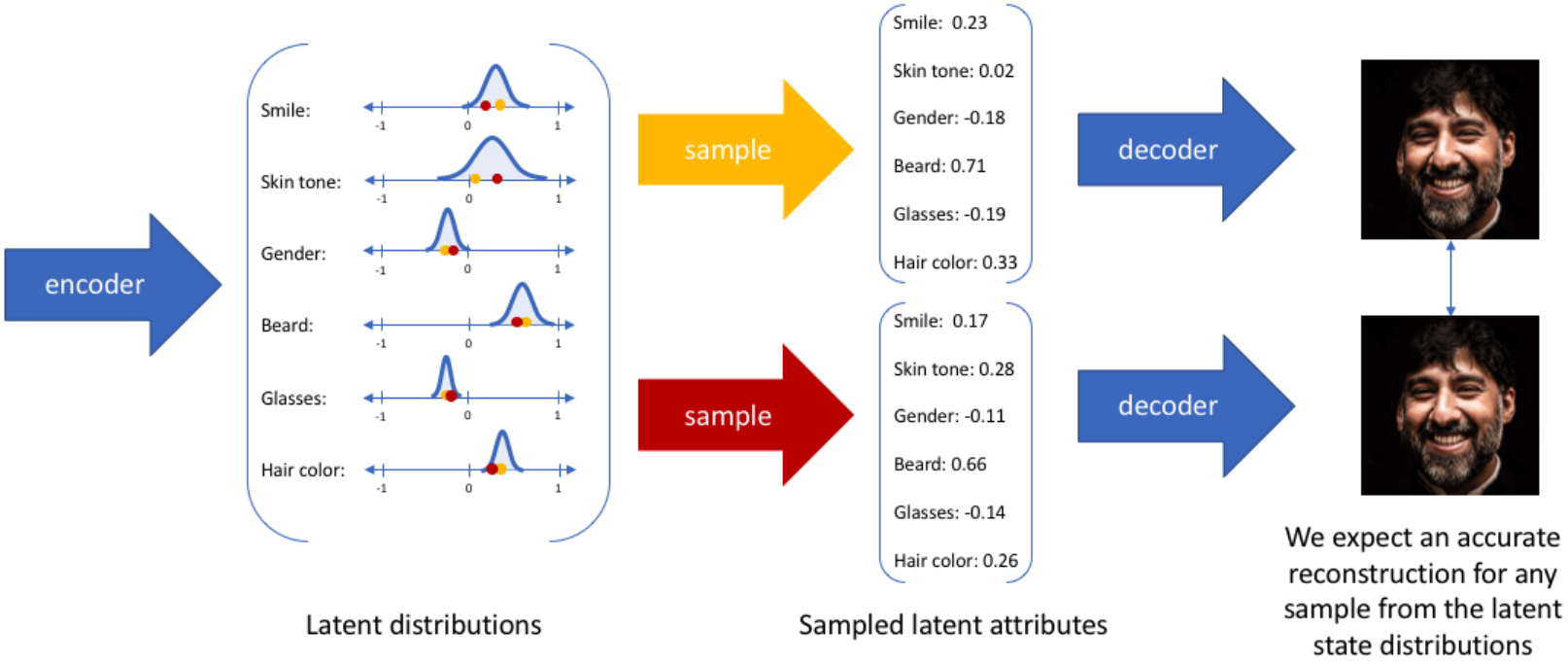
Caso com variáveis latentes determinísticas: autoencoder tradicional



- É possível, assim, definir a distribuição de probabilidade de características fundamentais existentes nos dados de treinamento do autoencoder.



- As variáveis latentes agora são distribuições gaussianas, ou seja, funções densidade de probabilidade.



- Como o decodificador é treinado já considerando a etapa de amostragem, as reconstruções tendem a ser acuradas mesmo para amostragens distintas (amarela e vermelha na figura acima) das distribuições de variáveis latentes.

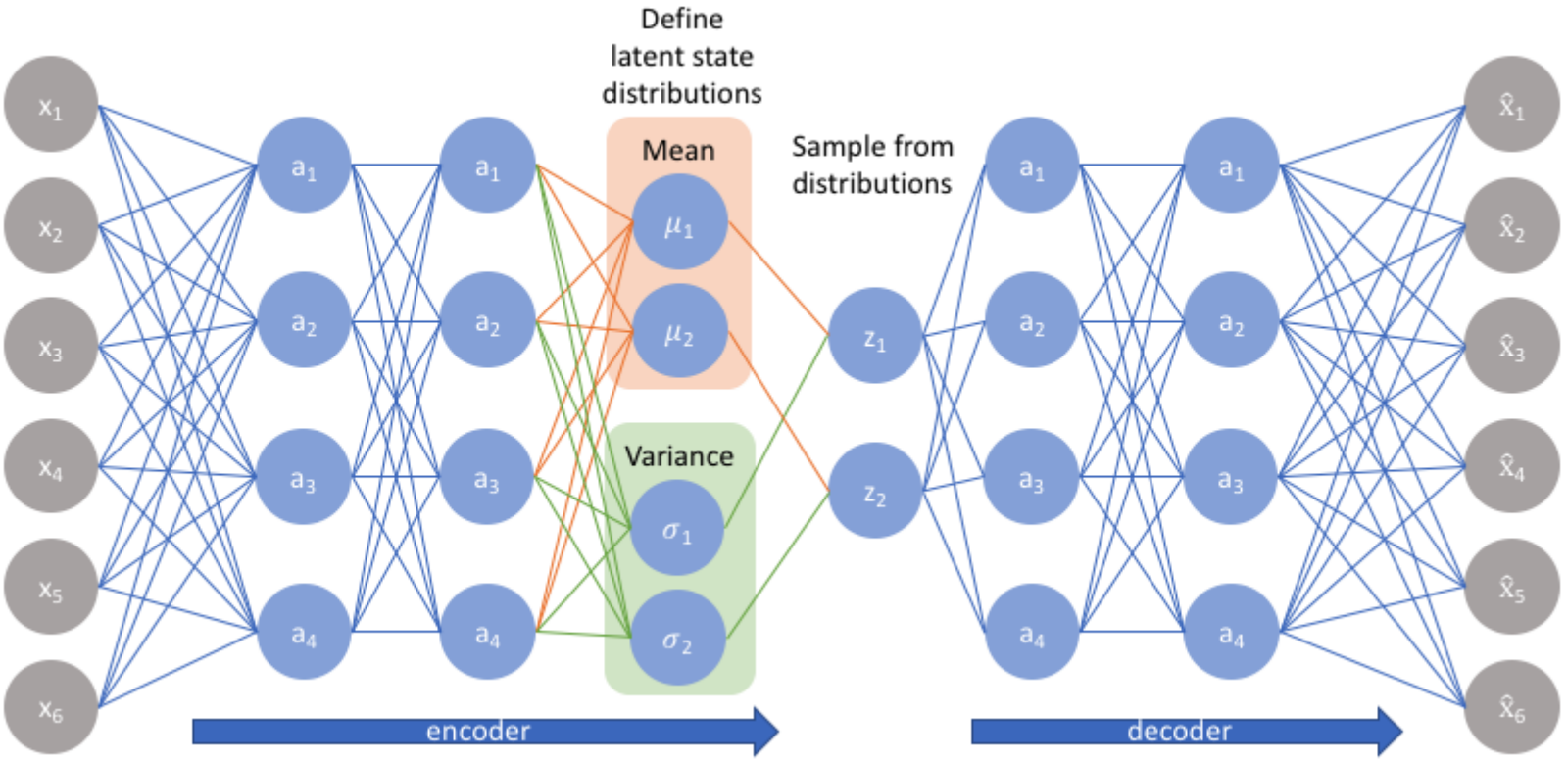


Figura 12 – Configuração do autoencoder variacional considerando duas variáveis latentes.

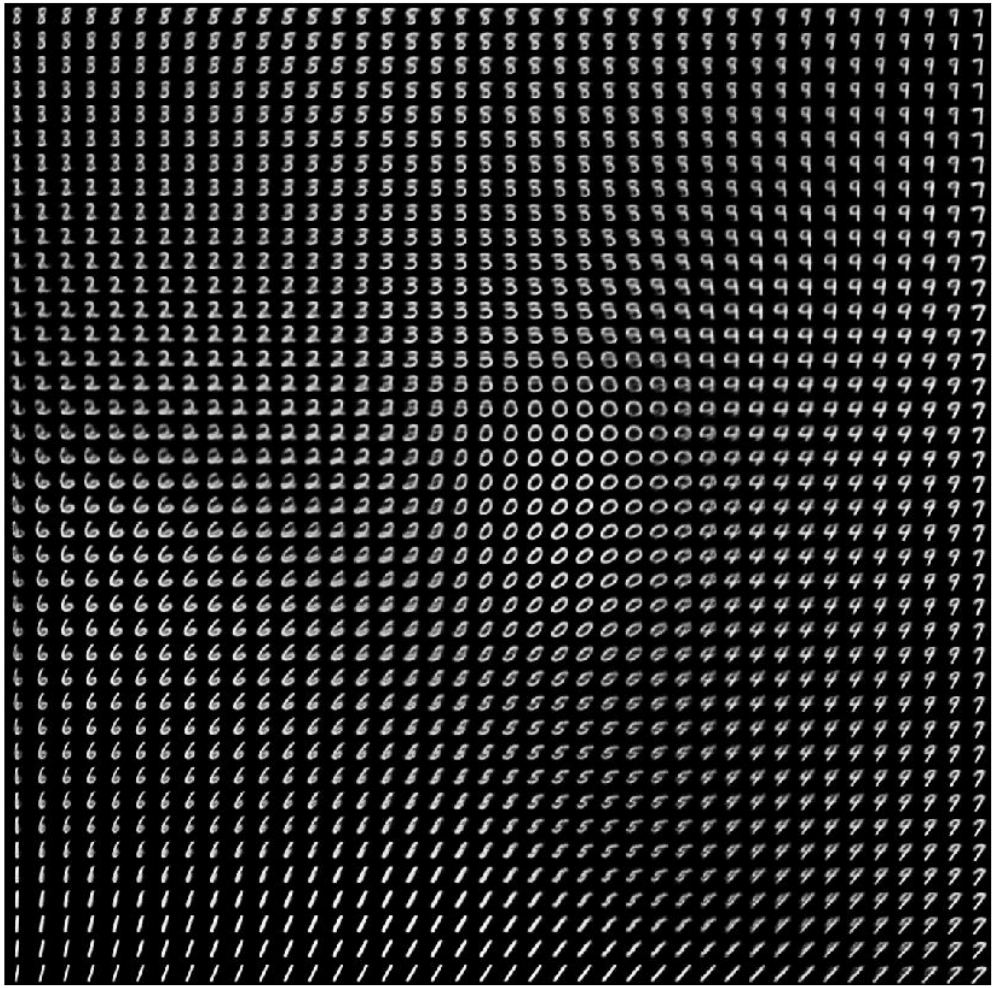


Figura 13 – Visualização das imagens geradas pelo decodificador de um autoencoder variacional com duas variáveis latentes, treinado para os dados MNIST. Foi usada uma variação em grade para as duas variáveis latentes. Cenário equivalente ao da Figura 3.

- A função-objetivo a ser minimizada no treinamento de um autoencoder variacional é a seguinte:

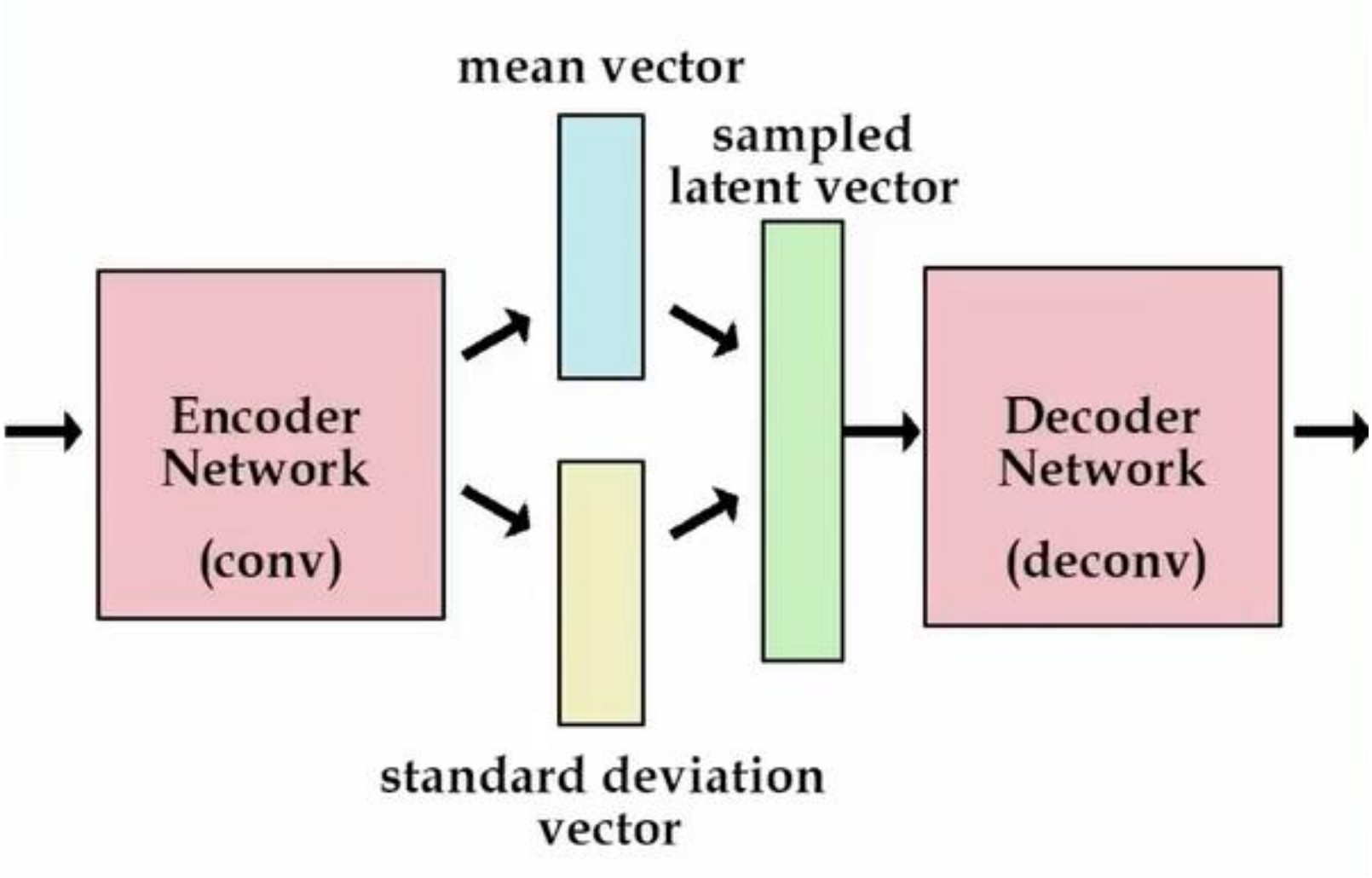
$$L(\theta,\phi;\mathbf{x},\mathbf{z}) = E_{q_\phi(\mathbf{z}|\mathbf{x})}\left\{\log\left[p_\theta(\mathbf{x}|\mathbf{z})\right]\right\} - D_{KL}\left[q_\phi(\mathbf{z}|\mathbf{x})\|p(\mathbf{z})\right]$$

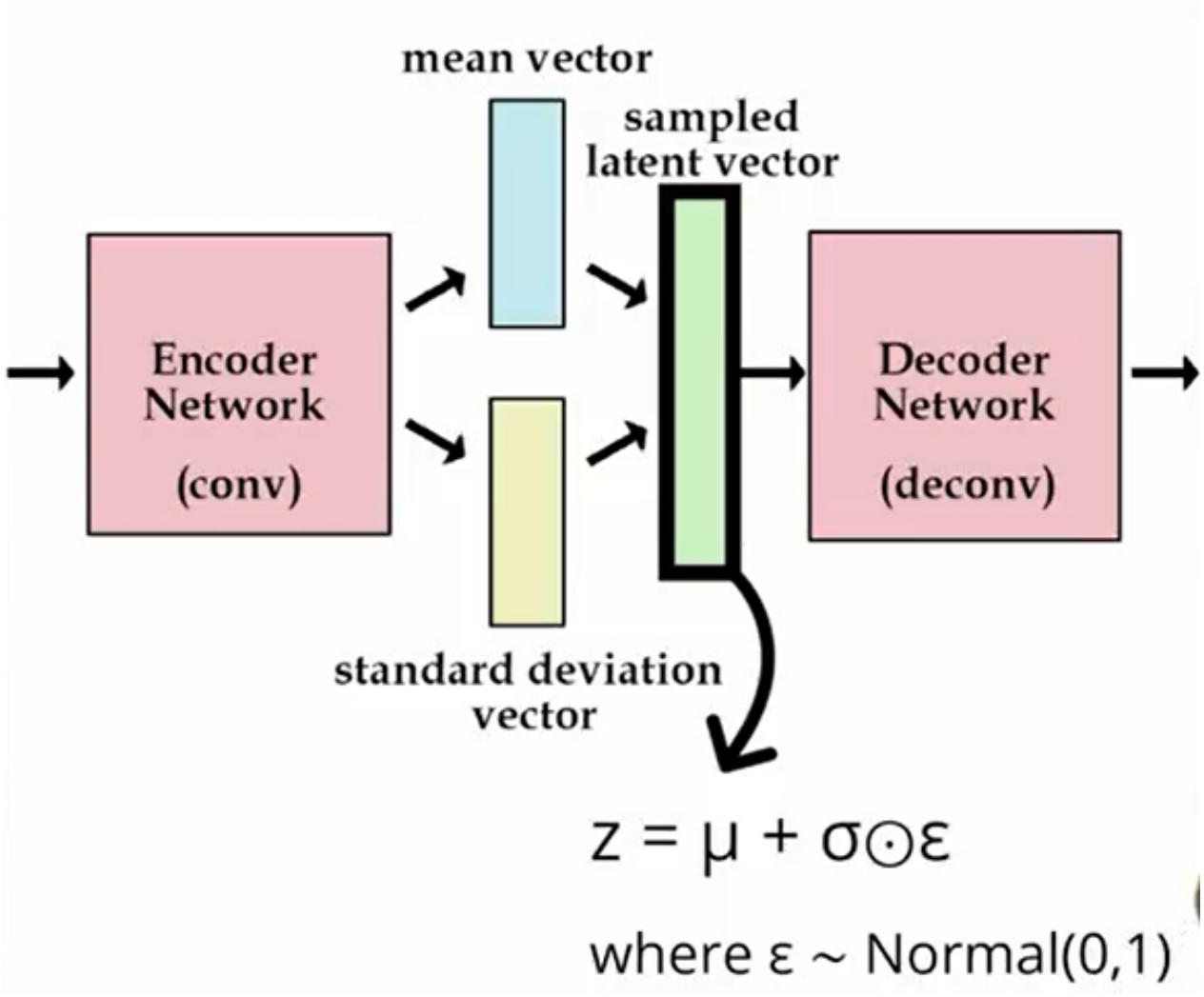
- O primeiro termo penaliza o erro de reconstrução, sendo uma expectativa matemática pelo fato de estarmos operando com distribuições estatísticas agora.
- O segundo termo, que corresponde ao divergente de Kullback-Leibler, penaliza a distância da distribuição resultante de uma distribuição normal com desvio padrão unitário.

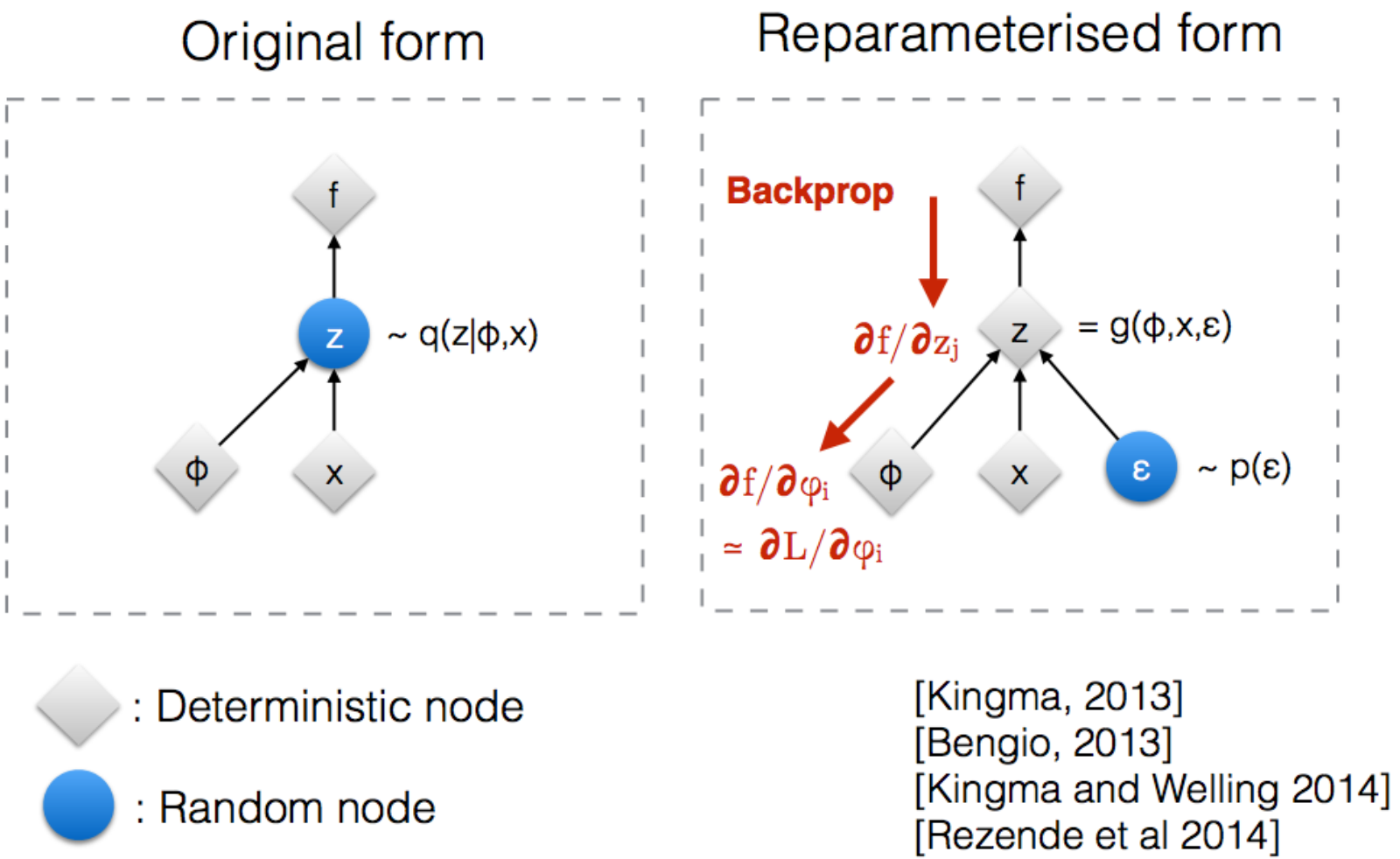
5.4 O truque da reparametrização

- Como retropropagar o erro através de uma operação de amostragem?
- Para maiores detalhes, consultar:
  - ✓ https://stats.stackexchange.com/questions/199605/how-does-the-reparameterization-trick-for-vaes-work-and-why-is-it-important

- ✓ http://blog.shakirm.com/2015/10/machine-learning-trick-of-the-day-4-reparameterisation-tricks/

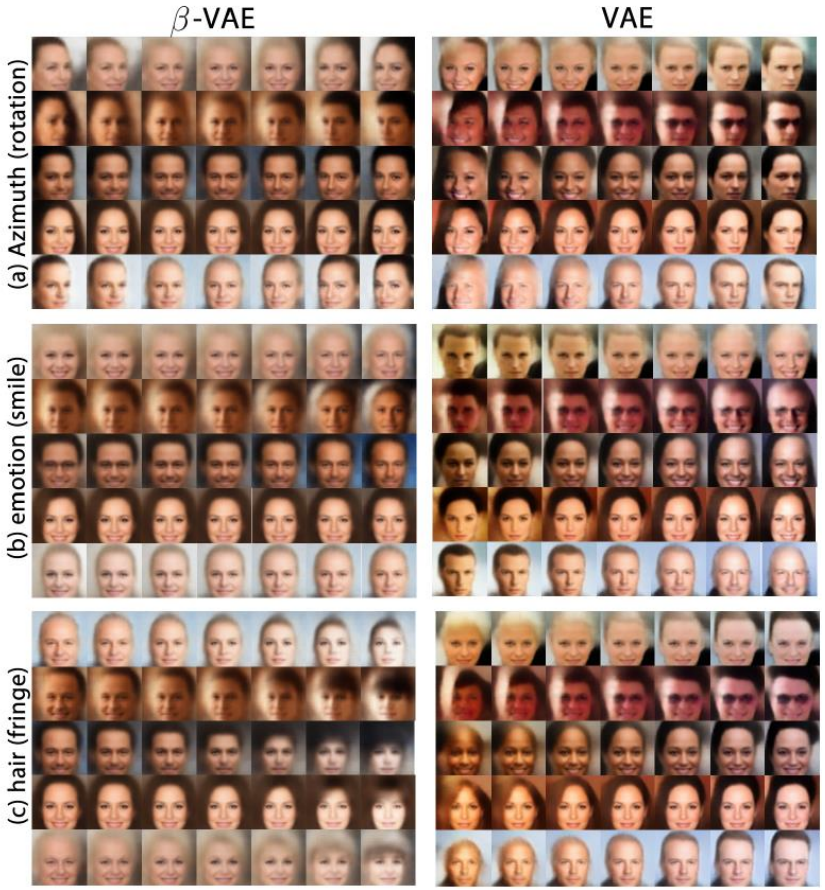



mean vector
sampled
latent vector
Encoder
Network
(conv)
Decoder
Network
(deconv)
standard deviation
vector
z = μ + σ⊙ε
where ε ~ Normal(0,1)


Original form
Reparameterised form
f
z
~ q(z|ϕ,x)
ϕ
x
Backprop
∂f/∂z_j
∂f/∂φ_i
≃ ∂L/∂φ_i
f
z
= g(ϕ,x,ε)
ϕ
x
ε
~ p(ε)
: Deterministic node
: Random node
[Kingma, 2013]
[Bengio, 2013]
[Kingma and Welling 2014]
[Rezende et al 2014]

5.5 Disentagled VAE

$$L(\theta,\phi;\mathbf{x},\mathbf{z},\beta) = E_{q_\phi(\mathbf{z}|\mathbf{x})}\left\{\log\left[p_\theta(\mathbf{x}|\mathbf{z})\right]\right\} - \beta D_{KL}\left[q_\phi(\mathbf{z}|\mathbf{x})\|p(\mathbf{z})\right]$$



6 Restricted Boltzmann machines (RBMs)

- Restricted Boltzmann Machines (RBMs) correspondem a grafos probabilísticos não-direcionados contendo uma camada de variáveis observáveis e uma camada de variáveis latentes, também chamadas de extratores de características.
- RBMs estão entre os mais comuns blocos construtivos de modelos probabilísticos profundos, sendo capaz de identificar automaticamente padrões existentes nos dados de entrada (HINTON et al., 2006). A composição de várias camadas de RBMs é denominada de *deep belief network*.
- As RBMs têm este nome por representarem uma versão restrita da máquina de Boltzmann, apresentada na Figura 14.
- Não iremos abordar aqui a modelagem e o treinamento de RBMs. Apresentamos apenas a derivação que leva à distribuição de probabilidade conjunta entre as unidades visíveis e escondidas. Para mais detalhes sobre o treinamento, favor recorrer a FISCHER & IGEL (2014).

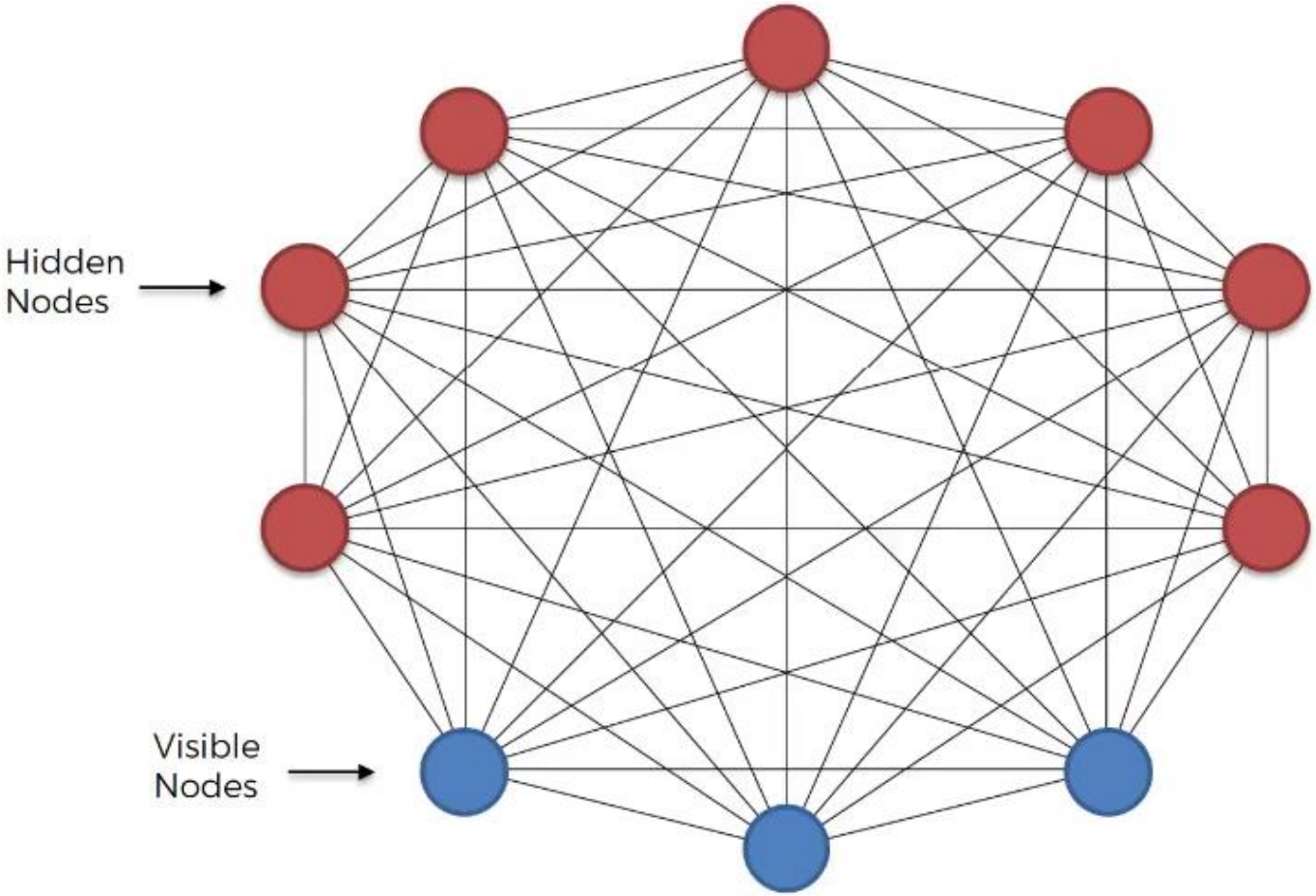


Figura 14 – Máquina de Boltzmann, composta por unidades visíveis e unidades escondidas, sendo que todas as unidades se conectam entre si.
Fonte: https://medium.com/datadriveninvestor/an-intuitive-introduction-of-boltzmann-machine-8ec54980d789

- O conjunto de figuras a seguir foi obtido de:

https://www.youtube.com/watch?v=puux7KZQfsE&t=178s

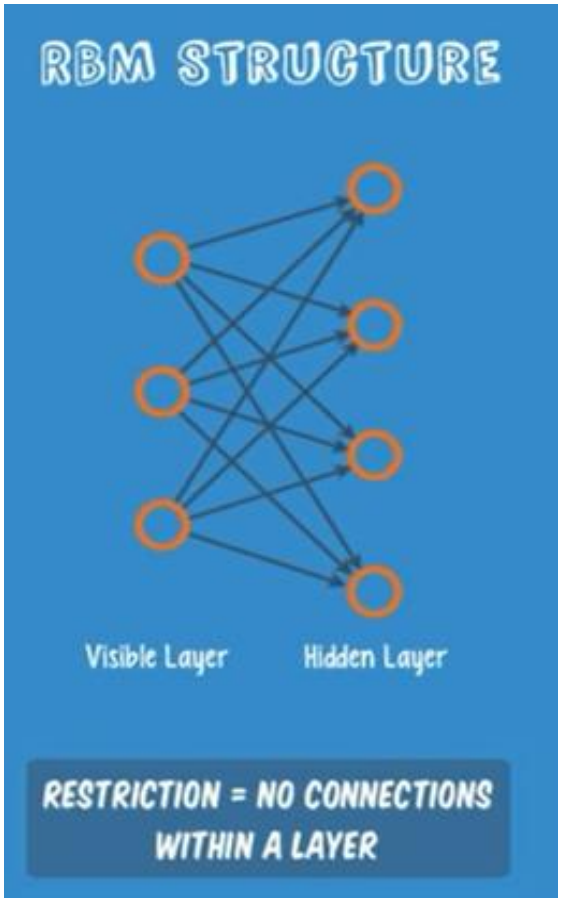
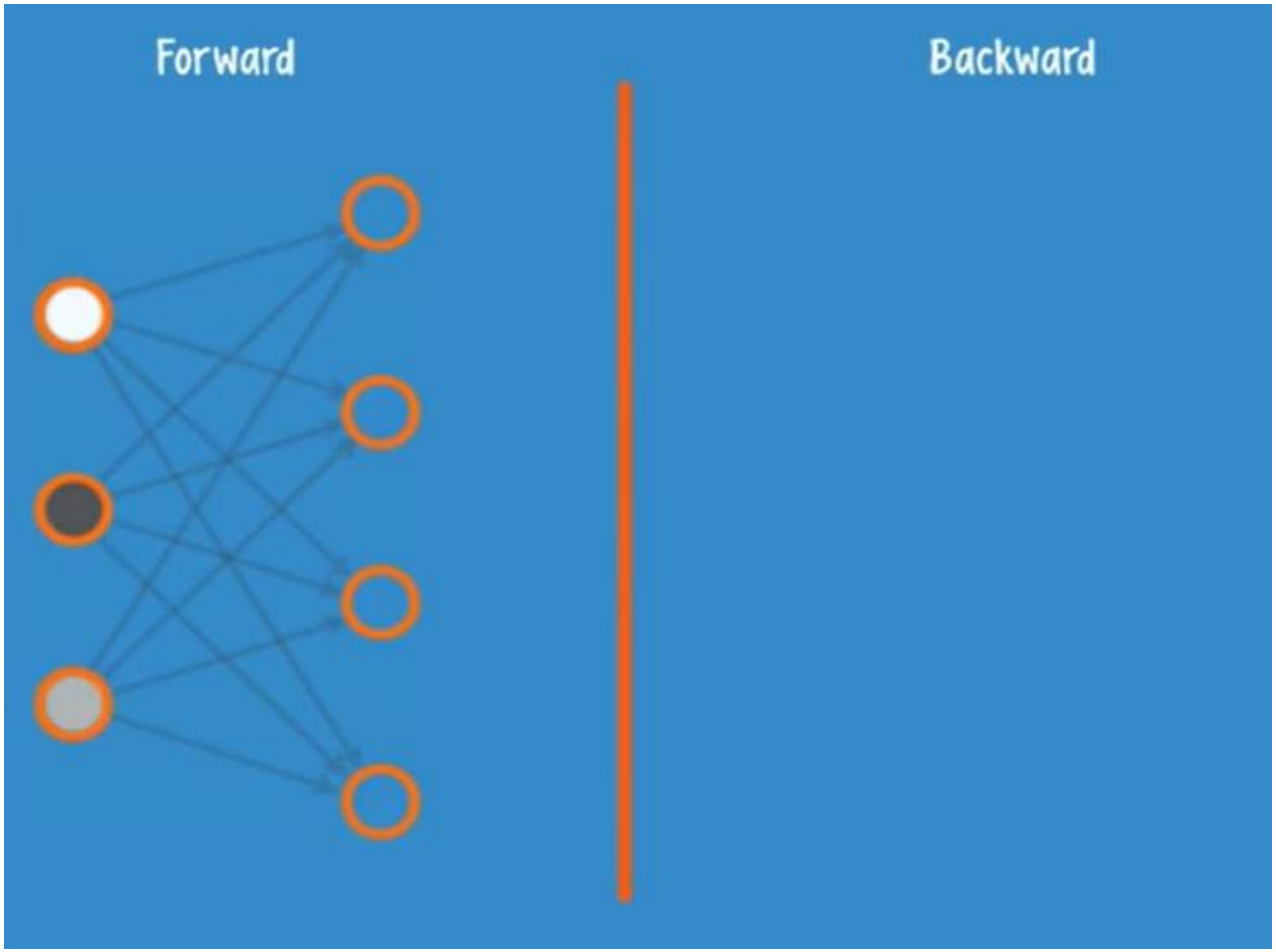
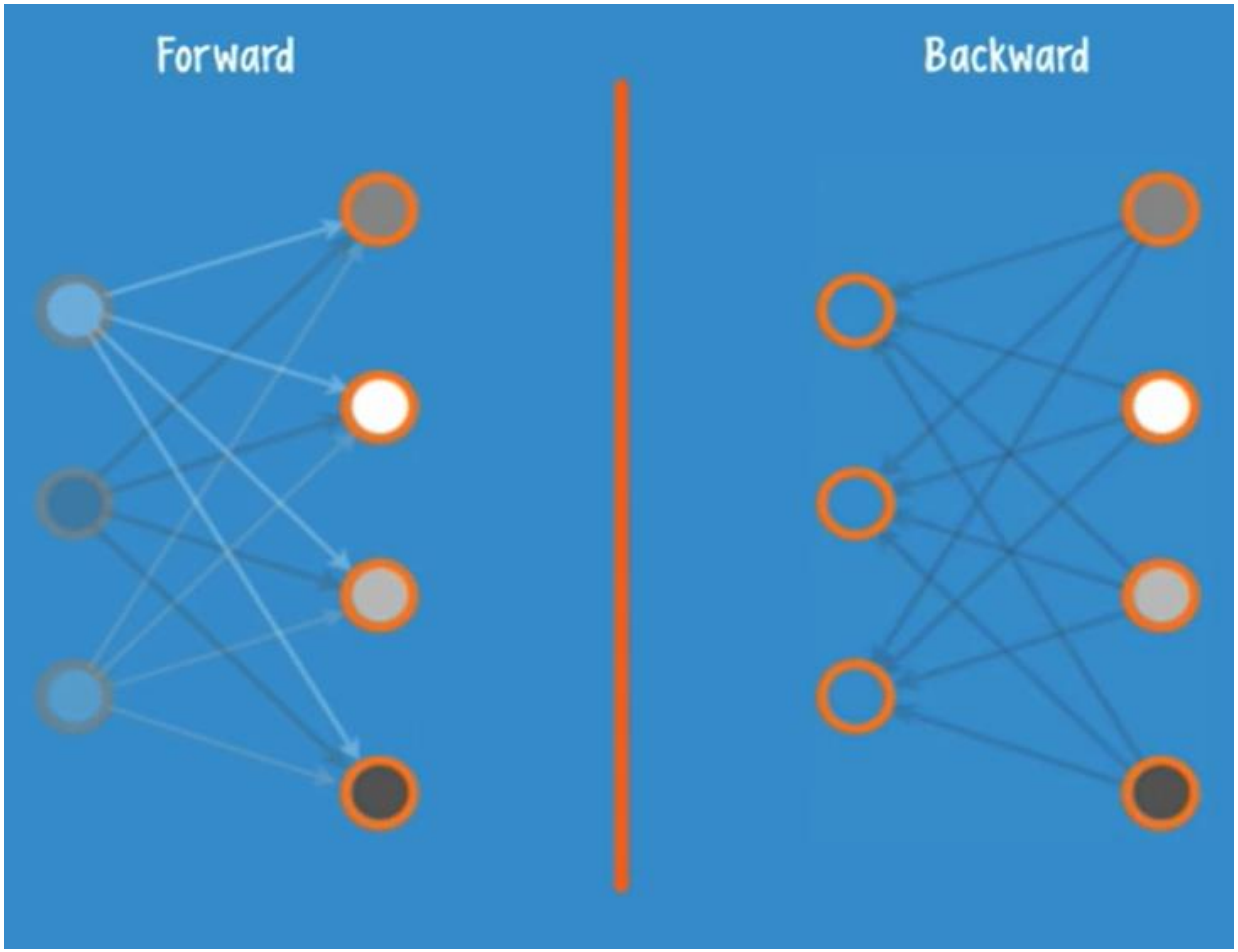


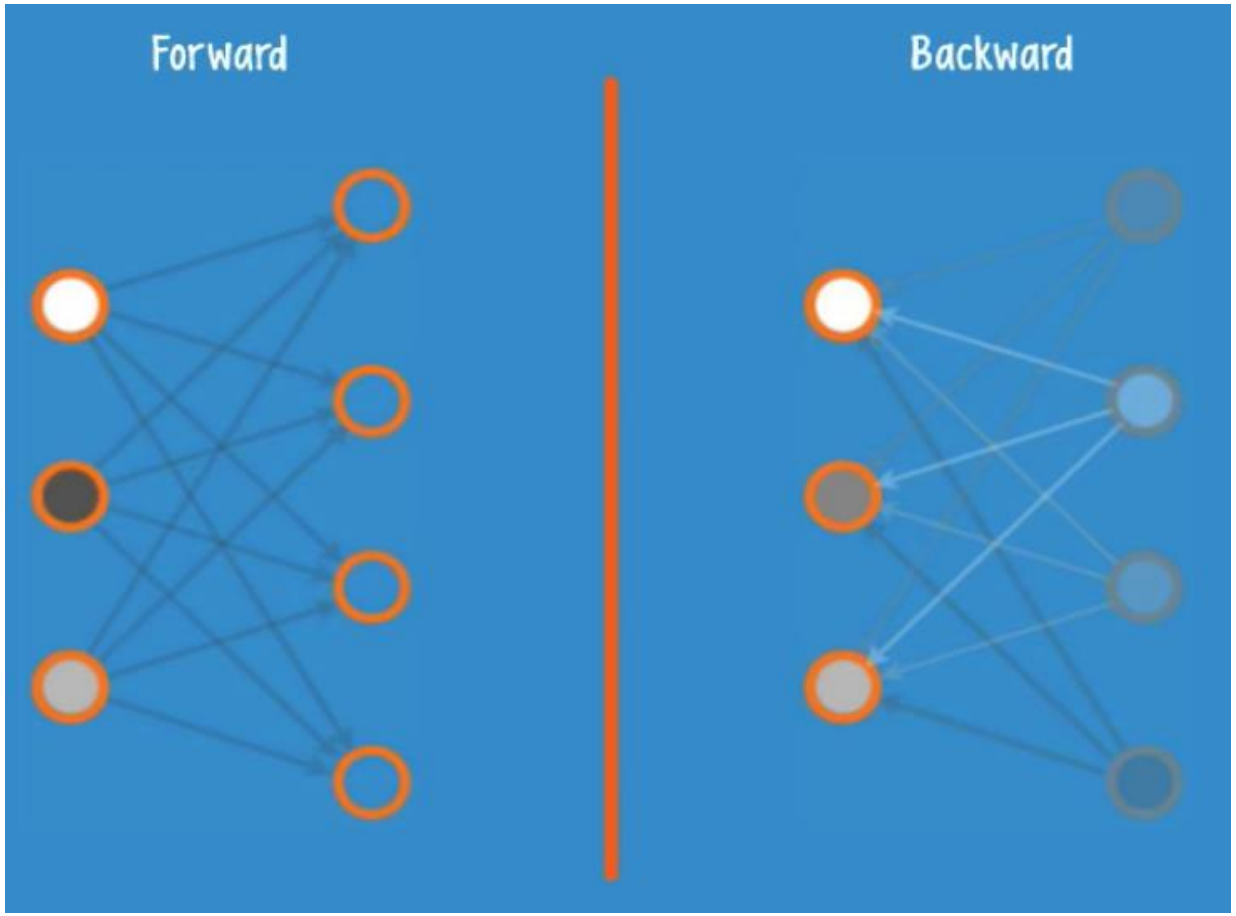
Figura 15 – Máquina de Boltzmann restrita, com conexões apenas entre unidades visíveis e escondidas.



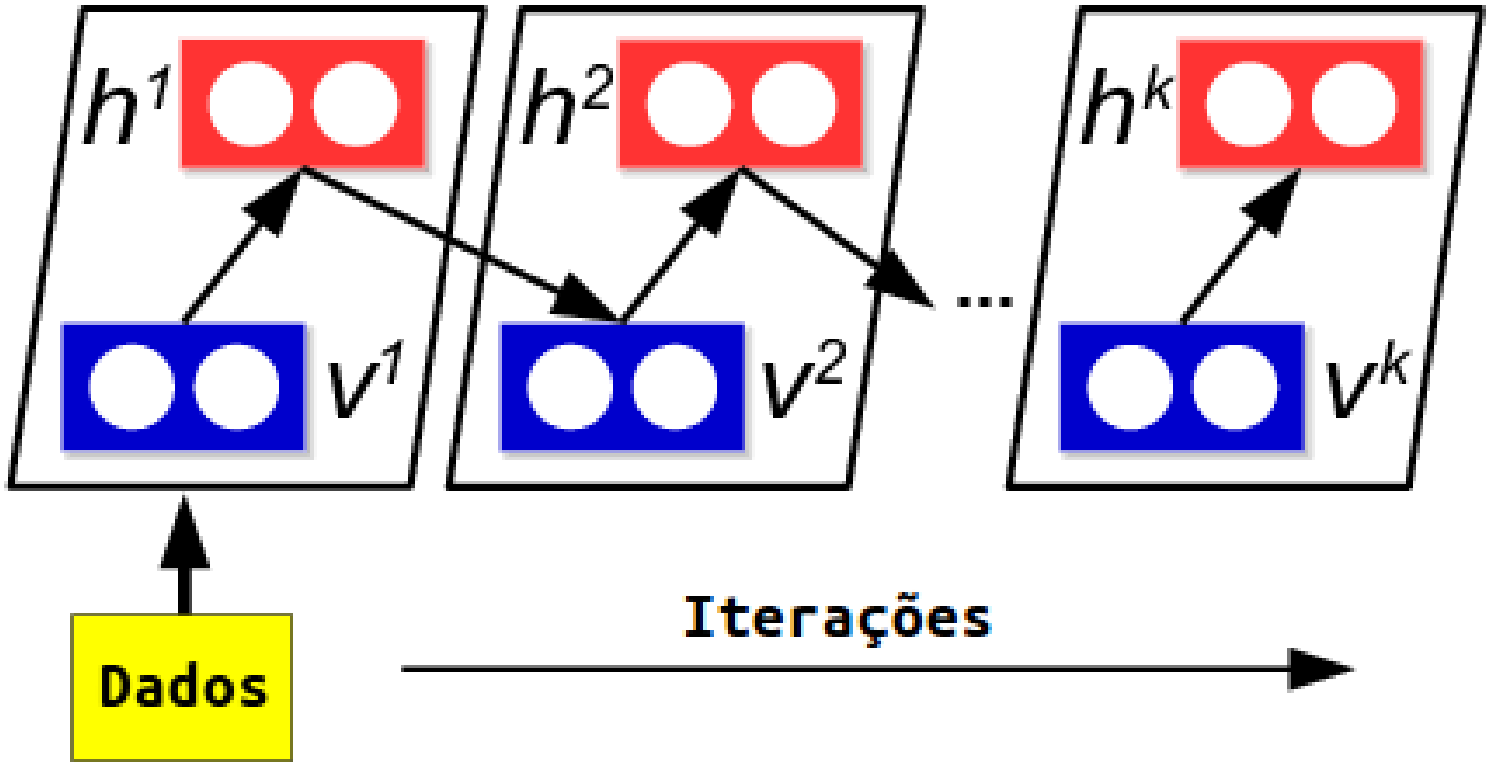
Passo 1: Para cada padrão de entrada, deve-se realizar o passo *forward* e produzir o valor das variáveis latentes.



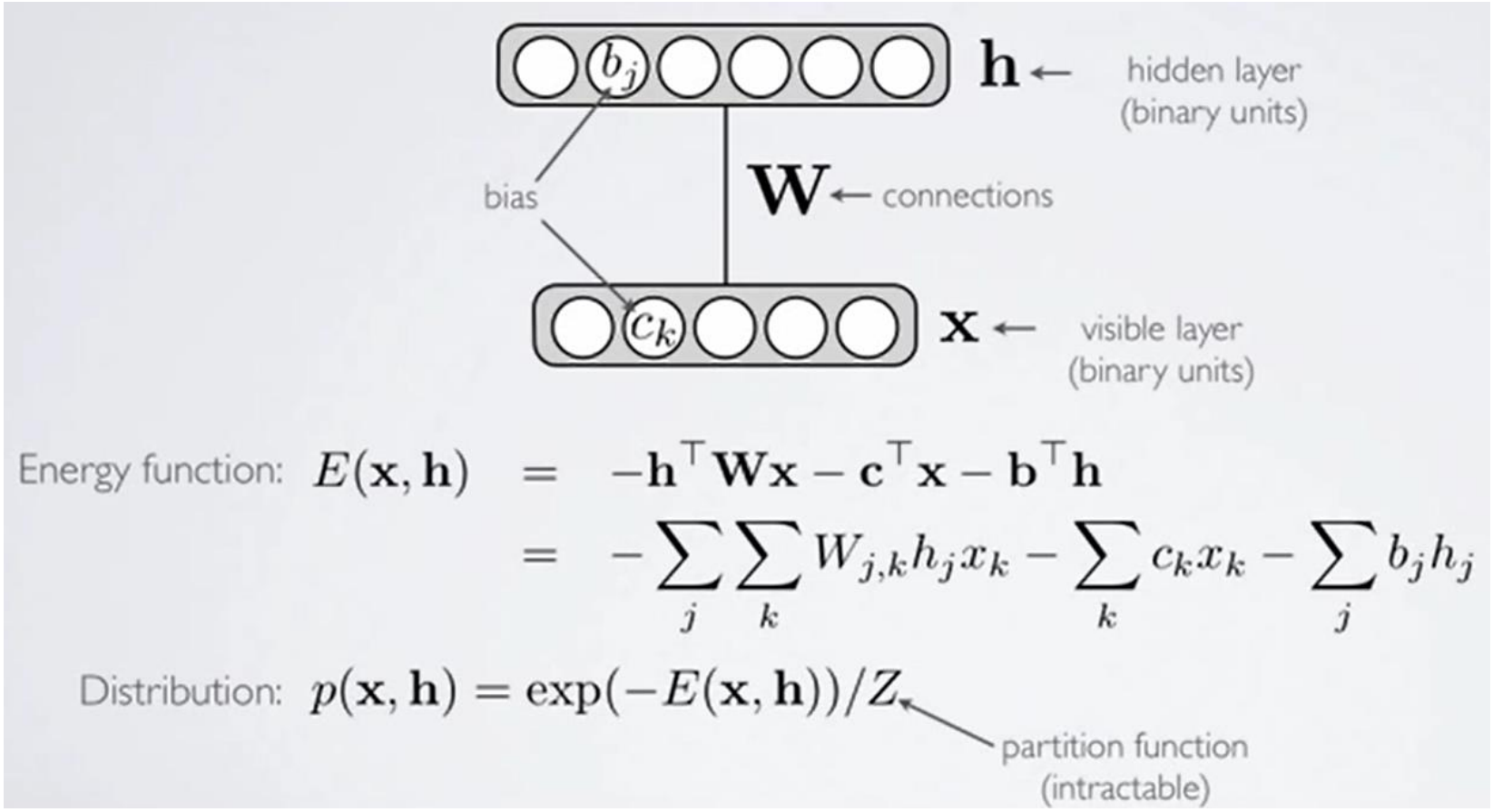
Passo 2: O valor das variáveis latentes do passo *forward* é o ponto de partida para o passo *backward*.



Passo 3: A diferença entre os valores de partida das entradas visíveis e os valores reconstruídos pelo passo backward é usada no ajuste dos pesos sinápticos.

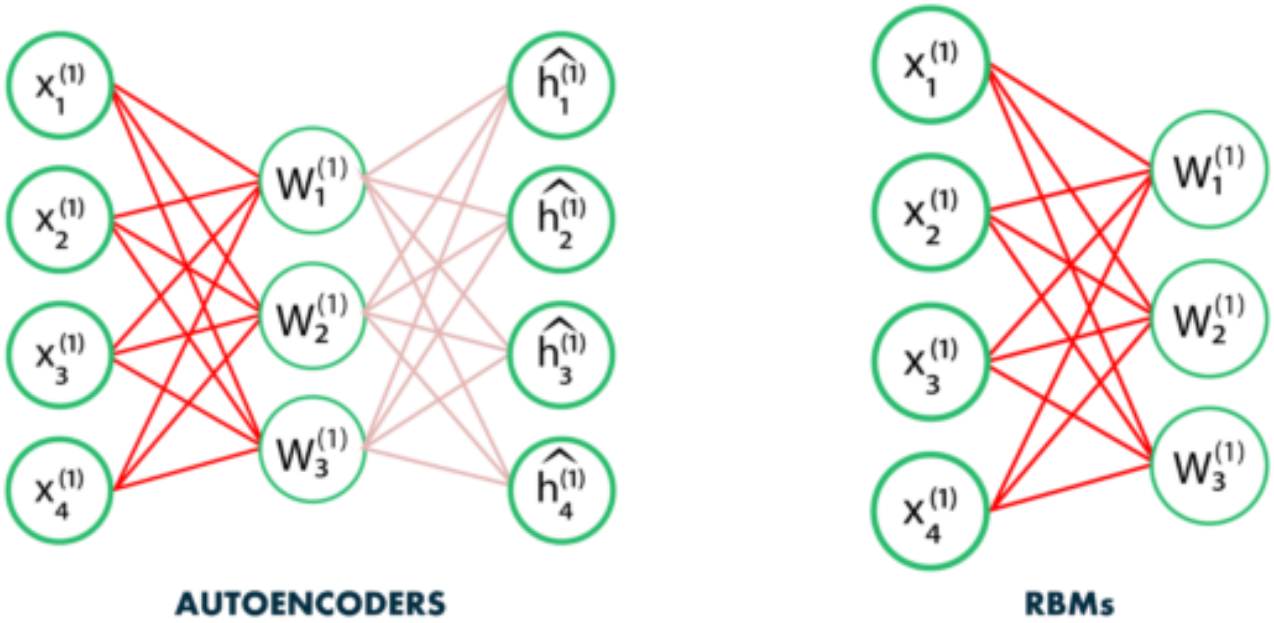


Fonte: https://matheusfacure.github.io/2017/07/30/RBM/



Fonte: https://www.youtube.com/watch?v=lekCh_i32iE

6.1 RBMs × (V)AEs



Fonte: https://www.edureka.co/blog/restricted-boltzmann-machine-tutorial/

- RBMs têm uma fundamentação estatística mais elaborada, que influencia também na técnica de regularização, mas ambas as máquinas buscam o mesmo propósito: codificar as instâncias de entrada de modo que o código gerado possa ser usado para reconstruir a referida instância com erro baixo.

7 Alucinação em redes neurais profundas

- As alucinações são caracterizadas quando a rede neural já treinada produz saídas ao se salientar o papel de algumas camadas da rede neural. É predominantemente voltada para o tratamento de imagens, as quais são chamadas de alucinações, por envolver estímulos internos da rede neural.

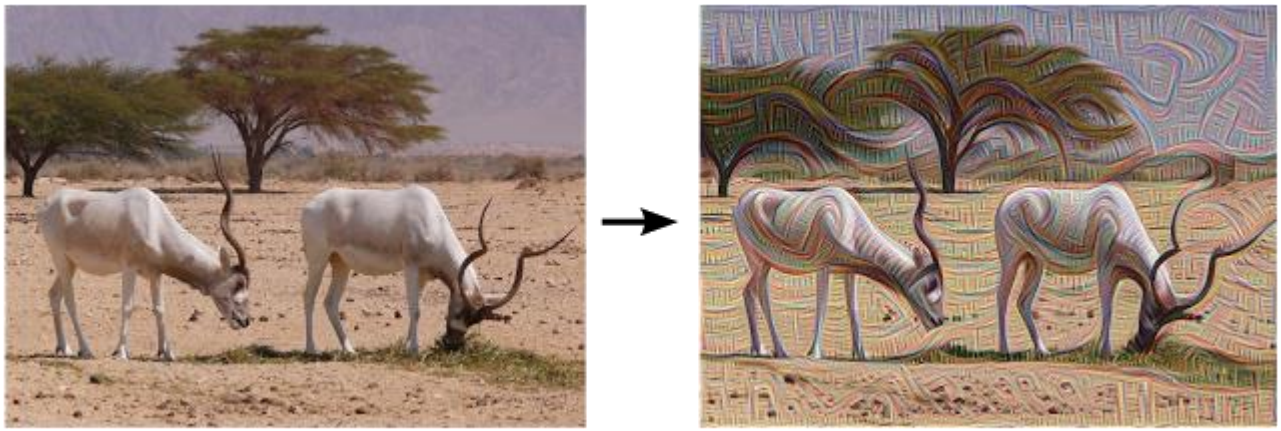
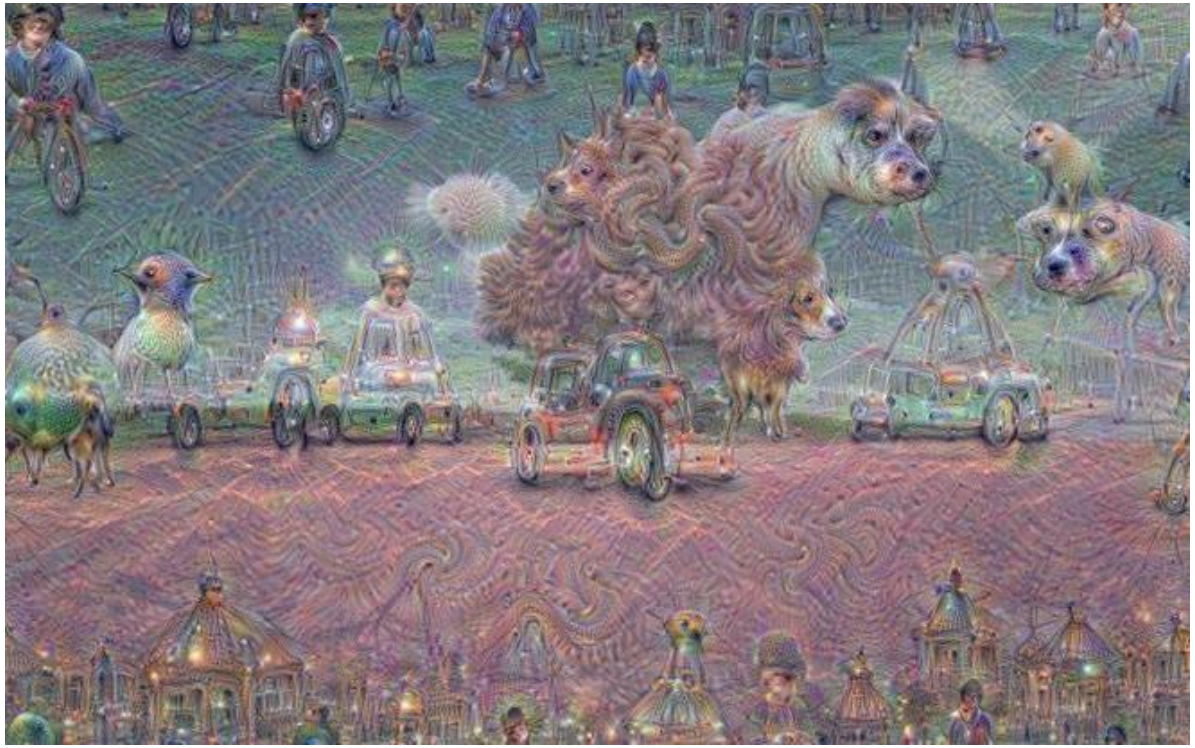


Figura 16 – Efeito de alucinação quando camadas mais próximas da entrada são salientadas.

Fonte: https://medium.com/hashworks/deep-dreaming-with-deep-learning-487835ebf315

- As figuras a seguir já envolvem salientar unidades mais distantes da entrada de uma rede profunda já treinada para a extração de atributos de imagens, produzindo alucinações mais complexas e abstratas, envolvendo uma mistura de padrões já aprendidos.



Fonte: https://medium.com/hashworks/deep-dreaming-with-deep-learning-487835ebf315

Fonte: https://medium.com/hashworks/deep-dreaming-with-deep-learning-487835ebf315

- Essas alucinações permitem constatar como uma rede neural profunda consolidou o aprendizado da representação de imagens durante o treinamento e como esses modelos de classificação podem ser ludibriados.
- Um vídeo contendo resultados recursivos de alucinações se encontra em:

https://www.youtube.com/watch?v=sh-MQboWJug

7.1 Neural inpainting

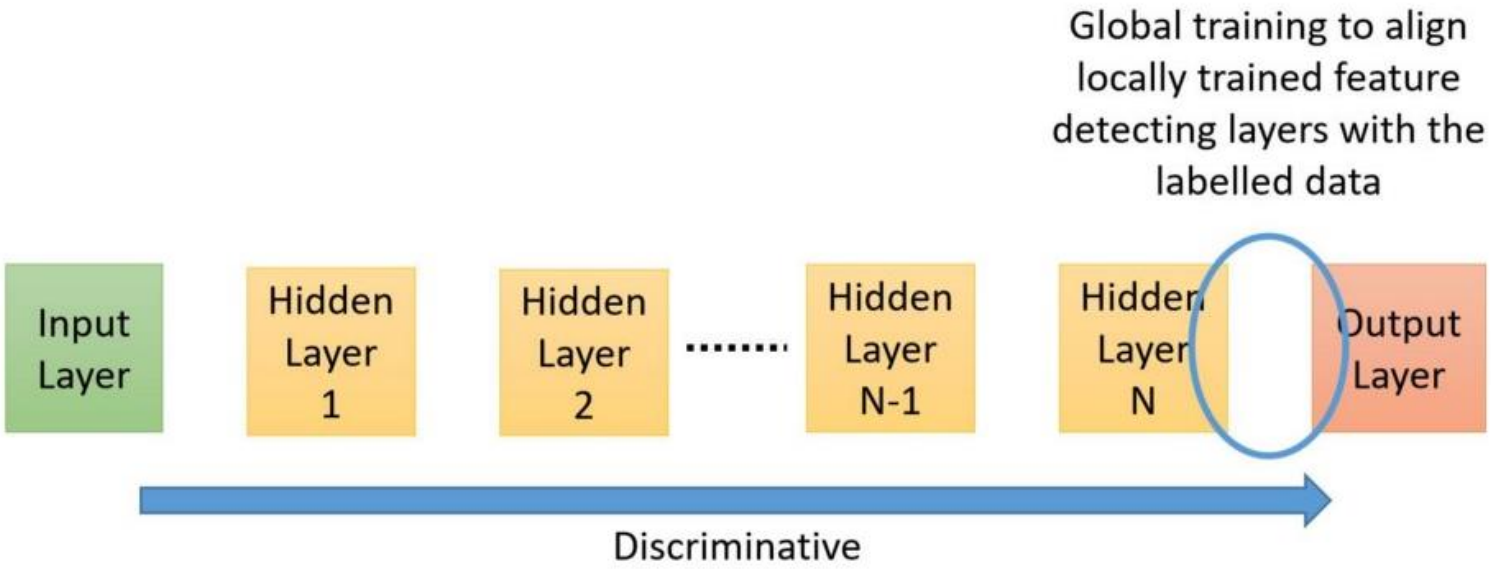
- "Inpainting refers to the art of restoring lost parts of an image and reconstructing them based on the background information."

  Fonte: https://medium.com/jamieai/image-inpainting-with-deep-learning-dd8555e56a32

- Para preencher partes grandes de imagens corrompidas, recorre-se à alucinação de uma rede neural profunda já treinada, explorando e regularizando a última camada da rede neural.

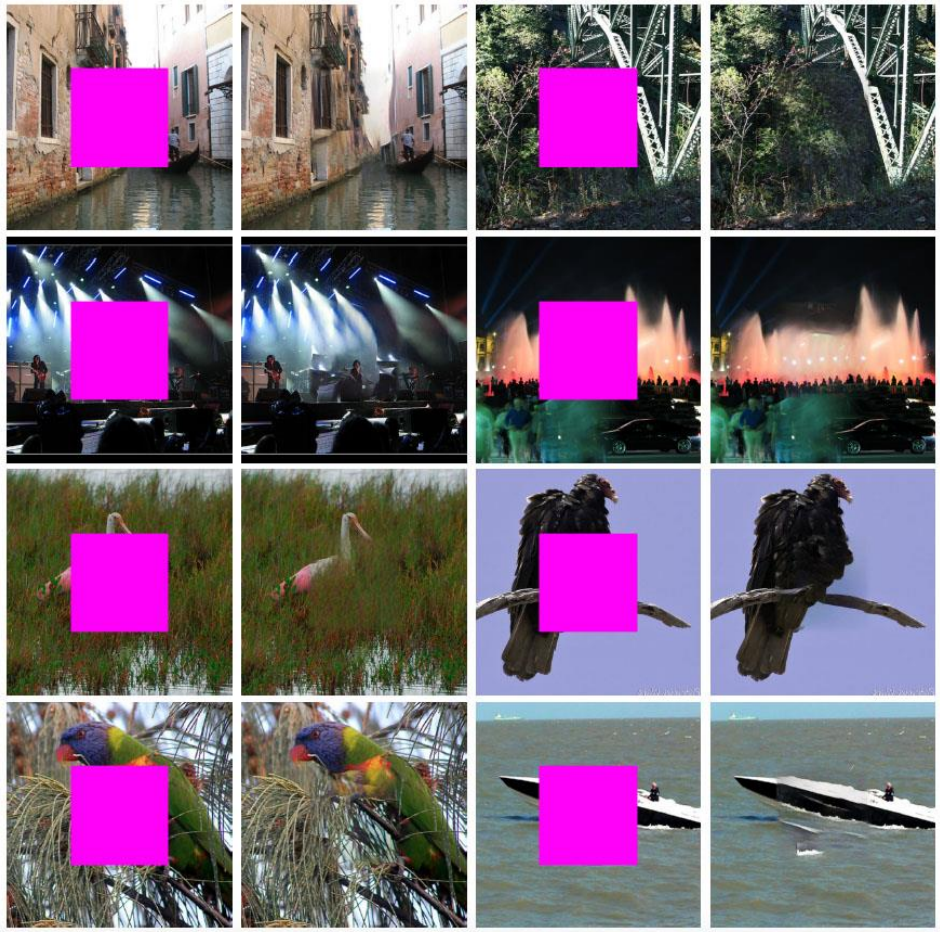
Fonte: https://www.sciencealert.com/nvidia-neural-network-editing-program-for-irregular-inpainting

- A reconstrução das partes faltantes da imagem deve recorrer ao conhecimento adquirido pelo classificador, que potencialmente foi treinado a partir de milhões ou até bilhões de imagens pertencentes ao conjunto de treinamento. Este conhecimento está condensado nas camadas que aprenderam a representação de imagens. Regularização e uma função de perda adequada devem ser buscadas.

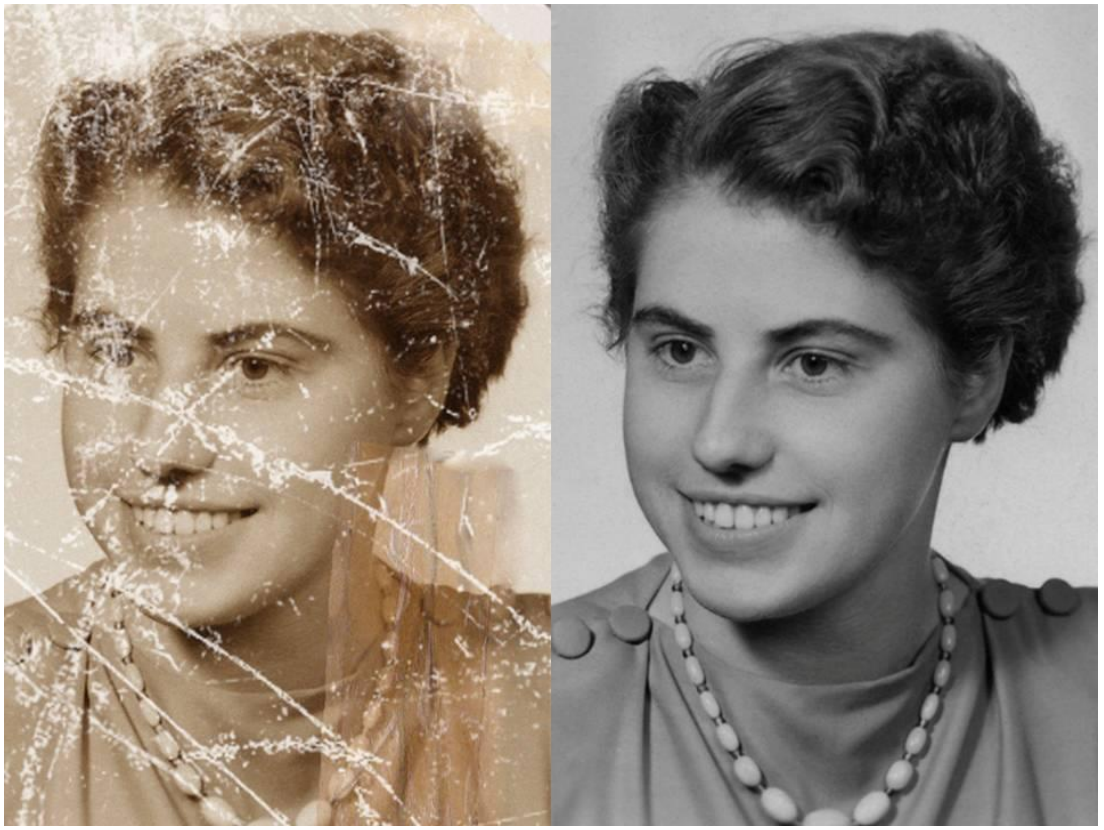


Fonte: https://medium.com/jamieai/image-inpainting-with-deep-learning-dd8555e56a32

- Para mais detalhes, consulte XIE et al. (2012).



Fonte: harryyang.org/inpainting/



Fonte: https://medium.com/jamieai/image-inpainting-with-deep-learning-dd8555e56a32

- Há técnicas alternativas para inpainting, incluindo GANs (*Generative Adversarial Networks*), as quais buscam igualar ou superar o desempenho da mente humana.

- Tende a ser fácil para o ser humano reconstruir uma seção perdida de uma imagem, pois comparamos o contexto com o conhecimento que temos do mundo. Isso nos permite interpolar e/ou extrapolar as partes ausentes.

Fonte: https://medium.com/jamieai/image-inpainting-with-deep-learning-dd8555e56a32

8 Referências bibliográficas

BENGIO, Y.; COURVILLE, A.; VINCENT, P. "Representation Learning: A Review and New Perspectives", IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 35, no. 8, pp. 1798-1828, 2013.

BENGIO, Y.; LAMBLIN, P.; POPOVICI, D.; LAROCHELLE, H. "Greedy Layer-Wise Training of a Deep Network", Advances in Neural Information Processing Systems 19 (NIPS), 2006.

CAYTON, L. "Algorithms for manifold learning", Technical Report CS2008-0923, UCSD, 2005.

BISHOP, C.M. "Pattern Recognition and Machine Learning", Springer, 2006.

CHEN, R.T.Q.; LI, X.; GROSSE, R.; DUVENAUD, D. "Isolating Sources of Disentanglement in VAEs", arXiv:1802.04942v5, 2019.

FISCHER, A.; IGEL, C. "Training Restricted Boltzmann Machines: An Introduction", Pattern Recognition, vol. 47, pp. 25-39, 2014.

GOODFELLOW, I.; BENGIO, Y.; COURVILLE, A. "Deep Learning", MIT Press, 2016.

HASTIE, T.; TIBSHIRANI, R.; FRIEDMAN, J.H. "The Elements of Statistical Learning: Data Mining, Inference, and Prediction", Springer, 2009.

HIGGINS, I.; MATTHEY, L.;PAL, A.; BURGESS, C.; GLOROT, X.; BOTVINICK, M.; MOHAMED, S.; LERCHNER, A. "β-VAE: Learning Basic Visual Concepts with a Contrained Variational Framework", ICLR, 2017.

HINTON, G.E. "A Practical Guide to Training Restricted Boltzmann Machines", UTML TR 2010-003, 2010.

HINTON, G.E.; OSINDERO, S.; TEH, Y.W. "A fast learning algorithm for deep belief nets", Neural Computation, vol. 18, no. 7, pp. 1527-1554, 2006.

NARAYANAN, H.; MITTER, S. "Sample complexity of testing the manifold hypothesis", Advances in Neural Information Processing Systems 23 (NIPS), 2010.

VINCENT, P.; LAROCHELLE, H.; LAJOIE, I.; BENGIO, Y. & MANZAGOL, P.-A. "Stacked Denoising Autorencoders: Learning Useful Representations in a Deep Network with a Local Denoising Criterion", Journal of Machine Learning Research, vol. 11, pp. 3371-3408, 2010.

XIE, J.; XU, L.; CHEN, E. "Image Denoising and Inpainting with Deep Neural Networks", Advances in Neural Information Processing Systems 25 (NIPS), 2012.