Modelagem Generativa com Equações Diferenciais Estocásticas (SDEs): Uma Breve Revisão

Ezequiel Braga
FGV EMAp
Rio de Janeiro, Brasil
ezequiel.braga.santos@gmail.com

Jairon Henrique
FGV EMAp
Rio de Janeiro, Brasil
henriquejairon@gmail.com

Abstract

"Criar ruído a partir de dados é fácil; criar dados a partir de ruído é modelagem generativa". Esse trabalho revisa o artigo *Score-Based Generative Modeling through Stochastic Differential Equations*, que faz o uso de SDEs como uma versão contínua do já famoso DDPM, e usa uma SDE correspondente em tempo reverso que transforma a distribuição conhecida de volta na distribuição de dados ao remover o ruído lentamente. Crucialmente, a SDE em tempo reverso depende apenas do gradiente dependente do tempo (também conhecido como score) do logaritmo da distribuição de dados perturbada. Aproveitando os avanços na modelagem generativa baseada em scores, podemos estimar esses scores com precisão usando redes neurais e utilizar solvers numéricos de SDE para gerar amostras. Foi mostrado que essa estrutura encapsula abordagens anteriores na modelagem generativa baseada em scores e na modelagem probabilística de difusão, permitindo novos procedimentos de amostragem e novas capacidades de modelagem. Esse modelo também é interessante por permitir amostragem condicional com pouco treinamento adicional. Nesse trabalho, fizemos alguns experimentos simples que ilustram o modelo.

1 Introdução

As abordagens mais famosas de modelos generativos probabilísticos baseiam-se na modificação dos dados de treinamento através de um processo de introdução de ruído crescente e, então, aprendendo o processo reverso para formar o modelo generativo dos dados. O modelo de *score matching with Langevin dynamics* (SMLD) estima a função score (gradiente da log-verossimilhança com respeito aos dados) em cada escala de ruído e usa a dinâmica de Langevin para amostrar uma sequência decrescente de ruídos durante a geração [1]. Por outro lado, o modelo de difusão de eliminação de ruído (*Denoising diffusion probabilistic modeling*, DDPM) usa a forma funcional da função score para treinar uma sequência de modelos para reverter cada passo ruidoso [2]. Tais modelos são denominados de *score-based generative models* e permitem uma abordagem efetiva para geração de imagens e aúdios, por exemplo. No entanto, a proposta descrita em [3] consiste em generalizar as abordagens dos modelos citados anteriormente através de equações diferenciais estocásticas (SDEs).

Ao invés de perturbar os dados com distribuições de ruídos, a ideia proposta consiste na utilização de distribuições que evoluem ao longo do tempo através de um processo de difusão: os dados são difundidos em ruídos aleatórios por meio de uma SDE que não depende dos dados e nem possuem parâmetros treináveis. O ponto crucial é que o processo de reversão é descrito por uma SDE com tempo reverso [4], que pode ser obtida dada o score marginal como função do tempo. Para isso, constrói-se uma aproximação da SDE reversa treinando uma rede neural para estimar os *scores* e, a partir disso, obter amostras solucionando a SDE, conforme descrito na imagem (1).

O *score* de uma distribuição com densidade $p_0(\boldsymbol{x})$ é definido como sendo $\nabla_{\boldsymbol{x}} \log p_0(\boldsymbol{x})$. Matematicamente, o objetivo de um modelo generativo é criar novas amostras de uma distribuição p_0, usando como *input* um conjunto finito de amostras. Nesse aspecto, a estratégia de interpolar a função *score*

no lugar da função p_0 é vantajosa pois a função *score* não sofre com problema de normalização e positividade. De fato se p é uma distribuição não normalizada, então

$$\nabla \log(p/Z) = \nabla \log p.$$

Uma vez obtido um bom estimador de $\nabla \log p_0$, podemos usar algum método de Monte Carlo Markov Chain para amostrar dessa distribuição. Por exemplo, podemos usar a dinâmica de Langevin [5] ou Hamilton Monte Carlo (HMC) [6]. Mas nesse trabalho, usaremos o próprio aparato das SDEs para realizar a amostragem.

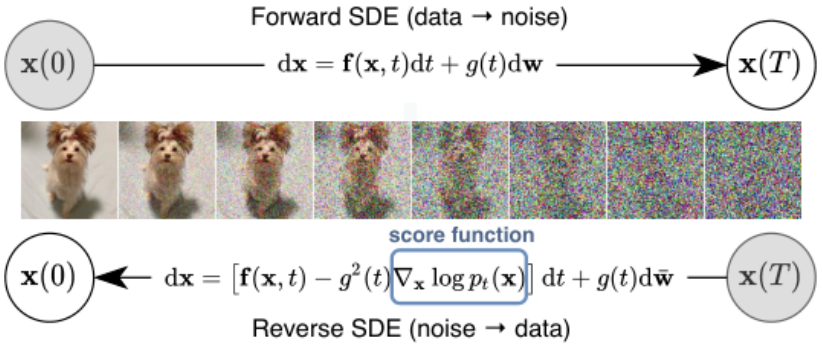


Figura 1: Funcionamento do processo de difusão com a SDE.

Essa abordagem fornece duas contribuições fundamentais:

- Geração controlável: é possível modular o processo de geração condicionando em informações não disponíveis durante o treinamento, uma vez que a SDE condicional de tempo reverso pode ser eficientemente estimada através de *scores* não condicionais.
- Estrutura unificada: essa metodologia unifica a modelagem generativa, já que SMLD e DDPM podem ser vistos como discretizações de SDEs.

2 Modelagem generativa com SDEs

2.1 Perturbando os dados com SDEs

Queremos construir um processo de difusão $\{\boldsymbol{x}(t)\}_{t=0}^{T}$ de modo que $\boldsymbol{x}_0 \sim p_0$ e $\boldsymbol{x}(T) \sim p_T$, onde p_T é uma distribuição a priori (que pode ser qualquer distribuição tratável) e p_0 é a distribuição dos dados. Tal processo pode ser modelado como a solução de Itô da SDE a seguir:

$$d\boldsymbol{x} = f(\boldsymbol{x}, t)dt + g(t)dW,$$

onde W é o movimento Browniano, $f(\cdot, t) : \mathbb{R}^d \mapsto \mathbb{R}^d$ é uma função chamada de coeficiente de *drift* de $\boldsymbol{x}(t)$, e $g(\cdot) : \mathbb{R} \mapsto \mathbb{R}$ é uma função conhecida como coeficiente de difusão de $\boldsymbol{x}(t)$.

Para gerar amostras, iniciando com $\boldsymbol{x}(T) \sim p_T$, é possível obter $\boldsymbol{x}(0) \sim p_0$ revertendo o processo. Um resultado importante é que o processo de reversão é um processo de difusão, $\tilde{\boldsymbol{x}}(t) = \boldsymbol{x}(T - t)$, dado pela SDE de tempo reverso [4]

$$d\tilde{\boldsymbol{x}} = [f(\tilde{\boldsymbol{x}}, t) - g(t)^2 \nabla_{\tilde{\boldsymbol{x}}} \log p_{T-t}(\tilde{\boldsymbol{x}})]dt + g(t)d\bar{W}, \tag{1}$$

onde $\bar{W}$ é o movimento Browniano de T a 0 e p_t é distribuição marginal de $\boldsymbol{x}(t)$. A figura (2) descreve todo esse processo. Outro fato importante é que a distribuição exata p_T é desconhecida, mas escolhemos a SDE de modo a termos uma distribuição invariante conhecida γ (costumeiramente normal, no caso de equações lineares), e usamos que $p_T \approx \gamma$.

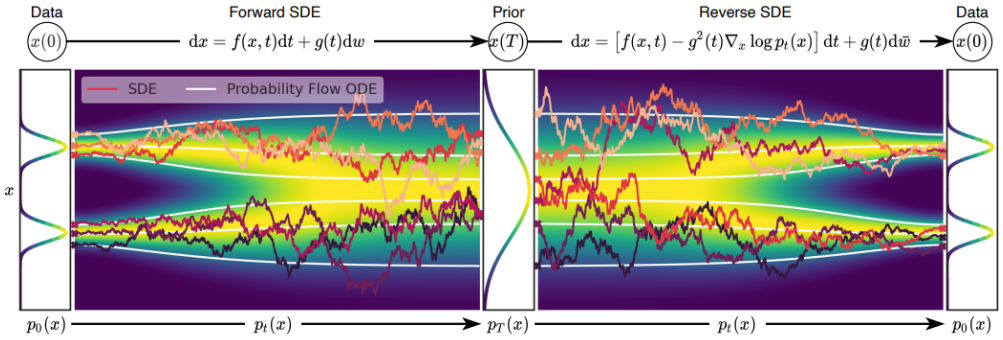


Figura 2: Visão geral da modelagem generativa com SDEs.

2.2 Estimando scores para a SDE

Similar ao que fazemos no caso discreto DDPM (veja apêndice B), a ideia é usar o processo de desconstrução para treinar um modelo enquanto usamos o processo reverso para amostragem. Em virtude da equação (1), para amostrarmos dos dados precisamos ter acesso a função *score*, $\nabla \log p_t(\boldsymbol{x}(t))$, mas essa distribuição é naturalmente inacessível. O que faremos é treinar uma rede neural $s_{\boldsymbol{\theta}}(\boldsymbol{x}(t), t)$ que aproxima essa função. Uma forma natural de escolher os parâmetros da rede seria

$$\boldsymbol{\theta}^\star = \arg\min_{\boldsymbol{\theta}} \mathbb{E}_{t\sim Unif([0,1])}[\lambda(t)\, \mathbb{E}_{\boldsymbol{x}(t)\sim p_t}[||s_{\boldsymbol{\theta}}(\boldsymbol{x}(t),t) - \nabla \log p_t(\boldsymbol{x}(t))||^2]],$$

onde $\lambda(t)$ é qualquer função positiva de pesos. Porém essa perda (aparentemente) é intratável, pois não sabemos avaliar $p_t(\boldsymbol{x}(t))$. Em virtude disso, nos será crucial o seguinte teorema, cuja a demonstração é uma simples integração por partes.

Teorema 1. *Se, para todo $\boldsymbol{\theta}$, $s_{\boldsymbol{\theta}}(\boldsymbol{x})$ e $p_{data}(\boldsymbol{x})$ são diferenciáveis, $\mathbb{E}_{p_{data}(\boldsymbol{x})}[\|s_{\boldsymbol{\theta}}(\boldsymbol{x})\|^2]$ e $\mathbb{E}_{p_{data}(\boldsymbol{x})}[\|\nabla_{\boldsymbol{x}} \log p_{data}(\boldsymbol{x})\|^2]$ são finitos, e*

$$\lim_{\|\boldsymbol{x}\|\to\infty} p_{data}(\boldsymbol{x}) s_{\boldsymbol{\theta}}(\boldsymbol{x}) = 0,$$

então é possível escrever

$$\mathbb{E}_{p_{data}(\boldsymbol{x})}\left[\|s_{\boldsymbol{\theta}}(\boldsymbol{x}) - \nabla_{\boldsymbol{x}} \log p_{data}(\boldsymbol{x})\|^2\right] = \mathbb{E}_{p_{data}(\boldsymbol{x})}\left[\|s_{\boldsymbol{\theta}}(\boldsymbol{x})\|^2 + 2\nabla_{\boldsymbol{x}} \cdot s_{\boldsymbol{\theta}}(\boldsymbol{x})\right] + C,$$

onde C não depende de $\boldsymbol{\theta}$.

Demonstração. Nós temos

$$\mathbb{E}[||s_{\boldsymbol{\theta}}(\boldsymbol{x}) - \nabla \log p_{data}(\boldsymbol{x})||^2] = \mathbb{E}[||s_{\boldsymbol{\theta}}(\boldsymbol{x})||^2] - 2\,\mathbb{E}[\nabla \log p_{data}(\boldsymbol{x}) \cdot s_{\boldsymbol{\theta}}(\boldsymbol{x})] + \mathbb{E}[||\nabla \log p_{data}(\boldsymbol{x})||^2].$$

Note que pela regra da cadeia $p_{data}(\boldsymbol{x})\nabla \log p_{data}(\boldsymbol{x}) = \nabla p_{data}(\boldsymbol{x})$, portanto,

$$\begin{aligned}
\mathbb{E}[\nabla \log p_{data}(\boldsymbol{x}) \cdot s_{\boldsymbol{\theta}}(\boldsymbol{x})] &= \int \nabla \log p_{data}(\boldsymbol{x}) \cdot s_{\boldsymbol{\theta}}(\boldsymbol{x}) p(\boldsymbol{x}) d\boldsymbol{x} \\
&= \int \nabla p_{data}(\boldsymbol{x}) \cdot s_{\boldsymbol{\theta}}(\boldsymbol{x}) d\boldsymbol{x} \\
&= -\int \nabla \cdot s_{\boldsymbol{\theta}}(\boldsymbol{x}) p(\boldsymbol{x}) d\boldsymbol{x} = -\,\mathbb{E}[\nabla \cdot s_{\boldsymbol{\theta}}(\boldsymbol{x})],
\end{aligned}$$

onde na última linha nós integramos por parte e usamos a hipótese do decaimento no termo de fronteira. Tomando $C = \mathbb{E}[||\nabla \log p_{data}(\boldsymbol{x})||^2]$, concluímos o desejado. □

Usando o teorema duas vezes, podemos reescrever a perda de duas formas tratáveis diferentes.

$$\mathbb{E}_t[\lambda(t)\,\mathbb{E}_{\boldsymbol{x}(t)\sim p_t}[||s_\theta(\boldsymbol{x}(t),t)-\nabla\log p_t(\boldsymbol{x}(t))||^2]] \tag{2}$$

$$=\mathbb{E}_t[\lambda(t)\,\mathbb{E}_{\boldsymbol{x}(t)\sim p_t}[||s_\theta(\boldsymbol{x}(t),t)||^2+2\nabla\cdot s_\theta(\boldsymbol{x}(t),t)]]+C \tag{3}$$

$$=\mathbb{E}_t[\lambda(t)\,\mathbb{E}_{\boldsymbol{x}_0\sim p_0}[\mathbb{E}_{\boldsymbol{x}(t)\sim p_{t|0}}[||s_\theta(\boldsymbol{x}(t),t)||^2+2\nabla\cdot s_\theta(\boldsymbol{x}(t),t)]]+C \tag{4}$$

$$=\mathbb{E}_t[\lambda(t)\,\mathbb{E}_{\boldsymbol{x}_0\sim p_0}[\mathbb{E}_{\boldsymbol{x}(t)\sim p_{t|0}}[||s_\theta(\boldsymbol{x}(t),t)-\nabla\log p_{t|0}(\boldsymbol{x}(t)|\boldsymbol{x}_0)||^2]]]+C_1. \tag{5}$$

Os termos dentro da esperança são avaliáveis usando métodos de derivação automática na equação (3), alternativamente usamos a equação (5), e então precisaremos conhecer o score **condicional** para a SDE escolhida - o que é verdade para SDEs lineares. A esperança pode ser aproximada via método de Monte Carlo a partir de soluções numéricas da SDE no tempo foward.

Assim, para estimar $\nabla_{\boldsymbol{x}}\log p_t(\boldsymbol{x})$, treinamos $s_{\boldsymbol{\theta}}(\boldsymbol{x},t)$ com:

$$\boldsymbol{\theta}^\star=\arg\min_{\boldsymbol{\theta}}\mathbb{E}_t\left[\lambda(t)\mathbb{E}_{\boldsymbol{x}(0)}\mathbb{E}_{\boldsymbol{x}(t)|\boldsymbol{x}(0)}\left[\|s_{\boldsymbol{\theta}}(\boldsymbol{x}(t),t)-\nabla_{\boldsymbol{x}(t)}\log p_{0t}(\boldsymbol{x}(t)|\boldsymbol{x}(0))\|_2^2\right]\right],$$

onde, t é amostrado uniformemente em $[0,T]$; $\boldsymbol{x}(0)\sim p_0(\boldsymbol{x})$ e $\boldsymbol{x}(t)\sim p_{0t}(\boldsymbol{x}(t)|\boldsymbol{x}(0))$. Sob certas condições e com dados suficientes, a solução ótima $s_{\boldsymbol{\theta}^\star}(\boldsymbol{x},t)$ equivale a $\nabla_{\boldsymbol{x}}\log p_t(\boldsymbol{x})$ para quase todo $\boldsymbol{x}$ e t. A escolha de função de pesos pode parecer arbitrária e leva ao questionamento de porque não por $\lambda\equiv 1$, mas sem uma escolha adequada dessa função o método não funciona. Uma escolha que é recomendada em [3] é $\lambda\propto\frac{1}{\mathbb{E}\left[\|\nabla_{x_t}\log p_{t|0}(x_t|x_0)\|_2^2\right]}$. Isso faz com que o começo do treino, onde tem regiões de baixa massa, vários modas e portanto é mais difícil de interpolar, receba menos peso na perda.

2.3 Exemplos: VE e VP SDEs

As perturbações com ruído usadas em SMLD e DDPM podem ser vistas como discretizações de duas SDEs diferentes.

Ao usar N escalas de ruído, cada kernel de perturbação $p_{\sigma_i}(\boldsymbol{x},\boldsymbol{x}_0)$ do SMLD corresponde a distribuição de $\boldsymbol{x}_i$ na seguinte cadeia de Markov:

$$\boldsymbol{x}_i=\boldsymbol{x}_{i-1}+\sqrt{\sigma_i^2-\sigma_{i-1}^2}z_{i-1},\ i=1,\dots,N, \tag{6}$$

com $z_{i-1}\sim\text{MVN}\,(0,\ I)$ e $\sigma_0=0$. Essa equação pode ser vista como uma versão discreta da seguinte SDE (veja detalhes no apêndice C)

$$d\boldsymbol{x}=\sqrt{\frac{d\sigma^2(t)}{dt}}dW.$$

Analogamente, no caso do DDPM, a cadeia discreta é

$$\boldsymbol{x}_i=\sqrt{1-\beta_i}\boldsymbol{x}_{i-1}+\sqrt{\beta_i}z_{i-1},\ i=1,\dots,N,$$

que pode ser vista como um versão discreta da seguinte SDE (veja detalhes no apêndice D)

$$d\boldsymbol{x}=-\frac{1}{2}\beta(t)\boldsymbol{x}dt+\sqrt{\beta(t)}dW. \tag{7}$$

A primeira SDE é chamada de *Variance Exploding* (VE), pois é um processo no qual a variância explode quando $t\to\infty$; enquanto a última SDE é dita *Variance Preserving* (VP). Vale destacar a sub-VP SDE inspirada na VP, cuja equação é dada por

$$d\boldsymbol{x}=-\frac{1}{2}\beta(t)\boldsymbol{x}dt+\sqrt{\beta(t)\left[1-\exp\left(-2\int_0^t\beta(s)ds\right)\right]}dW.$$

Todas essas SDEs possuem coeficientes de *drift* afins e, portanto, pode-se mostrar que seus kernels de pertubação $p_{0t}(\boldsymbol{x}(t)|\boldsymbol{x}(0))$ são todos gaussianos, conforme abaixo:

$$p_{0t}(\boldsymbol{x}(t)|\boldsymbol{x}(0)) = \begin{cases} \text{MVN}\left(\boldsymbol{x}(t);\ \boldsymbol{x}(0),\ [\sigma^2(t) - \sigma^2(0)]I\right), & \text{VE SDE} \\ \text{MVN}\left(\boldsymbol{x}(t);\ \boldsymbol{x}(0)\exp\left(-\int_0^t \beta(s)ds\right),\ I - I\exp\left(-\int_0^t \beta(s)ds\right)\right), & \text{VP SDE} \\ \text{MVN}\left(\boldsymbol{x}(t);\ \boldsymbol{x}(0)\exp\left(-\int_0^t \beta(s)ds\right),\ \left[1 - \exp\left(-\int_0^t \beta(s)ds\right)\right]^2 I\right), & \text{sub-VP SDE} \end{cases}$$

Em [3] o autor recomenda usar $\sigma(t) = \sigma_{min}\left(\frac{\sigma_{max}}{\sigma_{min}}\right)^t$, onde $\sigma_{min} = 0.01$ e σ_{max} é escolhido segundo alguma heurística, como em [7]. Enquanto $\beta(t) = \beta_{min} + (\beta_{max} - \beta_{min})t$, onde $\beta_{min} = 0.1$ e $\beta_{max} = 20.0$. Em todo caso, pomos o tempo final $T = 1$. Outra consideração prática importante é evitar avilar a SDE próximo de $t = 0$ para evitar problemas numéricos, consideramos então a SDE no domínio $[\varepsilon, T]$, onde $\epsilon = 10^{-6}$ ou 10^{-5}.

3 Usando a SDE reversa

3.1 Gerando amostras com a SDE reversa

De modo geral, há vários métodos numéricos para aproximar trajetórias de SDEs, como Euler-Maruyama e Runge-Kutta, baseados em diferentes discretizações. Porém, desenvolver formas de amostragem ancestral para SDEs pode ser não trivial. Como solução, é proposto um amostrador de difusão reversa: dada a equação *foward* $d\boldsymbol{x} = f(\boldsymbol{x}, t)dt + G(t)dW$ e supondo que $\boldsymbol{x}_{i+1} = \boldsymbol{x}_i + f_i(\boldsymbol{x}_i) + G_i z_i,\ i = 0, 1, \ldots, N-1$ é sua discretização, onde $z_i \sim \text{MVN}(0,\ I)$, é proposto discretizar a SDE de tempo reverso abaixo.

$$d\boldsymbol{x} = [f(\boldsymbol{x}, t) - G(t)G(t)^\top \nabla_{\boldsymbol{x}} \log p_t(\boldsymbol{x})]dt + G(t)d\bar{W},$$

cuja discretização é

$$\boldsymbol{x}_i = \boldsymbol{x}_{i+1} - f_{i+1}(\boldsymbol{x}_{i+1}) + G_{i+1}G_{i+1}^\top s_{\boldsymbol{\theta}^\star}(\boldsymbol{x}_{i+1}, i+1) + G_{i+1}z_{i+1},\ i = 0, 1, \ldots, N-1,$$

para o *score* treinado $s_{\boldsymbol{\theta}^\star}(\boldsymbol{x}_i, i)$. A equação escrita acima é o método de Euler-Maruyama para a equação reversa.

O artigo ainda comenta a possibilidade de usar um esquema numérico mais complexo, fazendo um método conhecido como Preditor-Corretor. Uma vez obtida uma aproximação $s_{\boldsymbol{\theta}^\star}(\boldsymbol{x}, t)$, é possível amostrar p_t usando MCMC, como Langevin MCMC ou HMC e, então, obter a solução numérica da SDE. Mais detalhadamente, o solucionador da SDE fornece uma estimativa da amostra no próximo passo, fazendo o papel de "preditor", enquanto a abordagem de MCMC corrige a distribuição marginal da amostra estimada, exercendo o papel de "corretor".

3.2 Fluxo de probabilidade e conexão com ODEs neurais

Modelos baseados na função score permitem outro método de resolução da SDE reversa. Para todo processo de difusão, existe um processo determinístico cuja trajetórias compartilham a mesma densidade de probabilidade marginal $\{p_t(\boldsymbol{x})\}_{t=0}^T$ com a SDE, desde que o ponto inicial seja amostrado conforma a distribuição inicial da SDE (que precisa ao menos ser C^2 com suporte cheio). Essa EDO é chamada de EDO de fluxo e é dada por:

$$d\boldsymbol{x} = \left[f(\boldsymbol{x}, t) - \frac{1}{2}g(t)\nabla_{\boldsymbol{x}} \log p_t(\boldsymbol{x})\right] dt.$$

Para fazer geração reversa, usamos essa EDO com modelo de ponto inicial $x_0 \sim \gamma$. O artigo recomenda simular essa EDO usando um *black box*, especificamente um RK45 chamado pela função scipy.integrate.solve_ivp com parâmetros atol=10^{-5} e rtol=10^{-5}.

4 Geração Condicional

Agora, imagine que treinamos um classificador que gera amostra de um determinado conjunto de imagens que tenha classes. A abordagem apresentada nesse texto permite-nos amostrar escolhendo a classe. A modelagem consiste em ver cada classe com uma variável $\boldsymbol{y}$ e então cada imagem tem uma probabilidade $p_0(\boldsymbol{y}|\boldsymbol{x}(0))$ de pertencer a classe $\boldsymbol{y}$. O que fazemos então é treinar um classificador que aproxima $p_t(\boldsymbol{y}|\boldsymbol{x}(t))$ minimizando uma simples entropia cruzada. Então se quisermos amostrar de uma classe específica, precisamos amostrar da seguinte SDE.

$$d\boldsymbol{x} = \left\{ f(\boldsymbol{x}, t) - g(t)^2 \left[\nabla_{\boldsymbol{x}} \log p_t(\boldsymbol{x}|\boldsymbol{y}) \right] \right\} dt + g(t) d\bar{W}.$$

Para fazer isso, usamos a regra de Bayes

$$p_t(\boldsymbol{x}|\boldsymbol{y}) \propto p_t(\boldsymbol{y}|\boldsymbol{x}) p_t(\boldsymbol{x}),$$

e então $\nabla \log p_t(\boldsymbol{x}|\boldsymbol{y}) = \nabla \log p_t(\boldsymbol{y}|\boldsymbol{x}) + \nabla \log p_t(\boldsymbol{x})$. Logo, a equação acima se escreve como

$$d\boldsymbol{x} = \left\{ f(\boldsymbol{x}, t) - g(t)^2 \left[\nabla \log p_t(\boldsymbol{y}|\boldsymbol{x}) + \nabla \log p_t(\boldsymbol{x}) \right] \right\} dt + g(t) d\bar{W}.$$

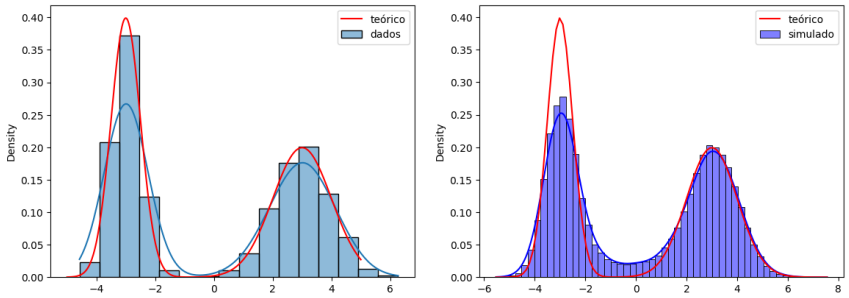
Em geral, $\boldsymbol{y}$ não necessariamente precisa ser uma classe, mas qualquer variável da qual possa se estimar $p_t(\boldsymbol{y}|\boldsymbol{x})$. Quando $\boldsymbol{y}$ representa classes, é possível treinar um classificador $p_t(\boldsymbol{y}|\boldsymbol{x}(t))$. Como a SDE é tratável, é fácil criar dados de treinamento $(\boldsymbol{x}(t), \boldsymbol{y})$ para o classificador amostrando $(\boldsymbol{x}(0), \boldsymbol{y})$ de algum conjunto de dados e, então, amostrando $\boldsymbol{x}(t) \sim p_{0t}(\boldsymbol{x}(t)|\boldsymbol{x}(0))$. A partir disso, pode-se empregar uma mistura de perdas de entropia cruzada ao longo de diferentes passos de tempo para treinar um classificador $p_t(\boldsymbol{y}|\boldsymbol{x}(t))$.

5 Resultados

Foi implementado o modelo e feito alguns experimentos, descritos a seguir.

5.1 Mistura de Gaussianas

O primeiro experimento feito, foi utilizar um dataset com 5000 pontos simulados de uma mistura de gaussianas, especificamente $\sim \frac{1}{2}\mathcal{N}(-3, 1/4) + \frac{1}{2}\mathcal{N}(3, 1)$. Os dados estão na figura (3a).



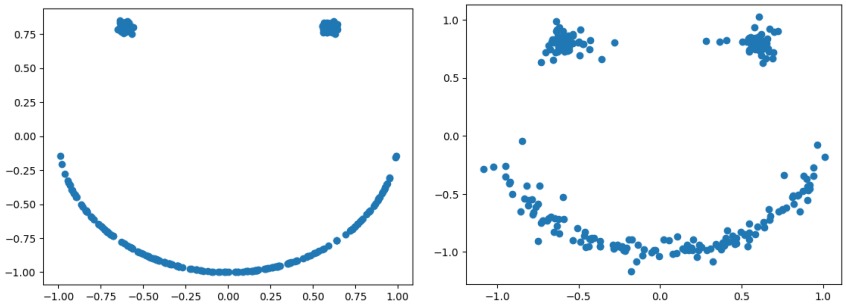
(a) Dados verdadeiros (b) Dados gerados pelo modelo

Figura 3: Mistura de Gaussianas

Usamos a SDE VP, descrita acima, e uma MLP com um única camada oculta com 128 neurônios e ativação ReLU, treinada com método Adam, com learning rate de 10^{-4}, batch size de 64, e por 300 épocas. Treinamos com um gride uniforme no tempo de 100 pontos e com 30 simulações para cada ponto. Por fim, usamos um método de Euler-Maruayama para amostrar de SDE reversa. Na figura (3b) encontra-se o resultado.

5.2 Sorriso

O próximo experimento foi feito com um dataset com 100 pontos gerados de uma normal bivariada de média $(-0.6, 0.8)$ e desvio padrão 0.02, 100 pontos gerados de uma normal bivariada de média $(0.6, 0.8)$ e desvio padrão 0.02, e 200 pontos gerados sobre a curva $(x, -\sqrt{1-x^2})$ com $x \sim Unif[-1, 1]$, veja a figura (4a).



(a) Dados reais (b) Dados gerados pelo modelo

Figura 4: Dados do sorriso

Usamos a SDE VP, descrita acima, e uma MLP com um duas camadas ocultas com 128 neurônios cada e ativação ReLU, treinada com método Adam, com learning rate de 10^{-4}, batch size de 64, e por 7000 épocas. Treinamos com um gride uniforme no tempo de 100 pontos e com 10 simulações para cada ponto. Na simulação reversa usamos a EDO de fluxo com implementação do scipy conforme descrito anteriormente. Na figura (4b) estão os dados simulados pelo modelo.

5.3 Dígitos Escritos a Mão

Por fim, usamos o conjunto de load_digits do scikit-learning como dado de treinamento (veja a figura (5)).

Figura 5: Amostra de números escritos a mão.

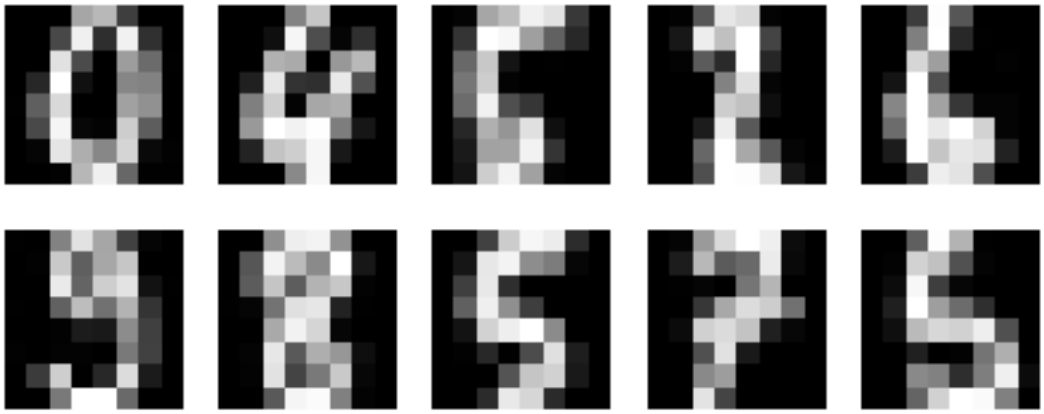


Figura 6: Amostra gerada de números escritos a mão.

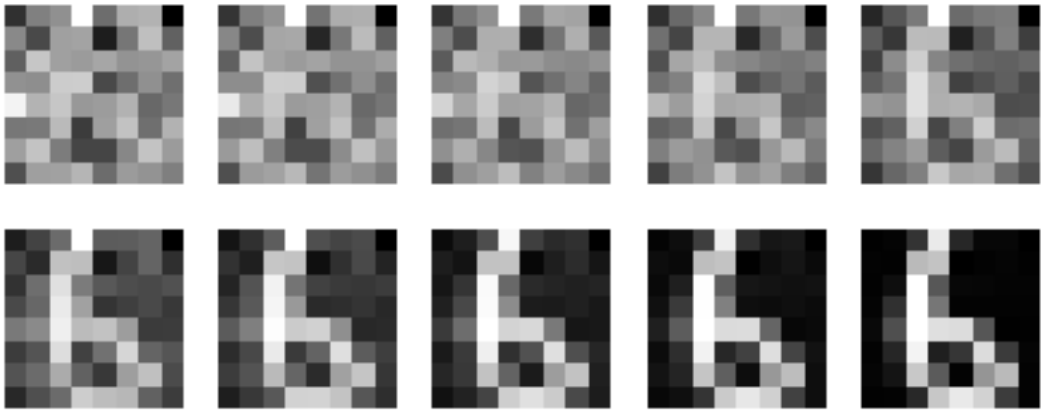


Figura 7: Passo da SDE reversa em uma geração.

A arquitetura desse exemplo é bem mais complicada. Primeiro precisamos de uma camada de *Projeções Gaussianas de Fourier* para transformar o input de tempo em 256 inputs. Depois intercalamos entre 7 camadas convolucionais e densa. Essa é a arquitetura sugerida pelo artigo e disponível no repositório do autores. Treinamos com batch size de 4, usando 150 pontos de discretização temporal e 10 amostras para cada tempo, por 15 épocas. Por fim, amostramos usando a EDO de fluxo. Na figura (6) vemos o resultado e na figura (7), um exemplo de um processo de geração.

6 Conclusão

Conseguimos reproduzir as ideias básicas apresentadas no artigo, onde o método se mostrou funcional em boa parte dos experimentos realizados, apesar da nossa implementação ter ficado lenta e ocupar muita memória. Isso se deve ao fato de que o artigo não comenta como ele lida com a discretização temporal no processo de treino e nem quantas simulações foram feitas para cada ponto, certamente, omitindo algum processo de amostragem que torna o código mais eficiente e lida melhor com a memória.

Referências

[1] Yang Song and Stefano Ermon. Generative modeling by estimating gradients of the data distribution, 2020.

[2] Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. *NeurIPS*, 2020.

[3] Yang Song, Jascha Sohl-Dickstein, Diederik P. Kingma, Abhishek Kumar, Stefano Ermon, and Ben Poole. Score-based generative modeling through stochastic differential equations, 2021.

[4] Brian D.O. Anderson. Reverse-time diffusion equation models. *Stochastic Processes and their Applications*, 12(3):313–326, 1982.

[5] Gareth O. Roberts and Richard L. Tweedie. Exponential convergence of langevin distributions and their discrete approximations. *Bernoulli*, 2(4):341–363, 1996.

[6] Steve Brooks, Andrew Gelman, Galin Jones, and Xiao-Li Meng. *Handbook of Markov Chain Monte Carlo*. Chapman and Hall/CRC, May 2011.

[7] Yang Song and Stefano Ermon. Improved techniques for training score-based generative models, 2020.

Apêndice

A *Score matching* com dinâmica de Langevin (SMLD)

Seja um kernel de pertubação $p_\sigma(\tilde{\boldsymbol{x}} \mid \boldsymbol{x}) = \text{MVN}\left(\tilde{\boldsymbol{x}};\ 0,\ \sigma^2 I\right)$ e $p_\sigma(\tilde{\boldsymbol{x}}) = \int p_{\text{data}}(\boldsymbol{x}) p_\sigma(\tilde{\boldsymbol{x}} \mid \boldsymbol{x}) d\boldsymbol{x}$, onde $p_{\text{data}}(\boldsymbol{x})$ denota a distribuição dos dados. Além disso, considere uma sequência de escalas de ruídos $\sigma_{\min} = \sigma_1 < \ldots < \sigma_N = \sigma_{\max}$, de modo que $\sigma_{\min}$ seja pequeno o suficiente para $p_{\sigma_{\min}}(\boldsymbol{x}) \approx p_{\text{data}}(\boldsymbol{x})$ e $\sigma_{\max}$ seja grande o suficiente para $p_{\sigma_{\max}}(\boldsymbol{x}) \approx \text{MVN}\left(\boldsymbol{x};\ 0,\ \sigma_{\max}^2 I\right)$.

Para obter uma aproximação da função score, $\nabla_{\boldsymbol{x}} \log p_\sigma(\boldsymbol{x})$, é proposto treinar uma rede condicional de score de ruído (NCSN), denotada por $s_{\boldsymbol{\theta}}(\boldsymbol{x}, \sigma)$, e obter um modelo ótimo $s_{\boldsymbol{\theta}^\star}(\boldsymbol{x}, \sigma)$ de modo que

$$\boldsymbol{\theta}^\star = \arg\min_{\boldsymbol{\theta}} \sum_{i=1}^{N} \sigma_i^2 \mathbb{E}_{p_{\text{data}}(\boldsymbol{x})} \mathbb{E}_{p_{\sigma_i}(\tilde{\boldsymbol{x}} \mid \boldsymbol{x})} [\| s_{\boldsymbol{\theta}}(\tilde{\boldsymbol{x}}, \sigma_i) - \nabla_{\tilde{\boldsymbol{x}}} \log p_{\sigma_i}(\tilde{\boldsymbol{x}} \mid \boldsymbol{x}) \|_2^2].$$

Com dados suficientes e com um modelo bem especificado, é possível mostrar que $s_{\boldsymbol{\theta}^\star}(\boldsymbol{x}, \sigma)$ corresponde a $\nabla_{\boldsymbol{x}} \log p_\sigma(\boldsymbol{x})$ para quase todo $\sigma \in \{\sigma_i\}_{i=1}^N$.

Para o processo de amostragem para cada $p_{\sigma_i}(\boldsymbol{x})$, considera-se M passos sequenciais de MCMC de Langevin da seguinte forma:

$$\boldsymbol{x}_i^m = \boldsymbol{x}_i^{m-1} + \epsilon_i s_{\boldsymbol{\theta}^\star}(\boldsymbol{x}_i^{m-1}, \sigma_i) + \sqrt{2\epsilon_i} \boldsymbol{z}_i^m,\ m = 1, \ldots, M,$$

onde ϵ_i é o tamanho do passo e $\boldsymbol{z}_i$ é normal padrão. Esse processo é repetido para $i = N, N-1, \ldots, 1$, com $\boldsymbol{x}_N^0 \sim \text{MVN}\left(0,\ \sigma_{\max} I\right)$ e $\boldsymbol{x}_i^0 = \boldsymbol{x}_{i+1}^M$ quando $i < N$. Sob algumas condições de regularidade, quando $M \to \infty$ e $\epsilon_i \to 0\ \forall i$, $\boldsymbol{x}_1^M$ é uma amostra exata de $p_{\sigma_{\min}}(\boldsymbol{x}) \approx p_{\text{data}}(\boldsymbol{x})$.

B Modelos probabilísticos de difusão de eliminação de ruído (DDPM)

Semelhante ao modelo anterior, considera-se uma sequência de escalas de ruído $\{\beta_i\}_{i=1}^N$ tal que $0 < \beta_i < 1$, $i = 1, \ldots, N$. A partir disso, para cada ponto de dado $\boldsymbol{x}_0 \sim p_{\text{data}}(\boldsymbol{x})$, constrói-se uma cadeia de Markov $\{\boldsymbol{x}_0, \ldots, \boldsymbol{x}_N\}$ de modo que a transição é dada por $p(\boldsymbol{x}_i \mid \boldsymbol{x}_{i-1}) = \text{MVN}\left(\boldsymbol{x}_i;\ \sqrt{1-\beta_i}\boldsymbol{x}_{i-1},\ \beta_i I\right)$. Denotando $\alpha_i = \prod_{j=1}^{i}(1-\beta_j)$, conclui-se que $p_{\alpha_i}(\tilde{\boldsymbol{x}} \mid \boldsymbol{x}_0) = \text{MVN}\left(\boldsymbol{x}_i;\ \sqrt{\alpha_i}\boldsymbol{x}_0,\ (1-\alpha_i) I\right)$ e $p_{\alpha_i}(\tilde{\boldsymbol{x}}) = \int p_{\text{data}}(\boldsymbol{x}) p_{\alpha_i}(\tilde{\boldsymbol{x}} \mid \boldsymbol{x}) d\boldsymbol{x}$. Uma cadeia de Markov variacional na direção reversa é parametrizada com $p_{\boldsymbol{\theta}}(\boldsymbol{x}_{i-1} \mid \boldsymbol{x}_i) = \text{MVN}\left(\boldsymbol{x}_{i-1};\ \frac{1}{\sqrt{1-\beta_i}}(\boldsymbol{x}_i + \beta_i s_{\boldsymbol{\theta}}(\boldsymbol{x}_i, i)),\ \beta_i I\right)$ e treinada com uma variante da ELBO:

$$\boldsymbol{\theta}^\star = \arg\min_{\boldsymbol{\theta}} \sum_{i=1}^{N} (1-\alpha_i) \mathbb{E}_{p_{\text{data}}(\boldsymbol{x})} \mathbb{E}_{p_{\alpha_i}(\tilde{\boldsymbol{x}}|\boldsymbol{x})} \left[\| s_{\boldsymbol{\theta}}(\tilde{\boldsymbol{x}}, i) - \nabla_{\tilde{\boldsymbol{x}}} \log p_{\alpha_i}(\tilde{\boldsymbol{x}} \mid \boldsymbol{x}) \|_2^2 \right].$$

Assim, dado o modelo ótimo $s_{\boldsymbol{\theta}^\star}(\boldsymbol{x}_i, i)$, amostras podem ser geradas iniciando com $\boldsymbol{x}_N \sim \text{MVN}\,(0,\ I)$ e seguindo a cadeia de Markov reversa:

$$\boldsymbol{x}_{i-1} = \frac{1}{\sqrt{1-\beta_i}} (\boldsymbol{x}_i + \beta_i s_{\boldsymbol{\theta}^\star}(\boldsymbol{x}_i, i)) + \sqrt{\beta_i} \boldsymbol{z}_i,\ i = N, \ldots, 1.$$

C SMLD como discretização da VP SDE

Ao usar N escalas de ruído, cada kernel de perturbação $p_{\sigma_i}(\boldsymbol{x}, \boldsymbol{x}_0)$ do SMLD corresponde a distribuição de $\boldsymbol{x}_i$ na seguinde cadeia de Markov:

$$\boldsymbol{x}_i = \boldsymbol{x}_{i-1} + \sqrt{\sigma_i^2 - \sigma_{i-1}^2} z_{i-1},\ i = 1, \ldots, N, \tag{8}$$

com $z_{i-1} \sim \text{MVN}\,(0,\ I)$ e $\sigma_0 = 0$.

Quando $N \to \infty$, $\{\sigma_i\}_{i=1}^N$ se torna uma função $\sigma(t)$, z_i se torna $z(t)$ e a cadeia de Markov $\{\boldsymbol{x}_i\}_{i=1}^N$ se torna um processo estocástico contínuo $\{\boldsymbol{x}(t)\}_{t=0}^1$, que é dado pela seguinte SDE:

$$d\boldsymbol{x} = \sqrt{\frac{d\sigma^2(t)}{dt}} dW.$$

Para verificar isso, considere $\boldsymbol{x}(i/N) = \boldsymbol{x}_i$, $\sigma(i/N) = \sigma_i$ e $z(i/N) = z_i$, $i = 1, \ldots, N$. Então, pode-se reescrever a equação (8) com $\Delta t = \frac{1}{N}$ e $t \in \{0, 1/N, \ldots, (N-1)/N\}$:

$$\boldsymbol{x}(t + \Delta t) = \boldsymbol{x}(t) + \sqrt{\sigma^2(t + \Delta t) - \sigma^2(t)} z(t) \approx \boldsymbol{x}(t) + \sqrt{\frac{d\sigma^2(t)}{dt} \Delta t} z(t).$$

Aplicando o limite $\Delta t \to 0$, o resultado segue.

D DDPM como discretização da VE SDE

No caso do DDPM, a cadeia discreta é

$$\boldsymbol{x}_i = \sqrt{1-\beta_i} \boldsymbol{x}_{i-1} + \sqrt{\beta_i} z_{i-1},\ i = 1, \ldots, N.$$

que pode ser vista como um versão discreta da seguinte SDE

$$d\boldsymbol{x} = -\frac{1}{2} \beta(t) \boldsymbol{x} dt + \sqrt{\beta(t)} dW. \tag{9}$$

Quando $N \to \infty$, a equação anterior converge para a SDE a seguir:

$$d\boldsymbol{x} = -\frac{1}{2} \beta(t) \boldsymbol{x} dt + \sqrt{\beta(t)} dW. \tag{10}$$

Para ter uma ideia da prova, considere escalas de ruídos auxiliares $\{\bar{\beta}_i = N\beta_i\}_{i=1}^N$. Assim, pode-se reescrever a equação (10) como

$$\boldsymbol{x}_i = \sqrt{1 - \frac{\bar{\beta}_i}{N}}\boldsymbol{x}_{i-1} + \sqrt{\frac{\bar{\beta}_i}{N}}z_{i-1},\ i = 1, \ldots, N.$$

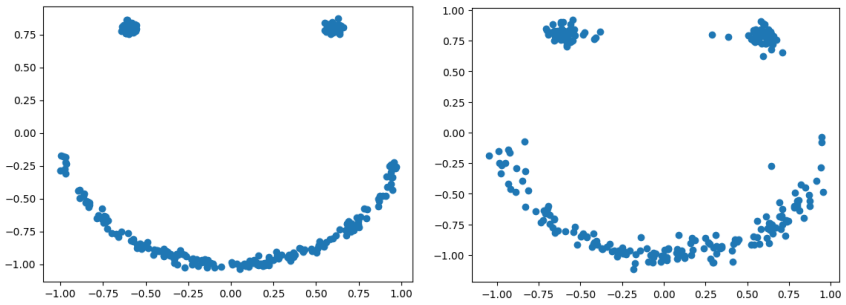
Agora, sejam $\beta(i/N) = \bar{\beta}_i,\ \boldsymbol{x}(i/N) = \boldsymbol{x}_i,\ z(i/N) = z_i$. Daí, a equação anterior resulta em

$$\begin{aligned}
\boldsymbol{x}(t + \Delta t) &= \sqrt{1 - \beta(t + \Delta t)\Delta t}\boldsymbol{x}(t) + \sqrt{\beta(t + \Delta t)\Delta t}z(t), \\
&\approx \boldsymbol{x}(t) - \frac{1}{2}\beta(t + \Delta t)\Delta t\boldsymbol{x}(t) + \sqrt{\beta(t + \Delta t)\Delta t}z(t), \\
&\approx x(t) - \frac{1}{2}\beta(t)\Delta t\boldsymbol{x}(t) + \sqrt{\beta(t)\Delta t}z(t).
\end{aligned}$$

Novamente, tomando $\Delta t \to 0$, conclui-se o resultado.

E Sorriso com ruído

Refizemos o experimento do sorriso, somando um pequeno ruído normal de média 0 e desvio padrão 0.02 na "boca" do sorriso. Os resultados estão na figuras (8a) e (8b)



(a) Dados reais (b) Dados gerados pelo modelo

Figura 8: Dados do sorriso com ruído

F Geração Condicional

Fizemos ainda a geração condicional, treinando um classificador para as labels do conjunto de dígitos. O classificador é um simples MLP com 65 entradas (64 pixels + tempo) com duas cadas ocultas e 128 neurônios cada com ativação ReLU e 10 saídas (as 10 classes do modelo). O resultado do classificador condicional está na figura (9).

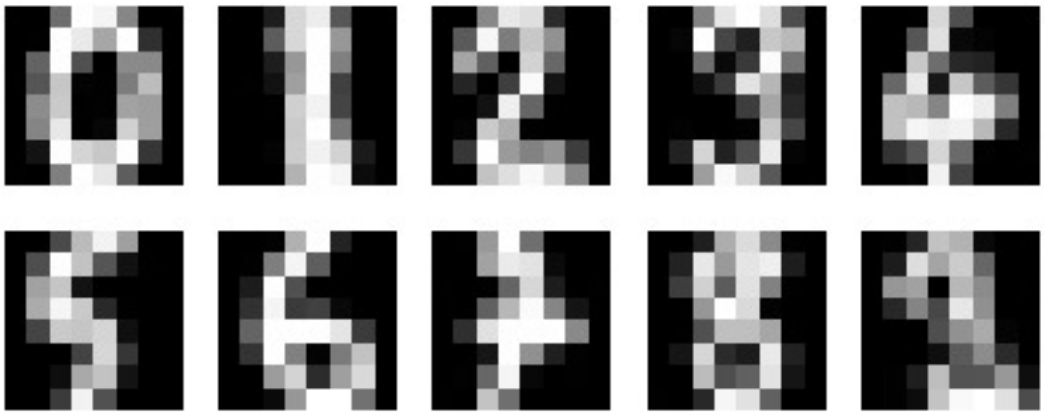


Figura 9: Geração condicional a label do conjunto de dados.

G Repositório do trabalho

Os códigos utilizados para a realização dos experimentos se encontram nesse repositório.