Capítulo 2

Introdução a Modelos de Difusão para Síntese de Imagens

Manuel Menezes de Oliveira Neto (UFRGS)

Abstract

Diffusion models have emerged as the dominant paradigm for high-quality image and video synthesis. These models generate visual content by reversing a gradual stochastic process that transforms training data into Gaussian noise while conditioning the generation process on guiding inputs such as text prompts. This tutorial provides a gentle yet technically rigorous introduction to diffusion models for image synthesis. Its goal is to help readers develop an intuitive understanding of the principles and techniques that enable these models to produce high-quality and diverse results.

Resumo

Modelos de difusão constituem o paradigma dominante para a síntese de imagens e vídeos de alta qualidade. Esses modelos geram conteúdos visuais ao reverter um processo estocástico gradual que transforma dados de treinamento em ruído Gaussiano, enquanto condicionam o processo de geração por meio de instruções, como o uso de descrições textuais. Este tutorial oferece uma introdução acessível porém tecnicamente rigorosa a modelos de difusão para síntese de imagens. Seu objetivo é apresentar de forma intuitiva os princípios e técnicas que lhes permitem produzir resultados de alta qualidade.

Este trabalho foi financiado com recursos do CNPq (Processo 305474/2022-7) e CAPES (Finance Code 001). Ferramentas de IA Generativa, incluindo ChatGPT e Gemini foram empregadas para geração de ilustrações. ChatGPT foi utilizado na revisão textual.

2.1. Introdução

Sistemas de IA generativa como Midjourney [Midjourney, Inc. 2026], Nano Banana Pro [Open AI 2026], DALL·E 3 [Betker et al. 2023], Imagen [Saharia et al. 2022], Stable Diffusion [Rombach et al. 2022] e Flux [Black Forest 2026] são capazes de produzir imagens de alta qualidade em resposta a uma descrição textual apresentada pelo usuário. Por sua vez, sistemas como Sora [OpenAI 2026], Runway Gen-4.5 [Runway AI 2026], Kling [Kuaishou 2026], Luma Dream Machine [Labs 2026], Deevid [AI 2026] e Wan [Alibaba 2026] geram vídeos de alta qualidade a partir de *prompts* de texto. Dentro de suas respectivas classes, cada um desses sistemas produz resultados que se destacam por certas características. Apesar de suas diferenças, todos eles têm algo em comum: utilizam **modelos de difusão** como mecanismo para síntese de imagens. Modelos de difusão constituem o paradigma dominante para a síntese de imagens e vídeos de alta qualidade a partir de descrições textuais. A Figura 2.1 mostra uma imagem gerada por um modelo de difusão com base em uma descrição. Observe a riqueza de detalhes gerada para esta cena virtual. O objetivo deste tutorial é apresentar de forma intuitiva, porém tecnicamente rigorosa, os princípios e as técnicas que tornam isso possível.

Figura 2.1. Imagem gerada utilizando um modelo de difusão (DALL·E 3 [Betker et al. 2023]) com a descrição: "Um ursinho andando de bicicleta em um parque, usando óculos escuros, um boné azul e uma camisa com a palavra INF".

2.2. Modelagem Generativa Como Um Processo de Amostragem

Como é possível gerar imagens como a mostrada na Figura 2.1 a partir de uma descrição textual? A resposta é: por meio de um **processo de amostragem** condicionada. Mas antes que possamos explicá-lo em detalhes, é necessário entender como gerar imagens (*i.e.*, como realizar amostragem) sem condicionamento ao texto. Vamos iniciar introduzindo algumas ideias-chave, a partir das quais o procedimento se torna bastante intuitivo.

Ideia-chave 1: **Modelagem generativa** pode ser entendida como o **processo de aprender uma distribuição de probabilidades** a partir de um conjunto de dados utilizado para treinamento, de modo que novas amostras semelhantes às deste conjunto de dados possam ser obtidas (geradas) amostrando-se esta distribuição. Uma **distribuição de probabilidades** para um conjunto de dados de treinamento é uma descrição matemática de quão prováveis são diferentes amostras de acordo com os dados utilizados no treinamento.

Modelos de difusão podem ser utilizados para geração de dados de diversas modalidades, como imagens [Betker et al. 2023], vídeos [OpenAI 2026], modelos 3D [Wang et al. 2025], nuvens de pontos [Luo and Hu 2021] e estruturas moleculares [Abramson et al. 2024], entre outras. Todas essas modalidades podem ser manipuladas utilizando representações vetoriais. Assim, uma imagem contendo H linhas, W colunas e C canais de cores pode ser representada por um vetor $z \in R^{H \times W \times C}$. Um vídeo corresponde a uma sequência de imagens ao longo de um período de tempo T e pode ser representado por um vetor $z \in R^{T \times H \times W \times C}$. Por sua vez, um modelo 3D, uma nuvem de pontos, ou uma estrutura molecular pode ser representado(a) por um vetor $z \in R^{3 \times N}$, onde N corresponde, respectivamente, ao número de vértices do modelo 3D, ao número de pontos da nuvem, ou ao número de átomos da molécula, cada um desses correspondendo a um elemento $w \in R^3$. No restante do texto, focaremos no caso de geração de imagens e, assim, o termo *amostra* será interpretado como *imagem*. Ressalte-se, entretanto, que os conceitos fundamentais sobre modelos de difusão e o processo aqui descrito para imagens se aplicam de forma similar às demais modalidades, diferindo apenas na representação dos dados, nas arquiteturas de redes neurais utilizadas para estimar ruído, e nos mecanismos de condicionamento, todos adaptados para cada domínio específico.

Ideia-chave 2: A obtenção da distribuição de probabilidades verdadeira (*true probability distribution*) p_{dados} para um conjunto de imagens não pode ser obtida a partir de número finito de amostras. Isto ocorre porque: (i) tipicamente, o processo gerador das amostras é desconhecido; (ii) o espaço no qual as amostras encontram-se representadas tem altíssima dimensionalidade (para imagens, $z \in R^{H \times W \times C}$); (iii) Uma quantidade muito pequena de amostras (relativamente às dimensões do espaço) está disponível e a maioria das amostras da distribuição verdadeira nunca será observada. Assim, **o processo generativo precisa "aprender" uma aproximação $\hat{p}_{\text{dados}}$ para p_{dados}** a partir das amostras de treinamento.

Um modelo de difusão aprende a distribuição $\hat{p}_{dados}$ iterando um procedimento que realiza adição e remoção de ruído às imagens de treinamento. Em um primeiro estágio, o **processo de difusão** (*forward diffusion process*) **adiciona ruído** Gaussiano com média zero e variância isotrópica unitária às imagens de treinamento x_0 até que estas se convertam em puro ruído x_T caracterizado por uma aproximação a uma distribuição normal padrão $\mathcal{N}(0, I)$. No segundo estágio, o processo de **difusão reversa aprende a estimar e remover o ruído presente nas imagens** ou, equivalentemente, aprende a mover amostras ruidosas nas direções do espaço mais prováveis de acordo com a distribuição $\hat{p}_{dados}$. O processo de **difusão reversa** constitui a **essência do modelo generativo**. Ao aplicar a remoção de ruído aprendida durante o treinamento a outras amostras ruidosas, o modelo reconstrói novas imagens que seguem a estrutura da distribuição $\hat{p}_{\text{dados}}$. O mecanismo que garante que o processo de remoção de ruído conduza amostras ruidosas x_T em direção a amostras $\hat{x}_0 \sim \hat{p}_{\text{dados}}$ é a existência de um campo vetorial chamado **função escore** que será discutido na Seção 2.3.

 DOI: 10.5753/sbc.20175.9 ©2026 SBC

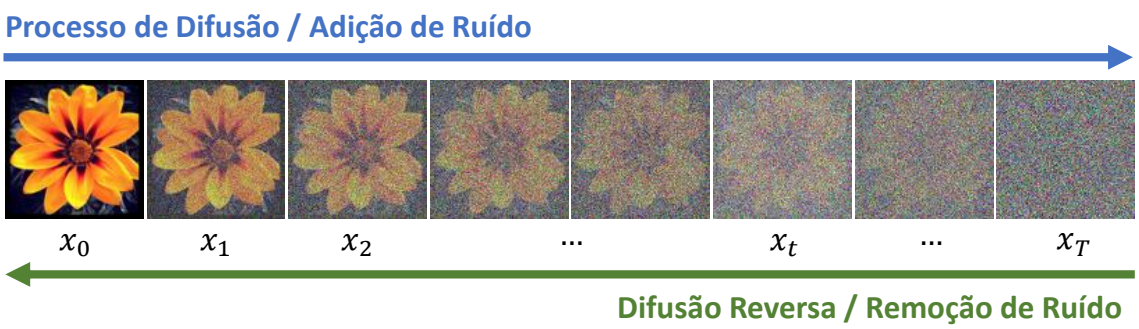

Figura 2.2. Processo de difusão: gradual atenuação do conteúdo e adição de ruído Gaussiano $\varepsilon \sim \mathcal{N}(0,I)$ à imagem de uma flor. Após T passos, a imagem x_0 é transformada em puro ruído $x_T \approx \mathcal{N}(0,I)$. O processo de difusão reversa remove gradualmente o ruído, restaurando a imagem original.

Os processos de adição e de remoção de ruído são ilustrados na Figura 2.2. Nela, durante o processo de difusão a imagem de uma flor (x_0) é progressivamente corrompida pela gradual atenuação de seu conteúdo e adição de ruído Gaussiano $\varepsilon \sim \mathcal{N}(0,I)$ até que, ao final de T passos, a representação resultante tenha se convertido em puro ruído $x_T \approx \mathcal{N}(0,I)$. O processo de difusão reversa remove gradualmente o ruído.

Distribuições Gaussianas estão no centro de modelos de difusão visto que possuem diversas propriedades que garantem que estes sejam matematicamente tratáveis. Elas são completamente caracterizadas utilizando apenas média e covariância, possuem representação analítica, sua amostragem é simples, e possuem função escore definida analiticamente. Outra aspecto fundamental: elas tornam o processo de adição de ruído simples, permitindo a geração de amostras com níveis arbitrários de ruído diretamente a partir de amostras de treinamento. O processo reverso pode ser aprendido de forma gradual e estável. Outro aspecto relevante é o fato do ruído Gaussiano ser isotrópico e suave, não introduzindo viés direcional no espaço. Assim, a distribuição normal padrão $\mathcal{N}(0,I)$ **é utilizada como ponto de partida** x_T para o processo de difusão reversa. Os termos difusão reversa e processo de amostragem são usados como sinônimos ao longo do texto.

Para avançarmos na compreensão do funcionamento de um modelo de difusão para síntese de imagens, vamos agora aplicar o processo de adição gradual de ruído Gaussiano às amostras de um conjunto de treinamento. Seja $\mathscr{D} = \{x_0^{(i)}\}_{i=1}^{N}$ o conjunto de imagens de treinamento, onde N representa o número total de imagens e $x_0^{(i)}$ i-ésima imagem. O processo de adição de ruído é ilustrado na Figura 2.3. À esquerda, há três agrupamentos de pontos azuis, onde cada ponto representa uma imagem (um vetor $x_0^{(i)} \in R^{H\times W\times C}$). Cada agrupamento contém imagens de apenas uma entre três classes distintas: flores, gatos e cães. Coletivamente, essas amostras constituem uma distribuição multimodal. Lembre-se que o objetivo do processo generativo é ser capaz de produzir novas amostras similares às do conjunto de treinamento. Assim, o modelo precisará aprender uma aproximação empírica $\hat{p}_{\text{dados}}$ para p_{dados}, estimada diretamente a partir das amostras observadas.

O processo de difusão aplica uma gradual atenuação do conteúdo original das imagens e adição de ruído Gaussiano $\varepsilon \sim \mathcal{N}(0,I)$ a cada uma das amostras do conjunto de treinamento $\mathscr{D}$ ao longo de T passos. A evolução dessas imagens é representada por

 DOI: 10.5753/sbc.20175.9 ©2026 SBC

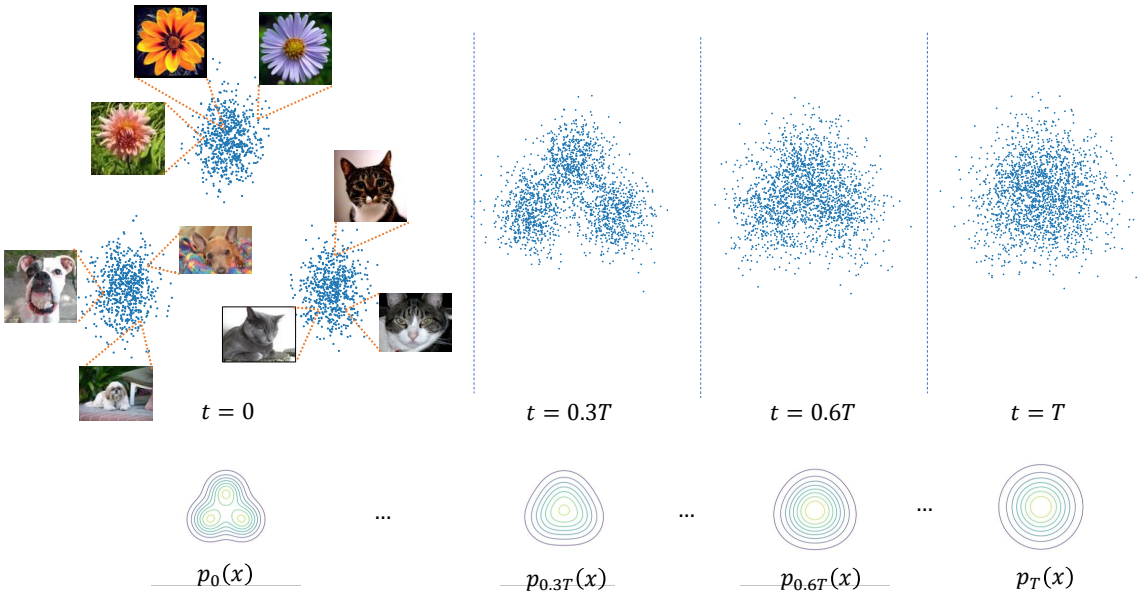

Figura 2.3. Representação abstrata do processo de difusão aplicado a imagens do conjunto de treinamento. As amostras de uma distribuição arbitrária contendo imagens de flores, gatos e cachorros (em $t = 0$) são convertidas em em amostras de uma distribuição Gaussiana $p_T \approx \mathcal{N}(0, I)$ em $t = T$. Isto ocorre por meio da gradual atenuação do conteúdo original das imagens e adição de ruído Gaussiano com média zero e variância isotrópica unitária ao longo de T passos, para T suficientemente grande. Na parte inferior, curvas de nível representam as funções de densidade de probabilidade $p_t(x)$ para diversos valores de t.

$x_t^{(i)}$, com $t \in \{0, 1, 2, ..., T\}$, onde $x_0^{(i)}$ corresponde a uma amostra do conjunto de treinamento (*i.e.*, sem ruído) e $x_T^{(i)}$ corresponde à imagem obtida a partir de $x_0^{(i)}$ após os T passos. Considerando T suficientemente grande, este processo transforma cada imagem $x_0^{(i)}$ em uma imagem ruidosa $x_T^{(i)} \approx \mathcal{N}(0, I)$. Ou seja, o processo de difusão transforma gradualmente amostras da distribuição desconhecida p_{dados} em amostras de uma distribuição que aproxima a distribuição normal padrão $\mathcal{N}(0, I)$. Esta situação é ilustrada na Figura 2.3, onde as amostras da distribuição multimodal $p_0 = p_{dados}$ à esquerda são gradualmente transformadas em amostras de uma distribuição Gaussiana $p_T \approx \mathcal{N}(0, I)$, à direita. Representações também são mostradas para $t = 0,3T$ e $t = 0,6T$.

2.3. A Função Escore

Conforme mencionado, a essência do processo generativo corresponde a mover amostras ruidosas nas direções do espaço mais prováveis de acordo com a distribuição empírica $\hat{p}_{dados}$. Mas como determinar essas direções?

Ideia-chave 3: O log do gradiente da função de densidade de probabilidade em cada uma das etapas do processo no qual t varia de 0 a T fornece um campo vetorial $s_\theta(x_t, t)$ que permite guiar amostras ruidosas em $p_T \approx \mathcal{N}(0, I)$ de volta à $\hat{p}_{\text{data}}$.

A porção inferior da Figura 2.3 mostra as funções de densidade $p_t(x)$ associadas às distribuições de probabilidade p_t para os diversos valores de t ao longo do processo de difusão. Na figura, cada função de densidade de probabilidade é representada por

 DOI: 10.5753/sbc.20175.9 ©2026 SBC

um conjunto de curvas de nível, nas quais os contornos mais internos correspondem a densidades maiores. Assim, o gradiente $\nabla_x p_t(x)$ dessas funções de densidade fornece, para cada distribuição marginal[1] p_t (*i.e.*, para cada estágio do processo de difusão), um campo vetorial que aponta para as regiões de maior concentração de amostras no estágio t. Tomados coletivamente de $t = T$ até $t = 0$, esses campos vetoriais definiriam o caminho para guiar as amostras $p_T(x) \approx \mathcal{N}(0, I)$ da distribuição Gaussiana terminal de volta para as regiões de maior densidade, e portanto mais prováveis, da distribuição $\hat{p}_{\text{dados}}$.

Entretanto, em espaços de alta dimensionalidade os valores de $p_t(x)$ tendem a ser extremamente pequenos em quase todo o espaço. Com isso, $\nabla_x p_t(x)$ resulta em valores muitíssimo pequenos, especialmente em regiões onde a densidade de dados é baixa. Nessas condições, as magnitudes desses campos vetoriais tendem a zero, fazendo com que o sinal disponível para guiar o processo reverso seja muito fraco, tornando o processo generativo (amostragem de $\hat{p}_{\text{data}}$) instável e inefetivo. Este problema é resolvido normalizando o gradiente pelas respectivas densidades:

$$\nabla_x \log p(x) = \frac{\nabla_x p(x)}{p(x)}. \tag{1}$$

A Equação 1 define a **função escore** (*score function*) **responsável por guiar as amostras** x_T **para** $\hat{x}_0$, que caracteriza o aspecto generativo dos modelos de difusão. Ao normalizar os gradientes pelas respectivas densidades, esta função preserva a informação direcional do campo mesmo onde as probabilidades assumem valores muito pequenos. A Figura 2.4 ilustra o uso da função escore para obtenção de um campo vetorial. A imagem à esquerda mostra uma função de densidade de probabilidade $p(x)$ representada por curvas de nível, onde os tons de vermelho e laranja correspondem a regiões de maior densidade. A imagem à direita exibe o campo vetorial definido pela função escore para $p(x)$. Localmente, o campo aponta em direção às maiores densidades.

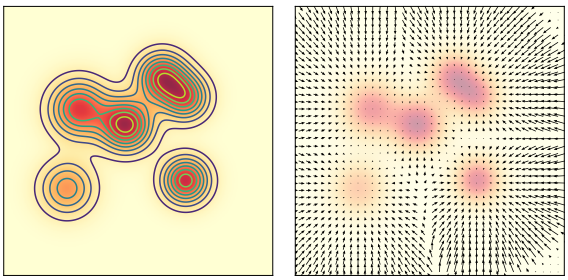

Figura 2.4. Uma função de densidade de probabilidade $p(x)$ representada utilizando curvas de nível. Os tons de vermelho e laranja correspondem às regiões de maior densidade (esquerda). Campo vetorial obtido aplicando-se a função escore a $p(x)$. Localmente, os vetores apontam em direção às maiores densidades.

2.3.1. Gerando Imagens a Partir da Função Escore

Utilizando a função escore, o processo de síntese de imagens se torna intuitivo, bastando para isso gerar uma amostra ruidosa $x_T \sim p_T$ e aplicar a ela o processo de difusão reversa. Como $p_T \approx \mathcal{N}(0, I)$, obtemos x_T amostrando diretamente a distribuição normal padrão:

[1] Cada distribuição individual p_t é chamada de uma *distribuição marginal*.

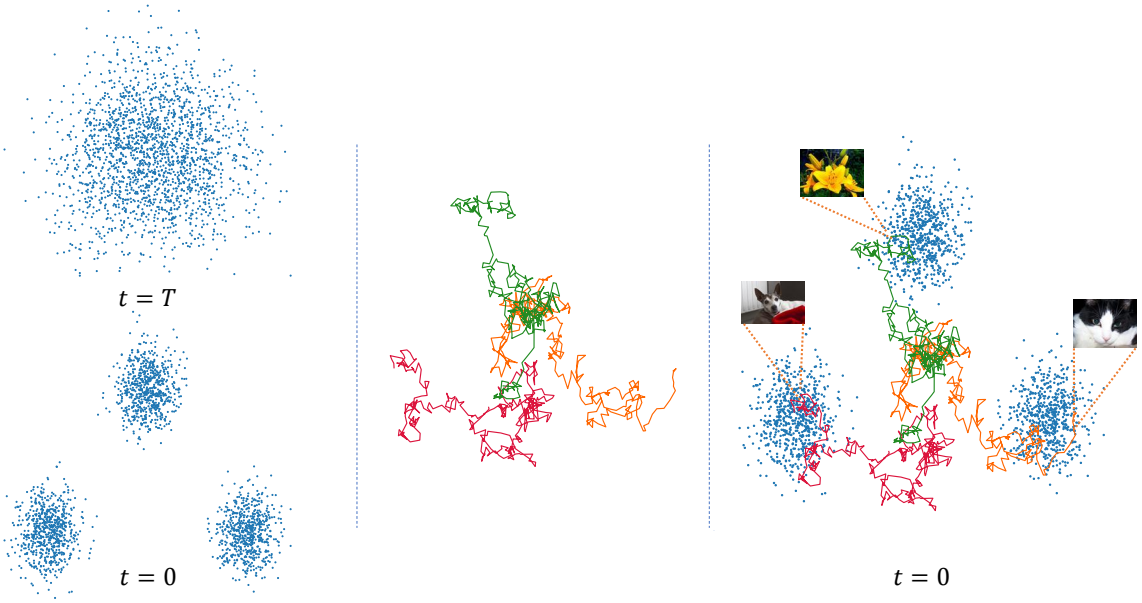

Figura 2.5. Processo de difusão reversa utilizado pela técnica DDPM. (esquerda) Nuvens de pontos representando a distribuição de imagens $p_0 = \hat{p}_{dados}$ no tempos $t = 0$ e e p_T no tempo $t = T$. (centro) trajetórias para três amostras x_t partindo de $x_T \sim \mathcal{N}(0, I)$ até $\hat{x}_0$. (direita) Trajetórias superpostas à nuvem representando $\hat{p}_{dados}$ e as imagens reconstruídas no processo.

$x_T \sim \mathcal{N}(0, I)$. Este processo é ilustrado na Figura 2.5. À esquerda, tem-se as nuvens de pontos que representam as amostras utilizadas no exemplo da Figura 2.3 posicionadas nos instantes $t = 0$ (abaixo) e $t = T$ (acima). Ao centro, vemos as trajetórias de três amostras x_t partindo de $x_T \sim \mathcal{N}(0, I)$ até $\hat{x}_0$. À direita, tem-se as trajetórias superpostas às nuvens de pontos representando $\hat{p}_{dados}$ e as respectivas imagens reconstruídas no processo.

2.3.2. Modelagem Generativa Baseada em Modelos de Difusão

As ideias fundamentais associadas a modelagem generativa baseada em difusão envolvendo corromper progressivamente os dados de treinamento pela adição de ruído Gaussiano e aprender um processo estocástico reverso para geração de novas amostras, descritas na Seção 2.2, foram introduzidas por Sohl-Dickstein e colaboradores [Sohl-Dickstein et al. 2015]. No artigo, os autores não utilizaram a função escore para guiar o processo reverso, tendo utilizado uma rede neural para reverter cada passo do processo de difusão.

Ho e colaboradores [Ho et al. 2020] introduziram diversas melhorias à técnica descrita em [Sohl-Dickstein et al. 2015], tornando modelos de difusão mais fáceis de treinar e melhorando significativamente a qualidade das imagens geradas. A técnica de Ho também não utiliza explicitamente a função escore. Ela treina uma rede neural para predizer o ruído Gaussiano ε acrescido a cada amostra x_t. Uma interpretação explícita do processo de difusão reversa baseado na função score e sua conexão com modelos de difusão tornou-se clara através dos trabalhos de Song e colaboradores [Song and Ermon 2019; Song et al. 2020]. A técnica introduza por Ho e colaboradores, conhecida como **DDPM** (acrônimo para *Denoising Diffusion Probabilistic Models*) inaugurou o que se convencionou chamar de sistemas de difusão modernos.

2.4. Abordagem Probabilística para Modelos de Difusão

Esta seção detalha a técnica DDPM, descrevendo o algoritmo utilizado para treinamento da rede neural utilizada para estimar o ruído em cada amostra x_t, bem como o algoritmo que implementa o processo de difusão reversa propriamente dito.

Seja $x_0 \to x_1 \to \cdots \to x_T$ uma sequência de amostras obtida durante a etapa de adição gradual de ruído Gaussiano. Este processo é definido pela distribuição Gaussiana condicional $q(x_t \mid x_{t-1}) = \mathcal{N}\left(x_t; \sqrt{1-\beta_t}\, x_{t-1}, \beta_t I\right)$. Neste caso, x_{t-1} e x_t pertencem à mesma trajetória de difusão, sendo x_t obtido a partir de x_{t-1} por meio da expressão

$$x_t = \sqrt{1-\beta_t}\, x_{t-1} + \sqrt{\beta_t}\, \varepsilon, \qquad \varepsilon \sim \mathcal{N}(0, I), \tag{2}$$

onde β_t é a variância do ruído Gaussiano acrescentado na etapa t, tal que $\beta_t \in (0,1)$ e $\beta_1 < \beta_2 < \cdots < \beta_T$. A sequência de valores de β_t, chamada de *schedule* de variância (ou *schedule* de ruído), é um parâmetro da técnica DDPM. De acordo com a Equação 2, o valor da amostra x_t é obtido atenuando-se o valor de x_{t-1} e acrescentando-se ruído Gaussiano. Assim, à medida que o valor de β_t se aproxima de 1, x_t se aproxima de ruído Gaussiano com média zero e variância unitária. Por pertencerem à mesma trajetória de difusão, poderíamos escrever $x_t^{(i)} = \sqrt{1-\beta_t}\, x_{t-1}^{(i)} + \sqrt{\beta_t}\, \varepsilon$ para explicitar que se trata de estágios consecutivos da transformação da amostra $x_0^{(i)}$ do conjunto de treinamento.

Fazendo $\alpha_t = 1 - \beta_t$, a Equação 2 pode ser reescrita como

$$x_t = \sqrt{\alpha_t}\, x_{t-1} + \sqrt{1-\alpha_t}\, \varepsilon, \qquad \varepsilon \sim \mathcal{N}(0, I). \tag{3}$$

Definindo $\bar{\alpha}_t = \prod_{i=1}^{t} \alpha_i$ e $\bar{\beta}_t = 1 - \bar{\alpha}_t$, pode-se expressar o valor de x_t diretamente a partir de x_0, conforme a Equação 4:

$$x_t = \sqrt{\bar{\alpha}_t}\, x_0 + \sqrt{1-\bar{\alpha}_t}\, \varepsilon, \qquad \varepsilon \sim \mathcal{N}(0, I). \tag{4}$$

A possibilidade de obter amostras x_t com níveis arbitrários de ruído diretamente a partir das imagens x_0 simplifica e acelera o processo de treinamento da rede neural ε_θ utilizada pela técnica DDPM para estimar a quantidade de ruído presente em x_t. Este treinamento é descrito no Algoritmo 1. Ele itera sobre imagens x_0 do conjunto de treinamento, e amostra uniformemente um nível de ruído t. Uma amostra ruidosa x_t é então obtida de acordo com a Equação 4 e passada juntamente com o valor de t para a rede neural que estima $\varepsilon_\theta(x_t, t)$, o ruído presente em x_t. Observe que o treinamento de ε_θ é realizado gerando-se amostras x_t para valores de t definidos estocasticamente, não requerendo percorrer os valores de t na sequência de 0 a T, como representado na etapa de difusão ilustrada na Figura 2.2.

A função de perda L utilizada pelo Algoritmo 1 calcula o erro quadrático médio entre a quantidade de ruído ε adicionada a x_t e seu valor estimado $\varepsilon_\theta(x_t, t)$ para as várias imagens em um lote. Observem que tanto ε quanto $\varepsilon_\theta(x_t, t)$ são tensores definidos no espaço $R^{H \times W \times C}$, ao passo que L é um escalar. Assim, o valor de L utilizado para atualização dos pesos da rede neural ε_θ é obtido em dois passos: para cada amostra ruidosa x_t em um lote (*minibatch*), calcula-se o erro quadrático médio (*MSE*) entre os tensores ε e $\varepsilon_\theta(x_t, t)$. O valor de L é obtido como a média destes erros para todas as amostras do lote. O mecanismo de retropropagação (*backpropagation*) então computa os gradientes da função L com relação aos parâmetros θ da rede neural e um otimizador (*e.g.*, SGD

 DOI: 10.5753/sbc.20175.9 ©2026 SBC

ou Adam) realiza um passo de atualização dos pesos da rede utilizando o algoritmo de descida do gradiente (linha 8 do Algoritmo 1). Este processo é repetido para diversas amostras x_0 e valores de t até que o valor de L e a qualidade das imagens geradas (considerando aspectos como qualidade, coerência e diversidade) se estabilizem. Observem que o Algoritmo 1 apenas otimiza a predição de ruído e não gera imagens diretamente. Para avaliar a qualidade visual das imagens geradas durante a fase de treinamento, os pesos do modelo são salvos periodicamente e utilizados para executar o procedimento de amostragem (Algoritmo 2) que gera imagens a partir de $x_T \sim \mathcal{N}(0,I)$: $x_T \to x_{T-1} \to \cdots \to \hat{x}_0$. As imagens geradas são então inspecionadas visualmente ou avaliadas utilizando métricas como FID (*Fréchet inception distance*) [Heusel et al. 2017]. FID é computada entre as imagens geradas $\hat{x}_0$ e imagens x_0 do conjunto de treinamento.

Algoritmo 1: Treinamento da rede neural ε_θ utilizada por DDPM estimar ruído em x_t.

1: **repeat**
2: $x_0 \sim p_{\text{data}}(x)$ // Amostra uma imagem do conjunto de treinamento
3: $t \sim \text{Uniform}(\{1,\ldots,T\})$ // Amostra um valor de passo t uniformemente
4: $\varepsilon \sim \mathcal{N}(0,I)$ // Gera ruído Gaussiano
5: $x_t = \sqrt{\bar{\alpha}_t}x_0 + \sqrt{1-\bar{\alpha}_t}\varepsilon$ // Gera amostra ruidosa a partir de x_0
6: Calcula função de perda e executa um passo de descida do gradiente
7: $L = \|\varepsilon - \varepsilon_\theta(x_t,t)\|^2$ // Avalia a função de perda (loss function)
8: $\theta \leftarrow \theta - \eta\nabla_\theta L$ // Atualiza os pesos da rede ε_θ utilizando descida do gradiente
9: **until** convergência

Algoritmo 2: Procedimento de amostragem utilizado pela técnica DDPM.

1: $x_T \sim \mathcal{N}(0,I)$ // Inicializa $x_T \sim \mathcal{N}(0,I)$ a partir de ruído Gaussiano
2: **for** $t = T,\ldots,1$ **do** // Percorre sequência reversa com t variando de T até 1
3: **If** $t > 1$ **then** $z \sim \mathcal{N}(0,I)$ **else** $z = 0$ // Ruído estocástico a ser reinjetado em x_{t-1}
4: $\hat{\varepsilon} = \varepsilon_\theta(x_t,t)$ // Estima o ruído presente em x_t usando a rede neural ε_θ
5: $\mu_\theta(x_t,t) = \frac{1}{\sqrt{\alpha_t}}\left(x_t - \frac{1-\alpha_t}{\sqrt{1-\bar{\alpha}_t}}\hat{\varepsilon}\right)$ // Estimativa mais provável de x_{t-1} que gerou x_t
6: $x_{t-1} = \mu_\theta(x_t,t) + \sigma_t z$ // Estima x_{t-1} com $\sigma_t < \sigma_{t+1}$
7: **end for**
8: **return** x_0 // Retorna a imagem gerada

O Algoritmo 2 detalha o processo de amostragem utilizado pela técnica DDPM e que efetivamente realiza a modelagem generativa. Partindo-se de uma amostra x_T obtida a partir de uma distribuição normal padrão (linha 1), o processo de remoção de ruído é aplicado à toda a sequência $x_T \to x_{T-1} \to \cdots \to \hat{x}_0$ (processo reverso mostrado na Figura 2.2). A rede neural treinada pelo Algoritmo 1 é utilizada para estimar a quantidade total de ruído presente na amostra x_t: $\varepsilon_\theta(x_t,t)$ (linha 4). Este ruído então é utilizado para obter $\mu_\theta(x_t,t)$ (linha 5), que corresponde à estimativa mais provável para o valor de x_{t-1} que teria produzido a amostra x_t após um passo de adição de ruído. O novo valor de x_{t-1} é então atualizado pela injeção de ruído Gaussiano com desvio padrão σ_t (linha 6), onde $\sigma_t < \sigma_{t+1}$. Esta adição de ruído permite a geração de resultados mais diversificados, ao invés de convergir deterministicamente de um dado valor de x_T para uma imagem $\hat{x}_0$.

2.4.1. A Rede Neural Utilizada por DDPM Aproxima a Função Escore

A obtenção de x_{t-1} a partir de x_t por meio de $\mu_\theta(x_t,t)$ segue o campo vetorial definido pela função escore. Podemos verificar este fato observando que dada uma função Gaussiana multivariada normalizada com média μ e variância σ^2

$$p(x) = \frac{1}{(2\pi\sigma^2)^{d/2}} \exp\left(-\frac{1}{2\sigma^2}(x-\mu)^T(x-\mu)\right),$$

sua função escore corresponde a

$$\nabla_x \log p(x) = -\frac{1}{\sigma^2}(x-\mu).$$

De acordo com a Equação 4, a etapa de difusão implementada pela técnica de DDPM define a distribuição condicional

$$p(x_t \mid x_0) = \mathcal{N}\left(x_t; \sqrt{\bar{\alpha}_t}\, x_0, (1-\bar{\alpha}_t)I\right),$$

para a qual a função escore com relação a x_t é dada por

$$\nabla_{x_t} \log p(x_t \mid x_0) = -\frac{x_t - \sqrt{\bar{\alpha}_t}\, x_0}{1-\bar{\alpha}_t}. \tag{5}$$

Novamente, de acordo com a Equação 4, o valor de x_0 pode ser aproximado utilizando o ruído estimado por $\varepsilon_\theta(x_t,t)$ como

$$x_0 \approx \frac{x_t - \sqrt{1-\bar{\alpha}_t}\,\varepsilon_\theta(x_t,t)}{\sqrt{\bar{\alpha}_t}}. \tag{6}$$

Substituindo a Equação 6 na Equação 5, obtem-se:

$$\nabla_{x_t} \log p(x_t \mid x_0) \approx -\frac{x_t - \left(x_t - \sqrt{1-\bar{\alpha}_t}\,\varepsilon_\theta(x_t,t)\right)}{1-\bar{\alpha}_t} = -\frac{\sqrt{1-\bar{\alpha}_t}\,\varepsilon_\theta(x_t,t)}{1-\bar{\alpha}_t} = -\frac{\varepsilon_\theta(x_t,t)}{\sqrt{1-\bar{\alpha}_t}}. \tag{7}$$

A Equação 7 mostra que a **função escore** com relação a x_t para a distribuição condicional $p(x_t \mid x_0)$ utilizada no processo de difusão da técnica DDPM é **proporcional ao ruído $\varepsilon_\theta(x_t,t)$ estimado pela rede neural**. Este fato garante a convergência e a estabilidade da técnica de DDPM. A linha 5 do Algoritmo 2 obtém $\mu_\theta(x_t,t)$, a estimativa mais provável para x_{t-1}, somando a x_t uma versão escalada da função escore e dividindo o resultado por $\sqrt{\alpha_t}$. Esta divisão compensa o fato de que x_t é obtido multiplicando x_{t-1} por $\sqrt{\alpha_t}$ durante a etapa de adição de ruído (Equação 3).

As arquiteturas de redes neurais mais comumente utilizadas na implementação modelos de difusão são baseadas na arquitetura U-Net [Ronneberger et al. 2015] e mais recentemente em Diffusion Transformers (DiT) [Peebles and Xie 2023]. Para o processo de amostragem descrito no Algoritmo 2, é necessário percorrer toda a sequência de T passos $x_T \to x_{T-1} \to \cdots \to \hat{x}_0$. A técnica clássica de DDPM utiliza $T = 1000$ como valor padrão para geração de imagens de alta qualidade. Isto torna o processo de amostragem relativamente lento, visto que a rede neural precisa ser avaliada a cada passo. A Sessão 2.5 e a discussão sobre *difusão latente* na Seção 2.6.3 apresentam estratégias para acelerar o processo de amostragem. Naturalmente, fatores como resolução das imagens, tamanho do modelo e hardware utilizado também afetam o tempo de processamento.

2.5. Abordagem Implícita para Modelos de Difusão

Song e colaboradores [Song et al. 2021] observaram que um modelo de difusão treinado ao estilo DDPM pode ser amostrado utilizando um **processo reverso determinístico** produzindo imagens de alta qualidade. Com base nesta observação, propuseram a técnica **DDIM** (acrônimo para *Denoising Diffusion Implicit Models*) que reformula a etapa de difusão reversa, reduzindo significativamente o número de passos necessários para gerar uma imagem. A **ideia chave da técnica DDIM** é estimar diretamente o valor de $\hat{x}_0$ a partir de qualquer amostra x_t, isolando x_0 na Equação 4:

$$\hat{x}_0 = \frac{x_t - \sqrt{1-\bar{\alpha}_t}\,\varepsilon_\theta(x_t,t)}{\sqrt{\bar{\alpha}_t}}, \tag{8}$$

e utilizá-lo para gerar novas amostras ruidosas para qualquer passo $t-k$:

$$x_{t-k} = \sqrt{\bar{\alpha}_{t-k}}\,\hat{x}_0 + \sqrt{1-\bar{\alpha}_{t-k}}\,\varepsilon_\theta(x_t,t). \tag{9}$$

A Equação 9 pode ser interpretada geometricamente como definindo a posição da nova amostra ruidosa x_{t-k} no espaço das transformações. Lembre-se que $\varepsilon_\theta(x_t,t)$ aproxima o campo vetorial reverso definido pela função escore para a distribuição marginal p_t em x_t, e que em DDPM/DDIM α_t, β_t e $\bar{\alpha}_t$ são valores escalares. Assim, x_{t-k} é obtida "movendo-se" uma versão escalada de $\hat{x}_0$ na direção da amostra x_t.

O processo de treinamento da rede neural ε_θ é o mesmo utilizado por DDPM e descrito no Algoritmo 1. Já o processo de amostragem realizado por DDIM consiste em inicializar $x_T \sim \mathcal{N}(0,I)$ e iterar as Equações 8 e 9 utilizando passos de tamanho k. Neste caso, DDIM percorre a sequência $x_T \to x_{T-k} \to x_{T-2k} \to \cdots \to \hat{x}_0$, utilizando um número consideravelmente menor de passos para obter uma amostra $\hat{x}_0$. O uso de 50 passos de difusão reversa (*e.g.*, $T = 1000$ e $k = 20$) tende a produzir resultados de ótima qualidade. Entretanto, amostradores modernos tendem a utilizar uma abordagem adaptativa, com passos menores ao final do processo, possibilitando um melhor refinamento dos detalhes das imagens. O processo de amostragem descrito é determinístico, significando que para um mesmo valor de x_T e mesma rede ε_θ, DDIM produzirá sempre a mesma imagem $\hat{x}_0$. Esta situação pode ser contornada acrescentando-se um termo estocástico à Equação 9:

$$x_{t-k} = \sqrt{\bar{\alpha}_{t-k}}\,\hat{x}_0 + \sqrt{1-\bar{\alpha}_{t-k}}\,\varepsilon_\theta(x_t,t) + \sigma_t z, \quad z \sim \mathcal{N}(0,I), \tag{10}$$

onde σ_t representa o desvio padrão do ruído Gaussiano acrescentado. Quando $\sigma_t = 0$, tem-se o caso determinístico; valores de $\sigma_t > 0$ introduzem randomicidade ao processo de amostragem. O Algoritmo 3 descreve o processo de amostragem utilizado por DDIM.

Algoritmo 3: Procedimento de amostragem utilizado pela técnica DDIM.

Require: σ_t

1: $x_T \sim \mathcal{N}(0,I)$ // Inicializa $x_T \sim \mathcal{N}(0,I)$ a partir de ruído Gaussiano
2: **for** $t = T,\ldots,1$ **step** k **do**
3: $\quad\hat{\varepsilon} = \varepsilon_\theta(x_t,t)$ // Estima o ruído presente em x_t usando a rede neural ε_θ
4: $\quad\hat{x}_0 = \frac{x_t - \sqrt{1-\bar{\alpha}_t}\,\hat{\varepsilon}}{\sqrt{\bar{\alpha}_t}}$ // Obtém a estimativa da imagem limpa $\hat{x}_0$
5: $\quad z \sim \mathcal{N}(0,I)$ // Ruído Gaussiano para versão estocástica, caso $\sigma_t > 0$
6: $\quad x_{t-k} = \sqrt{\bar{\alpha}_{t-k}}\,\hat{x}_0 + \sqrt{1-\bar{\alpha}_{t-k}}\,\hat{\varepsilon} + \sigma_t z$ // Executa etapa de difusão reversa
7: **end for**
8: **return** $\hat{x}_0$ // Retorna a imagem gerada

 DOI: 10.5753/sbc.20175.9 ©2026 SBC

2.6. Condicionando Modelos de Difusão

Os procedimentos descritos até aqui geram imagens considerando apenas a distribuição de dados aprendida $\hat{p}_{\text{dados}}$, sem um mecanismo que permita direcioná-las para geração de conteúdo específico. O exemplo da Figura 2.5 (direita) ilustra esta situação. Nele, os três valores sementes x_T geraram uma imagem de flor, uma de gato e outra de cachorro. Entretanto, essas imagens foram geradas de modo automático sem escolha do usuário.

É possível direcionar os resultados produzidos por modelos de difusão utilizando instruções para gerar conteúdo com características desejadas. Essas instruções ou comandos, chamados de *prompts*, podem incluir textos, imagens, áudios, ou uma combinação destes (*prompts* multimodais). A Figura 2.1 ilustra a geração de uma imagem utilizando o comando textual "*Um ursinho andando de bicicleta em um parque, usando óculos escuros, um boné azul e uma camisa com a palavra INF*". Note que, a menos que o modelo de difusão utilize um processo de amostragem determinístico, seja inicializado com o mesmo valor inicial de x_T e utilize os mesmos parâmetros da rede neural ε_θ, um mesmo *prompt* gerará imagens diferentes a cada execução.

2.6.1. Gerando Imagens Guiadas por Descrições Textuais

O processo de geração de imagens guiadas por descrições é conceitualmente bastante semelhante ao apresentado nas seções anteriores. Essencialmente, envolve apenas a substituição da função de distribuição de probabilidades $p(x)$ por **uma função de distribuição de probabilidade condicional** $p(x|c)$, onde c é a informação condicionante. No caso de geração de imagens a partir de descrições textuais, c corresponde a um *prompt* textual. Os processos de difusão e de difusão reversa são, consequentemente, semelhantes aos definidos para DDPM e DDIM nas Seções 2.4 e 2.5.

Assim como nos casos clássicos envolvendo DDPM e DDIM, a obtenção de um modelo de difusão condicional consiste em treinar uma rede neural para estimar o ruído $\varepsilon_\theta(x_t, t, c)$ adicionado à amostra ruidosa x_t. A etapa de difusão reversa segue então os campos vetoriais definidos pelas funções escore condicionais $s_\theta(x_t, t, c) \approx \nabla_{x_t} \log p_t(x_t \mid c)$ aprendidas por ε_θ, ao invés de $s_\theta(x_t, t) \approx \nabla_{x_t} \log p_t(x_t)$. Os procedimentos para treinamento e amostragem do modelo estão descritos nos Algoritmos 4 e 5, respectivamente.

Algoritmo 4: Treinamento da rede neural $\varepsilon_\theta(x_t, t, c)$ que estima ruído em x_t para um Modelo de Difusão Condicionado por Texto.

repeat
  $(x_0, c) \sim p_{\text{data}}(x, c)$ // Amostra uma imagem de treinamento x_0 e sua descrição c
  $t \sim \text{Uniform}(\{1, \ldots, T\})$ // Amostra um valor de passo t uniformemente
  $\varepsilon \sim \mathcal{N}(0, I)$ // Gera ruído Gaussiano
  $x_t = \sqrt{\bar{\alpha}_t}\, x_0 + \sqrt{1 - \bar{\alpha}_t}\, \varepsilon$ // Gera amostra ruidosa a partir de x_0
  Calcula função de perda e executa um passo de descida do gradiente
  $L = \|\varepsilon - \varepsilon_\theta(x_t, t, c)\|^2$ //Avalia a função de perda (loss function)
  $\theta \leftarrow \theta - \eta \nabla_\theta L$ // Atualiza os pesos da rede ε_θ utilizando descida do gradiente
until convergência

 DOI: 10.5753/sbc.20175.9 ©2026 SBC

Algoritmo 5: Amostragem no Estilo DDIM Condicionada por Texto.
Require: Texto c contendo descrição para a imagem e σ_t
$x_T \sim \mathcal{N}(0,I)$ // Inicializa $x_T \sim \mathcal{N}(0,I)$ a partir de ruído Gaussiano
for $t = T,\ldots,1$ **step** k **do**
  $\hat{\varepsilon} = \varepsilon_\theta(x_t,t,c)$ // Estima o ruído presente em x_t usando a rede neural ε_θ
  $\hat{x}_0 = \frac{x_t - \sqrt{1-\bar{\alpha}_t}\,\hat{\varepsilon}}{\sqrt{\bar{\alpha}_t}}$ // Obtém a estimativa da imagem limpa $\hat{x}_0$
  $z \sim \mathcal{N}(0,I)$ // Ruído Gaussiano para versão estocástica, caso $\sigma_t > 0$
  $x_{t-k} = \sqrt{\bar{\alpha}_{t-k}}\,\hat{x}_0 + \sqrt{1-\bar{\alpha}_{t-k}}\,\hat{\varepsilon} + \sigma_t z$ // Executa etapa de difusão reversa
end for
return $\hat{x}_0$

Lembre-se que a função escore define um campo vetorial para cada uma das T etapas do processo de difusão ou, mais precisamente, para cada uma das T distribuições marginais $p_t(x)$. Assim, é natural nos perguntarmos como c influencia o comportamento da rede neural ε_θ e, consequentemente, dos campos vetoriais $s_\theta(x_t,t,c)$. Para que isso ocorra, o *prompt* textual é convertido por um codificador de texto baseado em transformer [Vaswani et al. 2017] em representações numéricas (*embeddings*) que preservem a semântica do *prompt*. Essas representações são então usadas por camadas de atenção cruzada em ε_θ para guiar a geração de imagens.

O processo de condicionamento via *prompts* consiste em utilizar os *embeddings* do *prompt* para **modificar os mapas de ativação da rede** responsável por estimar o ruído em x_t. De modo sucinto, seja, por exemplo, $F(x_t,t)$ um conjunto de ativações (*features*) extraído por uma camada de uma rede U-Net a partir de uma imagem x_t. Um bloco transformer [Vaswani et al. 2017] na U-Net utiliza um mecanismo de atenção cruzada para obter $Q_F = F(x_t,t)W_Q$, $K_e = eW_K$, $V_e = eW_V$, onde e é o *embedding* do *prompt* c, e W_Q, W_K e W_V são as matrizes de projeção aprendidas que transformam vetores de características nas representações de *query*, *key*, e *value* usadas pelo mecanismo de atenção. Este obtém $A(x_t,t) = \text{softmax}\left(\frac{Q_F K_e^T}{\sqrt{d}}\right)V_e$. O valor A é então combinado a F por meio de uma conexão residual [He et al. 2016]: $F'(x_t,t) = F(x_t,t) + A(x_t,t)$ e $F'(x_t,t)$ é então passado para a camada subsequente da U-Net. Um bloco transformer é tipicamente inserido em múltiplas resoluções, incluindo vários níveis do decodificador, gargalo e codificador da U-Net. Isto permite que os *embeddings* de texto influenciem o processo de estimação de ruído e, portanto, as funções escore, ao longo de toda a rede.

A Figura 2.6 mostra um exemplo de imagem gerada por um modelo de difusão utilizando a descrição "*Um gato sentado em uma cadeira*". Note que a descrição menciona apenas um gato e uma cadeira, ao passo que a imagem reproduz um cenário bastante rico em detalhes. O mecanismo de atenção auxilia o modelo a organizar informações entre diferentes partes da imagem e entre a imagem e a descrição textual. Isto contribui para melhorar a coerência e composição da cena, além dos relacionamentos entre seus elementos. Entretanto, a imagem final resulta de propriedades que emergem da combinação de vários fatores, incluindo o treinamento do modelo de difusão, a estrutura estatística dos dados de treinamento, e o próprio mecanismo de atenção, entre outros.

 DOI: 10.5753/sbc.20175.9 ©2026 SBC

Figura 2.6. Imagem gerada utilizando o sistema Nano Banana Pro [Open AI 2026], utilizando a descrição: "Um gato sentado em uma cadeira".

2.6.2. Classifier-Free Guidance: Melhorando a Aderência das Imagens ao Prompt

Classifier-Free Guidance (CFG) é uma técnica utilizada durante o processo de difusão reversa para aumentar a influência do sinal de condicionamento c. Durante a etapa de treinamento, a rede ε_θ é ocasionalmente (*e.g.* com probabilidade entre 10% e 20%) treinada substituindo-se c por um *prompt* vazio ($c_\emptyset \leftarrow \emptyset$), enquanto a função de perda $L = \|\varepsilon - \varepsilon_\theta(x_t, t, c)\|^2$ permanece inalterada. Ao final do treinamento, a rede ε_θ terá aprendido a estimar tanto $\varepsilon_\theta(x_t, t, c)$ quanto $\varepsilon_\theta(x_t, t, \emptyset)$. Durante o processo de amostragem, o modelo avalia $\varepsilon_\theta(x_t, t, c)$ e $\varepsilon_\theta(x_t, t, \emptyset)$ e computa

$$\hat{\varepsilon} = \varepsilon_\theta(x_t, t, \emptyset) + w(\varepsilon_\theta(x_t, t, c) - \varepsilon_\theta(x_t, t, \emptyset)), \tag{11}$$

com $w \geq 1$ escalando o vetor guia $g = (\varepsilon_\theta(x_t, t, c) - \varepsilon_\theta(x_t, t, \emptyset))$. Ao amplificar a diferença entre as predições de ruído condicionada e não condicionada, CFG melhora a aderência das imagens geradas ao conteúdo do *prompt*.

2.6.3. Sistemas de Difusão Condicionados por Texto

Diversos sistemas foram desenvolvidos para geração de imagens a partir de descrições textuais. **Imagen** [Saharia et al. 2022], desenvolvido pela Google Research, produz imagens com alto grau de realismo, e elevado grau de compreensão e aderência à descrição textual. A principal premissa por trás deste sistema é a de que compreender bem a linguagem é mais importante para a geração de imagens de alta qualidade e para uma forte aderência ao *prompt* do que o tamanho do modelo de difusão (medido em termos do número de parâmetros treináveis da rede que estima o ruído). Assim, o sistema utiliza um codificador de texto T5-XLL com 11,3 bilhões de parâmetros para converter o texto do *prompt* em *embeddings*. Ele gera inicialmente imagens com 64×64 pixels, que são escaladas para 256×256 e posteriormente para 1024×1024 utilizando dois modelos de difusão para super-resolução.

Stable Diffusion [Rombach et al. 2022] realiza o processo de difusão em um espaço latente comprimido, reduzindo significativamente os custos computacional e de

 DOI: 10.5753/sbc.20175.9 ©2026 SBC

memória. A técnica, conhecida como difusão latente (*latent diffusion*), consiste em comprimir as imagens utilizando um Autoencoder Variacional [Kingma and Welling 2014] e realizar o treinamento da rede neural utilizando esta representação comprimida. O treinamento é realizado no estilo DDPM. O processo de geração de imagens consiste então em partir de $z_T \sim \mathcal{N}(0, I)$ para obtenção de uma amostra $\hat{z}_0$, a qual é então mapeada para uma imagem utilizando o decodificador do Autoencoder Variacional. Ao utilizar uma representação comprimida no espaço latente, o modelo reduz tanto o tempo de treinamento quanto de amostragem.

DALL·E 3 [Betker et al. 2023], desenvolvido pela OpenAI, se caracteriza por uma ótima compreensão do *prompt* e uma forte aderência às instruções fornecidas pelo usuário. A premissa por trás do sistema DALL·E 3 é a de que modelos para geração de imagens a partir de texto podem ter seus desempenhos significativamente melhorados se treinados com imagens contendo descrições detalhadas. Assim, um sistema de descrição de imagens foi utilizado para produzir descrições detalhadas para o conjunto de treinamento. DALL·E 3 interpreta e reescreve os *prompts* dos usuários com maior detalhamento antes de gerar uma imagem. Esta funcionalidade é facilitada pela integração do sistema com os modelos de linguagem (LLMs) da OpenAI.

A Figura 2.7 mostra duas imagens geradas pelo sistema Imagen. A Figura 2.8 exibe imagens geradas com Stable Diffusion (esquerda) e DALL·E 3 (direita).

Figura 2.7. Imagens geradas com o sistema Imagen utilizando as descrições: (esquerda) "A forest with big trees and colorful leaves with a chicken next to it". (direita) "Três carros de corrida de fórmula 1".

2.6.4. O Impacto do Idioma Utilizado para Descrição da Imagem

A língua utilizada para a escrita do *prompt* pode ter um impacto significativo na imagem gerada. Assim, se um sistema que gera imagens a partir de comandos textuais foi treinado predominantemente utilizando pares de imagens e descrições em um dado idioma, ele tenderá a interpretar comandos neste idioma de forma mais precisa e a gerar imagens mais alinhadas com tais descrições do que utilizando *prompts* em outras línguas. Além disso, alguns conceitos, referências culturais e expressões idiomáticas podem ser melhor

 DOI: 10.5753/sbc.20175.9 ©2026 SBC

Figura 2.8. Imagens geradas com Stable Diffusion (esquerda) e DALL·E 3 (direita) utilizando a descrição: "Fotografia de um homem em barco nas águas do rio Amazonas ao por do sol, cinemático.".

representadas em uma língua do que em outras. Para sistemas modernos como Imagen, DALL·E 3 e versões recentes de Stable Diffusion, descrições em português do Brasil tendem a produzir ótimos resultados, especialmente para conceitos comuns e descrições de cenas, como nos exemplos das Figuras 2.6 a 2.8. Em determinadas situações, a utilização de *prompts* em inglês pode resultar em maior aderência à descrição, visto que uma parcela considerável dos pares imagem-descrição disponíveis para treinamento de modelos de difusão encontram-se em inglês.

2.6.5. Condicionamento por Imagens e Condicionamentos Multimodais

Modelos de difusão podem utilizar várias modalidades de dados além de texto para guiar o processo de síntese de imagens. A princípio, qualquer modalidade que possa ser codificada na forma de *embeddings* que preservem informações semanticamente relevantes dos dados de entrada pode ser utilizada. Isso inclui, por exemplo, imagens, áudio e representações multimodais, em particular envolvendo texto e imagens.

Utilizando apenas imagem como *prompt*, ILVR (*Iterative Latent Variable Refinement*) [Choi et al. 2021] guia o processo de difusão em direção a uma imagem de referência injetando repetidamente informações de baixa frequência desta imagem durante o processo de difusão inversa. SDEdit [Meng et al. 2022] controla o processo de difusão em direção a uma imagem de referência x_0 acrescentando uma certa quantidade de ruído a esta com o intuito de eliminar os seus detalhes finos, mas preservando sua estrutura geral. A imagem resultante (e apenas parcialmente ruidosa) x_t é então usada em substituição a $x_T \sim \mathcal{N}(0, I)$ como ponto de partida para o processo de difusão reversa utilizado por DDPM/DDIM. ILVR e SDEdit assumem a existência de uma rede neural pre-treinada no estilo DDPM capaz de estimar a quantidade de ruído em uma imagem.

O condicionamento multimodal funciona de modo análogo ao descrito na Seção 2.6.1: as informações das diferentes modalidades são convertidas em *embeddings*, fundidas utilizando mecanismos de atenção cruzada, e reinjetadas nas camadas subsequentes

da rede neural ε_θ, influenciando os campos vetoriais da função escore condicionada. Isto faz com que as trajetórias definidas por estes campos vetoriais conduzam a amostras que tentam satisfazer as informações dos *prompts*. Neste caso, o modelo pode receber simultaneamente $c = (c_{\text{texto}}, c_{\text{imagem}}, c_{\text{áudio}}, \dots)$. Exemplos, incluem a geração de imagens a partir de texto + imagem, texto + áudio, geração de vídeo a partir de texto + imagem, etc. Exemplos recentes de modelos de difusão envolvendo texto + imagem utilizam filtragem para preservar a estrutura da imagem de entrada [Gu et al. 2024; Tamiosso et al. 2025].

2.7. Limitações e Questões Éticas

Modelos de difusão têm sido utilizados com sucesso em aplicações de diversas áreas, incluindo síntese de imagens e vídeos, criação e extração de modelos 3D a partir de imagens, aplicações médicas e design de novos produtos. Mas como todo modelo de aprendizagem de máquina, modelos de difusão apresentam vieses herdados de seus conjuntos de treinamento. Em geral, exigem grandes datasets e seus treinamentos costumam envolver alto custo computacional. Por se basearem em estatísticas de padrões visuais ao invés de em representações explícitas de objetos, modelos de difusão têm dificuldade com contagens e consistências estruturais (*e.g.*, frequentemente gerando uma quantidade de objetos diferente da solicitada, ou mãos com mais ou com menos de cinco dedos). Além dessas, outras limitações atuais são abordadas na Seção 2.8 que discute oportunidades de pesquisa. O impacto do idioma utilizado no treinamento do modelo foi tratado na Seção 2.6.4. Uma discussão de extrema relevância relacionada ao uso dessa tecnologia envolve **aspectos éticos** ligados ao seu elevado potencial para produção de *deep fakes*.

2.8. Oportunidades de Pesquisa

Apesar dos rápidos avanços observados em modelos de difusão, a área ainda apresenta inúmeros desafios, oferecendo diversas oportunidades de pesquisa. Alguns dos temas mais ativos no momento incluem: (i) A busca por **amostradores mais rápidos**, capazes de reduzir o tempo para geração de imagens sem degradar a qualidade dos resultados; (iii) **Condicionamento multimodal**: garantir condicionamento combinando diversas modalidades como texto, imagens, áudio, vídeos, modelos 3D, mapas de profundidade, etc.; (iii) **Geração de vídeos** garantindo consistência temporal, movimentação natural de câmera, produção de vídeos longos, e a redução do custo computacional para sua geração; (iv) **Aspectos físicos do mundo real**: atualmente os modelos aprendem relações entre elementos das cenas com base na regularidade estatística das imagens no conjunto de treinamento. Por exemplo, o gato da Figura 2.6 está em contato com a cadeira não devido ao conhecimento sobre a lei da gravidade, mas porque essa relação aparece sistematicamente no conjunto de treinamento; e (v) **Geração de cenas 3D**: ao invés de apenas imagens 2D.

Apesar do enorme sucesso alcançado por modelos de difusão, várias questões teóricas permanecem em aberto. Por exemplo, quais são os limites fundamentais de processos generativos baseados na função escore?

2.9. Conclusão

Este tutorial apresentou uma introdução a modelos de difusão para geração de imagens. Os princípios matemáticos e estatísticos que garantem a correção desta classe de técnicas

 DOI: 10.5753/sbc.20175.9 ©2026 SBC

e a obtenção de imagens de alta qualidade foram discutidos de forma intuitiva, porém tecnicamente rigorosa. Modelos de difusão utilizam a função escore para definir trajetórias que levam amostras consistindo de puro ruído para as regiões mais prováveis em uma distribuição empírica de probabilidades definida pelo conjunto de imagens de treinamento. A função escore, por sua vez, pode ser aproximada por uma rede neural ε_θ que estima o ruído acrescentado a amostras do conjunto de treinamento. O condicionamento de um modelo de difusão por meio de *prompts*, especialmente quando combinado com *classifier-free guidance* permite redefinir os campos vetoriais da função escore de modo a direcionar amostras para regiões do espaço de imagens mais alinhadas com as descrições dos *prompts*. Isto permite aos modelos de difusão gerar imagens de alta qualidade e alinhadas com as instruções dadas pelos usuários. Apesar do rápido avanço experimentado, modelos de difusão apresentam inúmeros desafios e oportunidades de pesquisa.

Créditos das imagens

As imagens nas Figuras 2.2 a 2.5 foram utilizadas de acordo com licença CC. A foto na Figura 2.2 (esquerda) é de mosher13; As fotos na Figura 2.3 são de - flores: mosher13, iGlacier e MadalenaPestana; gatos: nist6ss, Jon_a_ross e Mike_Knell; cães: basykes, cfiesler e twodolla. As fotos na Figura 2.5 são de - flor: FotoGuy$_4$9057; $gato : El\_C$; $cão : eXploration\_Etoile$.

Referências

[Abramson et al. (2024)] Josh Abramson, Jonas Adler, et al. Accurate structure prediction of biomolecular interactions with alphafold 3. *Nature*, 630(8016):493–500, 2024.

[AI (2026)] Deevid AI. Deevid. AI video generator, 2026. URL `https://deevid.ai/`. Last Access, May 2026.

[Alibaba (2026)] Alibaba. Wan video. AI video generator, 2026. URL `https://wan.video/`. Last Access, May 2026.

[Betker et al. (2023)] James Betker, Gabriel Goh, Li Jing, Tim Brooks, Jianfeng Wang, Linjie Li, Long Ouyang, Juntang Zhuang, Joyce Lee, Yufei Guo, et al. Improving image generation with better captions. *Computer Science. https://cdn.openai.com/papers/dall-e-3.pdf*, 2(3):8, 2023.

[Black Forest (2026)] Labs Black Forest. Flux. AI image generator, 2026. URL `https://www.fluxpro.ai`. Last Access, May 2026.

[Choi et al. (2021)] Jooyoung Choi, Sungwon Kim, Yonghyun Jeong, Youngjune Gwon, and Sungroh Yoon. Ilvr: Conditioning method for denoising diffusion probabilistic models. In *Proceedings of ICCV*, pages 14367–14376, 2021.

[Gu et al. (2024)] Zeqi Gu, Ethan Yang, and Abe Davis. Filter-guided diffusion for controllable image generation. In *ACM SIGGRAPH 2024 conference papers*, pages 1–10, 2024.

[He et al. (2016)] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proc. of CVPR*, pages 770–778, 2016.

[Heusel et al. (2017)] Martin Heusel, Hubert Ramsauer, Thomas Unterthiner, Bernhard Nessler, and Sepp Hochreiter. Gans trained by a two time-scale update rule converge to a local nash equilibrium. *NeurIPS*, 30, 2017.

[Ho et al. (2020)] Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. *NeurIPS*, 33:6840–6851, 2020.

[Kingma and Welling (2014)] Diederik P. Kingma and Max Welling. Auto-Encoding Variational Bayes. In *2nd Intern. Conf. on Learning Representations, ICLR 2014*, 2014.

[Kuaishou (2026)] Technology Kuaishou. Klingai. AI video generator, 2026. URL `https://kling.ai/`. Last Access, May 2026.

[Labs (2026)] Luma Labs. Luma dream machine. AI video generator, 2026. URL `https://lumalabs.ai/`. Last Access, May 2026.

[Luo and Hu (2021)] Shitong Luo and Wei Hu. Diffusion probabilistic models for 3d point cloud generation. In *Proceedings of CVPR*, pages 2837–2845, 2021.

[Meng et al. (2022)] Chenlin Meng, Yutong He, Yang Song, Jiaming Song, Jiajun Wu, Jun-Yan Zhu, and Stefano Ermon. SDEdit: Guided image synthesis and editing with stochastic differential equations. In *International Conference on Learning Representations*, 2022. URL `https://openreview.net/forum?id=aBsCjcPu_tE`.

[Midjourney, Inc. (2026)] Midjourney, Inc. Midjourney. AI image generator, 2026. URL `https://www.midjourney.com`. Last Access, May 2026.

[Open AI (2026)] Open AI. Nano banana pro. AI image generator, 2026. URL `https://chat.chatbotapp.ai/nano-banana`. Last Access, May 2026.

[OpenAI (2026)] OpenAI. Sora 2. AI video generator, 2026. URL `https://openai.com/index/sora-2/`. Last Access, May 2026.

[Peebles and Xie (2023)] William Peebles and Saining Xie. Scalable diffusion models with transformers. In *Proceedings of ICCV*, pages 4195–4205, 2023.

[Rombach et al. (2022)] Robin Rombach, Andreas Blattmann, Dominik Lorenz, Patrick Esser, and Björn Ommer. High-resolution image synthesis with latent diffusion models. In *Proceedings of CVPR*, pages 10684–10695, 2022.

[Ronneberger et al. (2015)] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. U-net: Convolutional networks for biomedical image segmentation. In *Intern. Conf. on Medical image computing and computer-assisted intervention*, pages 234–241, 2015.

[Runway AI (2026)] Inc Runway AI. Runway gen 4.5. AI video generator, 2026. URL `https://runwayml.com/`. Last Access, May 2026.

[Saharia et al. (2022)] Chitwan Saharia, William Chan, et al. Photorealistic text-to-image diffusion models with deep language understanding. *NeurIPS*, 35:36479–36494, 2022.

[Sohl-Dickstein et al. (2015)] Jascha Sohl-Dickstein, Eric Weiss, Niru Maheswaranathan, and Surya Ganguli. Deep unsupervised learning using nonequilibrium thermodynamics. In *International conference on machine learning*, pages 2256–2265. pmlr, 2015.

[Song et al. (2021)] Jiaming Song, Chenlin Meng, and Stefano Ermon. Denoising diffusion implicit models. In *International Conference on Learning Representations*, 2021. URL `https://openreview.net/forum?id=St1giarCHLP`.

[Song and Ermon (2019)] Yang Song and Stefano Ermon. Generative modeling by estimating gradients of the data distribution. *NeurIPS*, 32, 2019.

[Song et al. (2020)] Yang Song, Jascha Sohl-Dickstein, Diederik P Kingma, Abhishek Kumar, Stefano Ermon, and Ben Poole. Score-based generative modeling through stochastic differential equations. *arXiv preprint arXiv:2011.13456*, 2020.

[Tamiosso et al. (2025)] Gustavo Lopes Tamiosso, Caetano Müller, Lucas Spagnolo Bombana, and Manuel M Oliveira. Memory-efficient filter-guided diffusion with domain transform filtering. *Computers & Graphics*, page 104389, 2025.

[Vaswani et al. (2017)] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is All You Need. *NeurIPS*, 30, 2017.

[Wang et al. (2025)] Chen Wang, Hao-Yang Peng, Ying-Tian Liu, Jiatao Gu, and Shi-Min Hu. Diffusion models for 3d generation: A survey. *Computational Visual Media*, 11 (1):1–28, 2025.