

UNIVERSITÀ DEGLI STUDI DI  
CASSINO E DEL LAZIO MERIDIONALE



DIPARTIMENTO DI INGEGNERIA ELETTRICA  
E DELL'INFORMAZIONE "MAURIZIO SCARANO"

CORSO DI LAUREA IN INGEGNERIA INFORMATICA  
E DELLE TELECOMUNICAZIONI

**Generatori di immagini  
da prompt testuali: stato dell'arte**

**Relatore:**

Prof. Alessandro Bria

**Candidato:**

Giuseppe Alfieri

ANNO ACCADEMICO 2022/2023

*Alla cara memoria di mio padre Donato*

Se hai qualcuno nel cuore,  
l'hai vicino per sempre.

Marcia Grad Powers, *La principessa  
che credeva nelle favole*

“Credi che le persone scomparse che abbiamo amato ci lascino mai del tutto? Non credi che le ricordiamo più chiaramente che mai nei momenti di grande difficoltà? Tuo padre è vivo in te, Harry, e si mostra soprattutto quando hai bisogno di lui. [...] Quindi ieri notte hai visto tuo padre, Harry... l'hai trovato dentro di te.”

J.K. Rowling, *Harry Potter  
e il prigioniero di Azkaban*



# Abstract

I generatori di immagini da prompt testuali, quali *Midjourney* [14], sono in grado di creare, partendo da descrizioni testuali, immagini di alta qualità che siano quanto più fedeli possibile alla *semantica* del testo. Queste tecnologie si avvalgono di modelli generativi che, a loro volta, sottendono l'impiego di reti neurali.

In questa tesi si fornirà, preliminarmente, una panoramica generale della modellazione generativa, declinata al contesto della generazione di immagini, congiuntamente ad una visione d'insieme delle famiglie di modelli generativi più in auge in letteratura.

Successivamente verrà discussa la *ratio* sottesa a una delle tecnologie, nella pletora di modelli generativi, che rendono possibile la “magia” di generare immagini: i modelli probabilistici di diffusione del rumore (*Denoising Diffusion Probabilistic Models*) (DDPM).

# Indice

|          |  |           |
|----------|--|-----------|
| <b>1</b> | <b>Introduzione</b>  | <b>1</b>  |
| <b>2</b> | <b>La modellazione generativa</b>                                | <b>3</b>  |
| 2.1      | Introduzione . . . . .   | 3         |
| 2.2      | Modellazione generativa e discriminativa a confronto . . . . .   | 4         |
| 2.3      | Tassonomia dei modelli generativi . . . . .                      | 5         |
| <b>3</b> | <b>Modelli di diffusione</b>                                     | <b>8</b>  |
| 3.1      | Introduzione . . . . .   | 8         |
| 3.2      | Modelli probabilistici di diffusione del rumore (DDPM) . . . . . | 8         |
| 3.2.1    | Processo di diffusione in avanti . . . . .                       | 9         |
| 3.2.2    | Processo di diffusione inversa . . . . .                         | 17        |
| 3.2.3    | Addestramento . . . . .  | 18        |
| 3.2.4    | Generazione di immagini . . . . .                                | 23        |
| 3.2.5    | Prestazioni . . . . .  | 25        |
| <b>4</b> | <b>Conclusioni</b>   | <b>27</b> |
| <b>A</b> | <b>Nozioni di probabilità e statistica</b>                       | <b>29</b> |
| A.1      | Elementi di teoria della probabilità . . . . .                   | 29        |
| A.1.1    | Distribuzioni di probabilità . . . . .                           | 29        |
| A.1.2    | Caratterizzazione sintetica di vettori aleatori . . . . .        | 31        |
| A.1.3    | Indipendenza statistica . . . . .                                | 32        |
| A.1.4    | Distribuzione gaussiana . . . . .                                | 33        |
| A.1.5    | Catena Markoviana . . . . .                                      | 35        |
| A.2      | Problema della stima dei parametri . . . . .                     | 36        |
| A.2.1    | Modellazione parametrica . . . . .                               | 36        |
| A.2.2    | Funzione di verosimiglianza . . . . .                            | 36        |
| A.2.3    | Metodo della massima verosimiglianza . . . . .                   | 37        |
| <b>B</b> | <b>Derivazione funzione di costo DDPM</b>                        | <b>38</b> |

|                |           |
|----------------|-----------|
| <b>C U-Net</b> | <b>44</b> |
|----------------|-----------|

|                     |           |
|---------------------|-----------|
| <b>Bibliografia</b> | <b>47</b> |
|---------------------|-----------|

# Capitolo 1

## Introduzione

La generazione di immagini da prompt testuali è atta alla creazione di rappresentazioni visive realistiche e coerenti a partire da descrizioni linguistiche, avvalendosi di modelli generativi.

Stante la complessità che la generazione di immagini e la comprensione del testo *eo ipso* rappresentano, la generazione di immagini da prompt testuali, essendo il combinato disposto delle due, presenta un’ulteriore difficoltà: il passaggio da un dominio di rappresentazione all’altro per convertire esplicite relazioni testuali tra entità, in immagini consistenti con il significato del testo.

Inoltre, un buon modello deve poter mescolare concetti e stili che non ha mai visto prima per generare immagini inedite. Ad esempio, non esistono ritratti di Kim Jong-Un in camera da letto che stringe degli orsacchiotti, ma, ricorrendo a tecnologie che sottendono l’addestramento di reti neurali, è possibile creare una siffatta immagine (Figura 1.1). Inoltre, sarebbe auspicabile che il modello desumesse accuratamente il modo in cui gli oggetti nell’immagine generata si relazionano tra loro, in base alla semantica del messaggio testuale e *come* viene conferito significato alle parole attraverso il contesto [8]. Ad esempio, l’immagine di “una persona con un salvagente che galleggia in mare” dovrebbe apparire molto dissimile da quella di “una persona con un salvagente in un mare di gente”.

In questa trattazione si approfondirà la sola parte relativa alla generazione di immagini, piuttosto che il meccanismo con cui si estrapola la semantica da un blocco testuale e la conseguente creazione di un’immagine che ne rifletta il contenuto.

In particolare, il lavoro è così ripartito:

- nel Capitolo 2 si chiarisce cosa si intenda per modellazione generativa e la si confronta con la modellazione discriminativa. Inoltre, si riporta una tassonomia di modelli generativi.



(a) Prompt: "Pope Francis playing dj console, wearing dj headphones"



(b) Prompt: "Kim Jong-Un wearing hello kitty pattern pajama in bedroom holding a big hello kitty peluche"



(c) Prompt: "Trump escorted by police officers"



(d) Prompt: "Trump in prison doing bodybuilding"

**Figura 1.1:** Immagini da me create con *Midjourney* [14], corredate dai prompt testuali originali in lingua inglese.

- nel Capitolo 3 si illustra dettagliatamente una particolare tipologia di modelli di diffusione, che ha rappresentato un contributo pionieristico nell’ambito della modellazione generativa: i modelli probabilistici di diffusione del rumore (*Denoising Diffusion Probabilistic Models*) (DDPM). Tali modelli rimuovono gradualmente il rumore da una versione corrotta di una data immagine, pervenendo, con l’ausilio di una rete neurale, ad una stima della distribuzione dell’immagine originale. In appendice A si riportano alcune nozioni di base di probabilità e statistisca a supporto della discussione dei suddetti modelli, mentre nelle appendici B e C se ne forniscono, rispettivamente, i dettagli matematici e architetturali.

# Capitolo 2

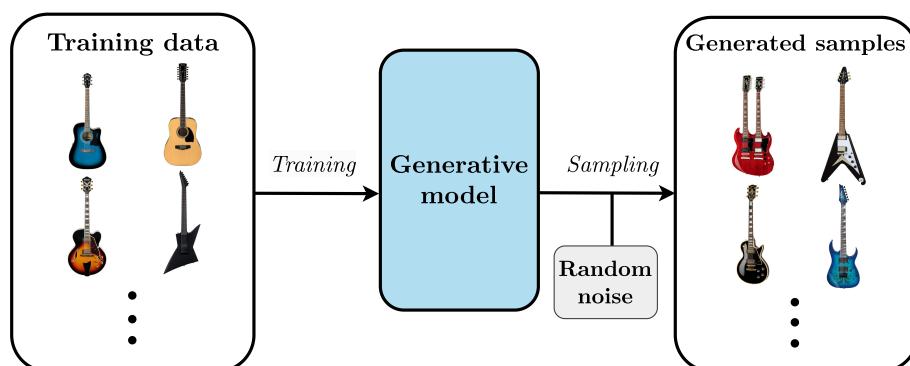
## La modellazione generativa

### 2.1 Introduzione

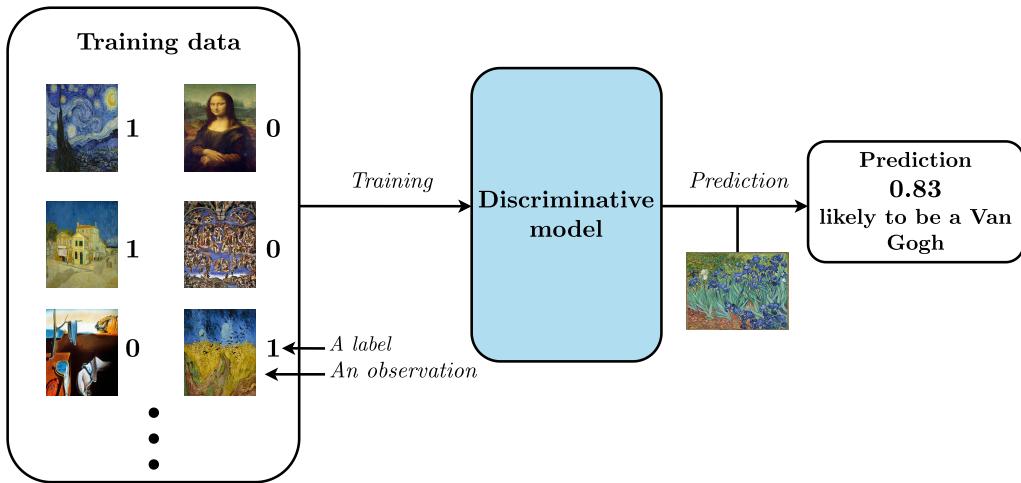
La modellazione generativa può essere definita come segue:

*“La modellazione generativa è una branca del machine learning che prevede l’addestramento di un modello per generare nuovi dati inediti, che sono correlati all’insieme di dati di addestramento originale.” (David Foster, 2023 [8])*

Si consideri, ad esempio, un insieme di dati (*dataset*) che annovera immagini di chitarre. È possibile *addestrare* un modello generativo su tale dataset per desumere le regole che governano le complesse relazioni sussistenti tra i pixel nelle immagini delle chitarre. A valle della fase di addestramento, avvalendosi del suddetto modello, è possibile creare nuove immagini inedite di chitarre che non erano presenti nel dataset di partenza (Figura 2.1).



**Figura 2.1:** Esempio di modello generativo. Fonte: [8].



**Figura 2.2:** Modello discriminativo addestrato a predire se una data immagine è stata dipinta da Van Gogh. Fonte: [8].

È importante osservare che un modello generativo è intrinsecamente *probabilistico*. Infatti la peculiare capacità di un modello generativo di produrre dati inediti sottende l'apprendimento della distribuzione incognita dei dati di addestramento, cosicché *campionando* da tale distribuzione sia possibile generare nuove istanze di dati: ciò richiede una certa variabilità e incertezza.

Qualora il modello fosse scevro da qualsivoglia elemento di aleatorietà (e.g. il modello è semplicemente un calcolo fisso, come prendere il valore medio di ogni pixel nel set di dati, [8]), stante la sua natura deterministica, non sarebbe generativo.

## 2.2 Modellazione generativa e discriminativa a confronto

Per una più accurata comprensione della modellazione generativa si rende opportuno un confronto con la controparte: la *modellazione discriminativa*.

In Figura 2.2 è riportato un modello discriminativo atto a discernere se la paternità di un dipinto è o no attribuibile a Van Gogh. Trattandosi di un tipico problema di classificazione binaria, il dataset di addestramento viene suddiviso in due gruppi: i dipinti di Van Gogh vengono etichettati con 1, laddove i dipinti di altri artisti sono etichettati con 0. Il suddetto modello viene quindi addestrato a *discriminare* tra i due gruppi e restituisce la probabilità che un'immagine, non presente nel dataset di addestramento, abbia come etichetta 1(i.e. sia effettivamente un quadro di Van Gogh) [8].

Si noti come, nel contesto della modellazione discriminativa, ogni osservazione nel dataset di addestramento sia corredata da un’etichetta (*label*). Di converso, l’etichettatura del dataset di addestramento non costituisce un requisito essenziale nell’ambito della modellazione generativa, concernente la creazione di immagini inedite piuttosto che la corretta attribuzione dell’etichetta ad una data immagine.

Di seguito si formalizza matematicamente quanto appena esposto:

- i modelli discriminativi stimano la probabilità  $p(y|\mathbf{x})$  che  $y$  sia l’etichetta corrispondente ad una data osservazione  $\mathbf{x}$  (Figura 2.2).
- i modelli generativi stimano la distribuzione incognita  $p(\mathbf{x})$  dei dati del dataset di addestramento: il modello genera, quindi, nuovi dati a partire dalla distribuzione appresa.

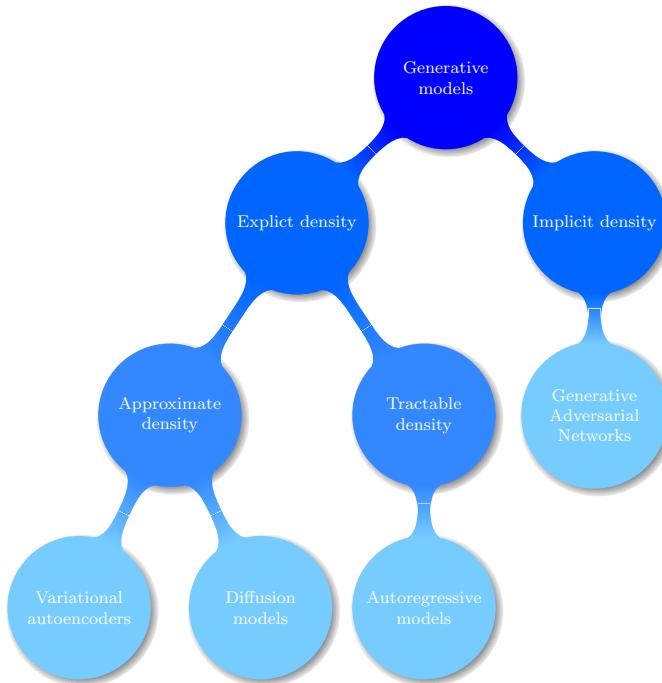
*Osservazione.* Si noti che un modello generativo può anche stimare la probabilità condizionata  $p(\mathbf{x}|y)$  di osservare il dato  $\mathbf{x}$  la cui etichetta è  $y$ . Ad esempio, se il dataset di addestramento contenesse immagini di diversi tipi di alberi, si potrebbe ricorrere ad un modello generativo per generare un’immagine di un cipresso: in tal caso si parla di *modelli generativi condizionati* [8].

Dunque, mentre l’obiettivo della modellazione discriminativa è quello di classificare i dati in base alle loro caratteristiche (*features*), la modellazione generativa concerne la creazione di nuovi dati che esibiscono affinità con il dataset di addestramento.

## 2.3 Tassonomia dei modelli generativi

Sebbene in questa trattazione si investigherà dettagliatamente solo una particolare categoria di modelli diffusione (i.e. i modelli di diffusione del rumore (DDPM)), è bene inquadrare il ruolo di quest’ultimi nel contesto più ampio dei modelli generativi che si avvalgono del metodo di massima verosimiglianza (Appendice A.2.3).

*Osservazione.* A rigore non tutti i modelli generativi ricorrono al metodo della massima verosimiglianza: le GAN, ad esempio, non utilizzano il metodo di massima verosimiglianza. Tuttavia, “*ignorando quei modelli che non ricorrono alla massima verosimiglianza, e concentrandosi sulla versione di massima verosimiglianza dei modelli che solitamente non utilizzano tale metodo (come le GAN), si possono eliminare alcune delle differenze che più distraggono tra modelli diversi*”(Goodfellow, 2016 [9]), a beneficio di una più compatta visione d’insieme dei modelli generativi.



**Figura 2.3:** Tassonomia dei modelli generativi. Adattato da [8, 9].

I modelli generativi che sottendono il ricorso al metodo della massima verosimiglianza (Appendice A.2.3) differiscono per il modo in cui rappresentano, o approssimano, la distribuzione di probabilità  $p_{\theta}(\mathbf{x})$ [9].

In letteratura si ravvisano tre approcci possibili [8] nella modellazione di  $p_{\theta}(\mathbf{x})$ :

- modellare *esplicitamente* la suddetta distribuzione di probabilità.
- modellare *esplicitamente* un'approssimazione della distribuzione di probabilità  $p_{\theta}(\mathbf{x})$ .
- modellare *implicitamente* la distribuzione di probabilità  $p_{\theta}(\mathbf{x})$ , delegando ad un processo stocastico la creazione dei dati.

In Figura 2.3 è riportata una *tassonomia* dei modelli generativi.

*Osservazione.* Le famiglie di modelli riportate in Figura 2.3 non sono mutuamente esclusive: in letteratura si individuano molti esempi di modelli che sono degli ibridi tra approcci differenti. Nella suddetta tassonomia, quindi, le diverse famiglie di modelli vanno interpretate come *approcci generali* alla modellazione generativa, piuttosto che architetture esplicite di modelli [8].

I modelli a densità implicita (*Implicit density models*) “non mirano a stimare la densità di probabilità  $p_\theta(\mathbf{x})$ , ma si concentrano unicamente sulla produzione di un processo stocastico che genera direttamente i dati” [8].

I modelli a densità esplicita (*Explicit density models*) si possono ulteriormente suddividere in quelli che ottimizzano direttamente la distribuzione di probabilità (*tractable models*), e quelli che ottimizzano un’approssimazione della stessa.

“*Un filo conduttore che attraversa tutti i tipi di modelli generativi è il deep learning. Quasi tutti i modelli generativi più sofisticati sottendono il ricorso a reti neurali.*” [8].

# Capitolo 3

## Modelli di diffusione

### 3.1 Introduzione

Scopo di questo capitolo è quello di analizzare la tecnologia che rappresenta il *deus ex machina* del complesso processo che rende possibile la “magia” di generare immagini inedite da prompt testuali: i modelli di diffusione.

A rigore, nel contesto della generazione di immagini, i modelli di diffusione non sono l'unica opzione possibile nel panorama dei modelli generativi. Tuttavia i modelli di diffusione, superando in termini prestazionali le GAN nella generazione di immagini (Dhariwal e Nichol, 2021 [6]), sono assurti allo stato dell'arte in tale campo.

Il termine *diffusione* è mutuato dal concetto di diffusione termodinamica: Sohl-Dickstein et al. (2015 [19]) hanno stabilito un importante collegamento tra tale concetto, inerente alla fisica, e il deep learning.

Nel prosieguo si investigherà dettagliatamente la *ratio* sottesa ad una particolare categoria di modelli di diffusione: i modelli probabilistici di diffusione del rumore (DDPM).

### 3.2 Modelli probabilistici di diffusione del rumore (DDPM)

I modelli probabilistici di diffusione del rumore (DDPM) sono stati introdotti in un articolo pubblicato nell'estate del 2020 (Ho et al., [12]) che ha segnato un punto di svolta nell'ambito dei modelli di diffusione e della modellazione generativa.

La *pipeline* di un DDPM è la seguente:

- nel *processo di diffusione in avanti*, un’immagine è corrotta dal rumore fino a risultare, all’acme del processo, indistinguibile dallo stesso. La diffusione in avanti è funzionale a ricondursi, da un distribuzione di probabilità arbitrariamente complessa dell’immagine di partenza, al rumore puro che, invece, esibisce una distribuzione nota e facilmente manipolabile.
- nel *processo di diffusione inversa*, con l’ausilio di una rete neurale, si rimuove progressivamente il rumore, ripercorrendo a ritroso la diffusione in avanti, fino a pervenire all’immagine scevra da rumore.

### 3.2.1 Processo di diffusione in avanti

Il processo di diffusione in avanti (*forward diffusion process*) è definito come una *catena markoviana* (si veda A.1.5) che va a perturbare la distribuzione originale, incognita,  $q(\mathbf{x}_0)$  di un’immagine  $\mathbf{x}_0$  del dataset di addestramento, corrompendola con rumore additivo gaussiano (Figura 3.1).

Qualora il suddetto rumore additivo abbia entità contenuta, le *transizioni* della catena markoviana nel processo di diffusione in avanti possono essere modellate come distribuzioni gaussiane condizionate (Ho et al., 2020 [12]). Avvalendosi del formalismo matematico risulta che:

$$q(\mathbf{x}_{1:T}|\mathbf{x}_0) \triangleq \prod_{t=1}^T q(\mathbf{x}_t|\mathbf{x}_{t-1}) \quad \text{Processo di diffusione in avanti} \quad (3.1)$$

$$q(\mathbf{x}_t|\mathbf{x}_{t-1}) \triangleq \mathcal{N}(\mathbf{x}_t; \sqrt{1 - \beta_t} \mathbf{x}_{t-1}, \beta_t \mathbf{I}) \quad \text{Generica transizione} \quad (3.2)$$

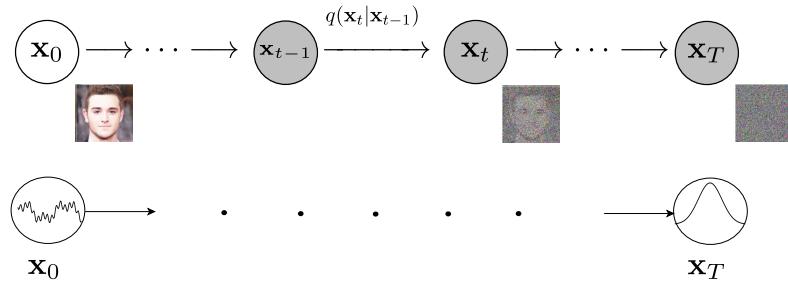
dove

- $t \in [1, \dots, T]$  è il generico passo temporale.
- $\{\beta_t \in (0, 1)\}_{t=1}^T$  implementano uno *schema di diffusione*.
- $\mathbf{x}_t$  è l’immagine a valle della  $t$ -esima transizione della diffusione in avanti.
- $\mathbf{x}_0$  è l’immagine originale di partenza che, dopo  $T$  passi, converge a  $\mathbf{x}_T$ .

La (3.2), in virtù dell’*artificio di riparametrizzazione* (si veda (A.25)), è equivalente ad asserire che

$$\mathbf{x}_t = \sqrt{1 - \beta_t} \mathbf{x}_{t-1} + \sqrt{\beta_t} \boldsymbol{\epsilon}_{t-1} \quad (3.3)$$

dove  $\boldsymbol{\epsilon}_{t-1} \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$ .



**Figura 3.1:** Processo di diffusione in avanti. Fonte [12].

*Osservazione.* Si osserva esplicitamente che, nella (3.2),  $\mathbf{x}_{t-1}$  è *fissato*: il vettore  $\sqrt{1 - \beta_t} \mathbf{x}_{t-1}$  è, quindi, *deterministico*. Pertanto, dalla (3.2) si evince che l'immagine  $\mathbf{x}_t$  è distribuita come una gaussiana condizionata con media  $\boldsymbol{\mu}_t = \sqrt{1 - \beta_t} \mathbf{x}_{t-1}$  e varianza  $\boldsymbol{\Sigma}_t = \beta_t \mathbf{I}$ .

Dalla (3.3) si desume *come* si ottiene l'immagine al passo temporale corrente  $\mathbf{x}_t$  a partire dall'immagine  $\mathbf{x}_{t-1}$  allo stato precedente: ad una versione scalata di quest'ultima si aggiunge rumore gaussiano scalato. Avendo l'accortezza di scegliere i  $\beta_t$  opportunamente, si perverrà ad un'immagine  $\mathbf{x}_T$  che, per  $T$  sufficientemente elevato, approssima una distribuzione gaussiana standard ed è, pertanto, indistinguibile dal rumore gaussiano puro [12].

*Osservazione.* Il passaggio dalla (3.2) alla (3.3), ricorrendo all'artificio di riparametrizzazione, sarà un *pattern* ricorrente nel prosieguo, in particolare nella riformulazione del processo di diffusione in avanti atta a pervenire ad una versione ottimizzata dello stesso.

*Osservazione.* Il numero totale degli step di diffusione  $T$  è un *iperparametro* e, in quanto tale, va fissato a monte della fase di addestramento del modello. Ho et al. (2020 [12]) hanno proposto per  $T = 1000$ .

*Osservazione.* È importante osservare che, poiché ad ogni transizione della catena markoviana in avanti viene aggiunto solo del rumore, il processo di diffusione in avanti risulta *fissato* qualora si scelgano i  $\beta_t$ .

Ricapitolando, il processo di diffusione in avanti trasforma, mediante l'aggiunta progressiva di rumore gaussiano, la complessa distribuzione incognita  $q(\mathbf{x}_0)$  iniziale in una distribuzione normale standardizzata  $\mathcal{N}(\mathbf{0}, \mathbf{I})$ . Il punto di forza di una siffatta distribuzione è insito nel suo essere facilmente manipolabile. La distribuzione terminale della diffusione in avanti costituisce il punto di partenza del processo di diffusione inversa.

### Implementazione del processo di diffusione in avanti

A beneficio di una maggiore ricchezza argomentativa, nonché per avere un riscontro pratico di quanto esposto precedentemente, viene riportata una possibile implementazione del processo di diffusione in avanti, partendo da una mia fotografia.

**Listing 3.1:** Codice adattato da [15] e implementato con Google Colab [10]

```

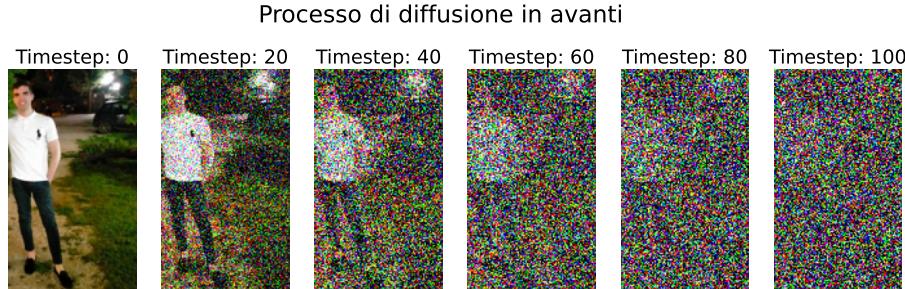
1 import numpy as np                      # per manipolare array e matrici
2 from PIL import Image                  # per manipolare le immagini
3 from matplotlib import pyplot as plt   # per creare e visualizzare grafici
4 from google.colab import files        # per scaricare l'immagine finale
5
6 def forward_diff_process(img_t_meno_1, beta, t):
7     """
8         Tale funzione implementa la generica transizione dall'immagine
9         al passo  $t-1$  all'immagine al passo  $t$  del processo di diffusione
10        in avanti.
11
12    Input:
13        img_t_meno_1: immagine al passo precedente ( $x_{t-1}$ )
14        beta: schema di diffusione (vettore di numeri)
15        t: passo corrente
16    Output:
17        img_t: immagine a valle della transizione ( $x_t$ )
18    """
19
20    # 1. Si ricava  $\beta_t$ 
21    beta_t = beta[t].reshape(-1, 1, 1)
22
23    # 2. Calcolo media statistica e deviazione standard
24    mu = np.sqrt((1.0 - beta_t)) * img_t_meno_1
25    sigma = np.sqrt(beta_t)
26
27    # 3. Generazione dell'immagine al passo  $t$  tramite l'equazione (3.3)
28    img_t = mu + sigma * np.random.randn(*img_t_meno_1.shape)
29    return img_t
30
31 # -----
32 # Esempio di applicazione del processo di diffusione in avanti
33 #
34
35 # 1. Si carica l'immagine di partenza  $x_0$ 
36 img = Image.open("../content/selfie.jpg")
37
38 # 2. Si ridimensiona l'immagine secondo le dimensioni desiderate
39 IMG_SIZE = (100, 172)
40 img = img.resize(size=IMG_SIZE)
41
42 # 3. Si definisce il numero dei passi temporali (T)
43 timesteps = 100
44
45 # 4. Implementazione schema di diffusione lineare
46 beta_start = 0.0001
47 beta_end = 0.05
48 beta = np.linspace(beta_start, beta_end, num=timesteps, dtype=np.float32)
49
50 immagini_processate = []
51 img_t = np.asarray(img.copy(), dtype=np.float32) / 255.

```

```

52
53 # 5. Esecuzione del processo di diffusione in avanti
54 # per ottenere l'immagine a valle di  $T = 100$  passi
55 for t in range(timesteps):
56     img_t = forward_diff_process(img_t_meno_1=img_t, beta=beta, t=t)
57     if t%20==0 or t==timesteps - 1:
58         immagine = (img_t.clip(0, 1) * 255.0).astype(np.uint8)
59         immagini_processate.append(immagine)
60
61 # 6. Visualizzazione delle immagini per diversi passi di diffusione
62 _, ax = plt.subplots(1, len(immagini_processate), figsize=(15, 8))
63
64 for i, immagine in enumerate(immagini_processate):
65     ax[i].imshow(immagine)
66     ax[i].set_title(f'Timestep: {i*20}', fontsize=18)
67     ax[i].axis("off")
68     ax[i].grid(False)
69
70 plt.suptitle("Processo di diffusione in avanti", fontsize=22, y=0.78)
71 plt.savefig("forward_diff.pdf", pad_inches=0, bbox_inches='tight')
72 files.download('forward_diff.pdf')
73 plt.show()
74 plt.close()

```



Si noti la progressiva perdita di connotati dell'immagine di partenza che, a valle di  $T$  passi di diffusione, risulta indistinguibile dal rumore puro.

### Ottimizzazione del processo di diffusione in avanti

Da un'attenta analisi del codice 3.1, in particolare dalla riga 55 alla riga 59, si evince che, per pervenire all'immagine terminale  $\mathbf{x}_T$ , è necessario simulare l'*intera* catena markoviana della diffusione in avanti: all'aumentare del numero di step totali  $T$  ciò risulta fortemente inefficiente. Per ovviare a tale problema, definendo preliminarmente

$$\alpha_t \triangleq 1 - \beta_t, \quad \bar{\alpha}_t \triangleq \prod_{i=1}^t \alpha_i \quad (3.4)$$

Ho et al. [12] hanno osservato che la (3.3) ammette la seguente riformulazione:

$$\mathbf{x}_t = \sqrt{1 - \beta_t} \mathbf{x}_{t-1} + \sqrt{\beta_t} \boldsymbol{\epsilon}_{t-1} \quad (3.5)$$

$$= \sqrt{\alpha_t} \mathbf{x}_{t-1} + \sqrt{1 - \alpha_t} \boldsymbol{\epsilon}_{t-1} \quad (3.6)$$

$$= \sqrt{\alpha_t} (\sqrt{\alpha_{t-1}} \mathbf{x}_{t-2} + \sqrt{1 - \alpha_{t-1}} \boldsymbol{\epsilon}_{t-2}) + \sqrt{1 - \alpha_t} \boldsymbol{\epsilon}_{t-1} \quad (3.7)$$

$$= \sqrt{\alpha_t \alpha_{t-1}} \mathbf{x}_{t-2} + \sqrt{\alpha_t(1 - \alpha_{t-1})} \boldsymbol{\epsilon}_{t-2} + \sqrt{1 - \alpha_t} \boldsymbol{\epsilon}_{t-1} \quad (3.8)$$

$$= \sqrt{\alpha_t \alpha_{t-1}} \mathbf{x}_{t-2} + \sqrt{1 - \alpha_t \alpha_{t-1}} \bar{\boldsymbol{\epsilon}}_{t-2} \quad (3.9)$$

$$= \dots \quad (3.10)$$

$$= \sqrt{\alpha_t \alpha_{t-1} \dots \alpha_1} \mathbf{x}_0 + \sqrt{1 - \alpha_t \alpha_{t-1} \dots \alpha_1} \boldsymbol{\epsilon}_0 \quad (3.11)$$

$$= \sqrt{\bar{\alpha}_t} \mathbf{x}_0 + \sqrt{1 - \bar{\alpha}_t} \boldsymbol{\epsilon}_0, \quad (3.12)$$

dove:

- $\{\bar{\boldsymbol{\epsilon}}_t, \boldsymbol{\epsilon}_t\}_{t=0}^T \stackrel{\text{iid}}{\sim} \mathcal{N}(\mathbf{0}, \mathbf{I})$ .
- dalla (3.6) alla (3.7) ci si è avvalsi dell’artificio di riparametrizzazione (A.25).
- il passaggio dalla (3.8) alla (3.9) è il combinato disposto dell’artificio di riparametrizzazione e del risultato per cui una combinazione lineare di vettori gaussiani *indipendenti* è ancora un vettore gaussiano (A.24). Infatti, poiché

$$\sqrt{1 - \alpha_t} \boldsymbol{\epsilon}_{t-1} \sim \mathcal{N}(\mathbf{0}, (1 - \alpha_t) \mathbf{I}) \quad (3.13)$$

$$\sqrt{\alpha_t(1 - \alpha_{t-1})} \boldsymbol{\epsilon}_{t-2} \sim \mathcal{N}(\mathbf{0}, (\alpha_t - \alpha_t \alpha_{t-1}) \mathbf{I}) \quad (3.14)$$

ed essendo  $\boldsymbol{\epsilon}_{t-1}$  ed  $\boldsymbol{\epsilon}_{t-2}$  *indipendenti*, la somma di (3.13) e (3.14) è distribuita come

$$\mathcal{N}(\mathbf{0}, (1 - \alpha_t + \alpha_t - \alpha_t \alpha_{t-1}) \mathbf{I}) = \mathcal{N}(\mathbf{0}, (1 - \alpha_t \alpha_{t-1}) \mathbf{I})$$

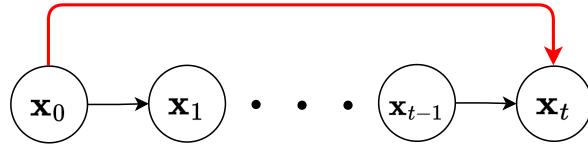
il che, ricorrendo all’artificio di riparametrizzazione, equivale a:

$$\sqrt{1 - \alpha_t} \boldsymbol{\epsilon}_{t-1} + \sqrt{\alpha_t(1 - \alpha_{t-1})} \boldsymbol{\epsilon}_{t-2} = \sqrt{1 - \alpha_t \alpha_{t-1}} \bar{\boldsymbol{\epsilon}}_{t-2}$$

Pertanto, il processo di diffusione in avanti può essere così riformulato:

$$q(\mathbf{x}_t | \mathbf{x}_0) = \mathcal{N}(\mathbf{x}_t; \sqrt{\bar{\alpha}_t} \mathbf{x}_0, (1 - \bar{\alpha}_t) \mathbf{I}) \quad (3.15)$$

La riformulazione (3.15) del processo di diffusione in avanti, basata essenzialmente sull’applicazione ricorsiva dell’artificio di riparametrizzazione, apporta un duplice beneficio:



**Figura 3.2:** Passaggio diretto da  $\mathbf{x}_0$  a  $\mathbf{x}_t$ . Fonte: [22].

- dall’immagine di partenza  $\mathbf{x}_0$  è possibile raggiungere un  $\mathbf{x}_t$  *qualsiasi* con un *unico* step di diffusione in avanti (Figura 3.2).
- per progettare uno *schema di diffusione* è possibile avvalersi degli  $\bar{\alpha}_t$ , in luogo dei  $\beta_t$  originali. Il vantaggio di ciò risiede nell’interpretazione di  $\bar{\alpha}_t$  come la varianza del segnale (l’immagine di partenza  $\mathbf{x}_0$ ) e  $1 - \bar{\alpha}_t$  come la varianza del rumore  $\epsilon$  [8].

Si riporta di seguito l’implementazione della versione ottimizzata del processo di diffusione in avanti, per corroborare la perfetta equivalenza sussistente tra le due formulazioni dello stesso, precedentemente investigate.

**Listing 3.2:** Diffusione in avanti ottimizzata. Codice adattato da [15]

```

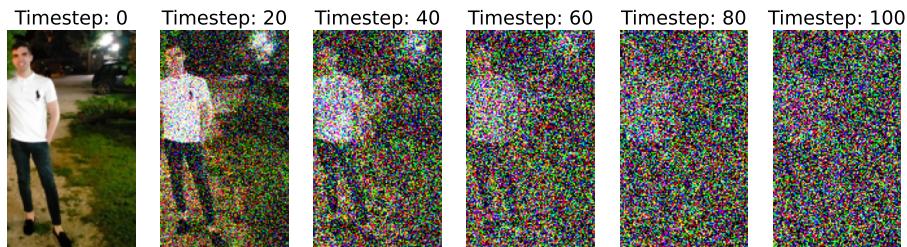
1 import numpy as np
2 from PIL import Image
3 from matplotlib import pyplot as plt
4 from google.colab import files
5
6 def forward_diff_process_ott(img_orig, alpha_bar, t):
7     """
8         Input:
9             img_orig: immagine al passo t=0 (x0)
10            alpha_bar: versione riparametrizzata di beta
11            t: passo corrente
12        Output:
13            img_t: immagine ottenuta al passo corrente (x_t)
14    """
15
16     # 1. Si ricava  $\bar{\alpha}_t$ 
17     alpha_bar_t = alpha_bar[t].reshape(-1, 1, 1)
18
19     # 2. Calcolo media statistica e deviazione standard
20     mu = np.sqrt(alpha_bar_t) * img_orig
21     sigma = np.sqrt(1.0 - alpha_bar_t)
22
23     # 3. Generazione dell’immagine al passo t tramite l’equazione (3.12)
24     img_t = mu + sigma * np.random.randn(*img_orig.shape)
25     return img_t
26
27
28     # 1. Si carica l’immagine di partenza (x0)
29     img = Image.open("../content/selfie.jpg")
30
31     # 2. Si ridimensiona l’immagine secondo le dimensioni desiderate
32     IMG_SIZE = (100, 172)
33     img = img.resize(size=IMG_SIZE)
  
```

```

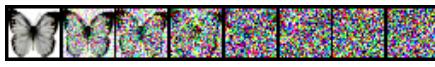
34
35 # 3. Si definisce il numero dei passi temporali (T)
36 timesteps = 100
37
38 #-----
39 # Versione ottimizzata del processo di diffusione in avanti
40 #-----
41
42 # 4. Implementazione schema di diffusione lineare
43 beta_start = 0.0001
44 beta_end = 0.05
45 beta = np.linspace(beta_start, beta_end, num=timesteps, dtype=np.float32)
46
47 # 5. Si definiscono alpha e alpha_bar secondo la (3.4)
48 alpha = 1.0 - beta
49 alpha_bar = np.cumprod(alpha)
50
51 immagini_processate = [img] # immagine x0
52 img_orig = np.asarray(img.copy(), dtype=np.float32) / 255.
53
54 # 6. Esecuzione del passo di diffusione in avanti per timestep specifici
55 # Si scelgono gli stessi timestep visualizzati nel codice 3.1
56 timestep_specifici = [19, 39, 59, 79, 99]
57 for step in timestep_specifici:
58     img_t = forward_diff_process_ott(img_orig, alpha_bar, step)
59     img_t = (img_t.clip(0, 1) * 255.0).astype(np.uint8)
60     immagini_processate.append(img_t)
61
62
63 # 7. Visualizzazione delle immagini in diversi passi di diffusione
64 _, ax = plt.subplots(1, len(immagini_processate), figsize=(15, 8))
65
66 for i, sample in enumerate(immagini_processate):
67     ax[i].imshow(sample)
68     ax[i].set_title(f"Timestep: {i*20}", fontsize=18)
69     ax[i].axis("off")
70     ax[i].grid(False)
71
72 plt.suptitle("Diffusione in avanti ottimizzata", fontsize=22, y=0.78)
73 plt.savefig("forward_process_v2.pdf", pad_inches=0, bbox_inches='tight')
74 files.download('forward_process_v2.pdf')
75 plt.show()
76 plt.close()

```

Diffusione in avanti ottimizzata



Si noti che, sebbene il risultato del codice 3.2 sia il medesimo del codice 3.1, il discriminio tra i due è insito nel *modo* in cui si perviene alle immagini nei diversi passi di diffusione.

| Noise Schedule | Visualization  |
|----------------|--|
| Linear         |  |
| Simple Linear  |  |
| Cosine         |  |
| Exponential    |  |
| Sigmoid        |  |

**Figura 3.3:** Diversi schemi di diffusione. Fonte: [1].

### Schemi di diffusione

Uno schema di diffusione si esplica in una particolare istanza del vettore<sup>1</sup>  $\beta = (\beta_1, \dots, \beta_T)^T$ , e controlla la quantità di rumore che viene aggiunta ad ogni passo della catena markoviana nel processo di diffusione in avanti.

In letteratura si ravvisano due approcci:

- si addestra una rete neurale ad apprendere i  $\beta_t$  che, in tal caso, sono considerati alla stregua di parametri ordinari della rete. Perseguendo tale approccio è possibile implementare schemi di diffusione *diversi* per la fase di addestramento e la fase di inferenza [1].
- i  $\beta_t$  assurgono ad iperparametri e, in quanto tali, sono fissati a priori (*hyperparameter tuning*). In tal caso, per progettare i  $\beta_t$ , in letteratura, si ricorre ad un ampio ventaglio di tecniche *euristiche* [1].

Ho et al. [12] hanno optato per quest'ultimo approccio, propendendo per uno schema di diffusione *lineare* in cui i  $\beta_t$  crescono linearmente da  $\beta_1 = 10^{-4}$  a  $\beta_T = 0.02$ .

Tuttavia, è stato dimostrato (Nichol et al., 2021 [16]) che l'impiego di uno schema di diffusione sinusoidale implica un incremento delle prestazioni del modello di diffusione. In Figura 3.3 sono riportati diversi schemi di diffusione.

Nel progettare uno schema di diffusione non si deve trascurare l'ingrediente fondamentale nel deep learning: i *dati*. Dal punto di vista della rappresentazione dei dati, la dimensionalità degli stessi e la massima distanza euclidea tra i campioni di addestramento sono fattori da contemplare nel design di uno

---

<sup>1</sup>Abuso di notazione: l'apice  $T$  è l'operazione di *trasposizione*, il pedice  $T$  è il numero totale degli step di diffusione.

schema di diffusione [20]. Inoltre, uno schema di diffusione deve tener conto della complessità e della ridondanza insita nei dati [2]. Ad esempio, immagini più grandi potrebbero richiedere più rumore additivo, a parità di passi di diffusione in avanti, rispetto ad immagini di dimensioni più piccole [16].

Il *tipo di rumore* aggiunto ad ogni passo del processo di diffusione in avanti è un altro iperparametro: Ho et al. [12] hanno proposto per rumore gaussiano  $\epsilon \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$ .

### 3.2.2 Processo di diffusione inversa

La magia dei modelli di diffusione avviene nel processo di diffusione inversa (*reverse diffusion process*). La *ratio* sottesa al processo di diffusione inversa è quella di rimuovere iterativamente rumore (*denoising*) dall'immagine  $\mathbf{x}_T \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$  (a cui si perviene a valle del processo di diffusione in avanti), per risalire alla distribuzione incognita dell'immagine di partenza  $q(\mathbf{x}_0)$  (Figura 3.4). Tuttavia per assolvere a tale scopo, il processo di diffusione inversa non può servirsi della distribuzione a posteriori  $q(\mathbf{x}_{t-1}|\mathbf{x}_t)$ . Infatti, ricorrendo alla legge di Bayes (A.7), risulta che

$$q(\mathbf{x}_{t-1}|\mathbf{x}_t) = \frac{q(\mathbf{x}_t|\mathbf{x}_{t-1})q(\mathbf{x}_{t-1})}{q(\mathbf{x}_t)} \quad (3.16)$$

da cui si evince come l'intrattabilità di  $q(\mathbf{x}_{t-1}|\mathbf{x}_t)$  sia imputabile alle distribuzioni marginali  $q(\mathbf{x}_{t-1})$  e  $q(\mathbf{x}_t)$  in quanto <sup>2</sup>:

$$q(\mathbf{x}_t) = \int q(\mathbf{x}_0, \mathbf{x}_1, \dots, \mathbf{x}_t) d\mathbf{x}_0 d\mathbf{x}_1 \dots d\mathbf{x}_{t-1} \quad (3.17)$$

$$= \int q(\mathbf{x}_1, \dots, \mathbf{x}_t|\mathbf{x}_0)q(\mathbf{x}_0) d\mathbf{x}_0 d\mathbf{x}_1 \dots d\mathbf{x}_{t-1} \quad (3.18)$$

dove  $q(\mathbf{x}_1, \dots, \mathbf{x}_t|\mathbf{x}_0)$  è la catena markoviana della diffusione in avanti definita dalla (3.1), e nella (3.17) si è effettuata l'operazione di marginalizzazione (A.4).

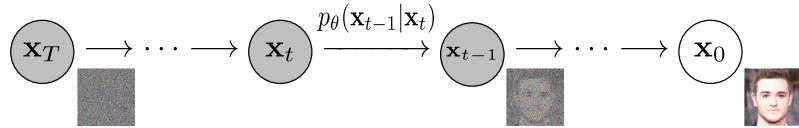
Dalla (3.18) si evince chiaramente che il problema sotteso al calcolo di  $q(\mathbf{x}_t)$  è duplice:

- la distribuzione  $q(\mathbf{x}_0)$  è *incognita*.
- l'integrale nella (3.18), essendo esteso ad uno spazio (di pixel) ad alta dimensionalità, è *intrattabile*.

Ho et al. [12], sulla scia di Sohl-Dickstein et al. [19], propongono l'addestramento di una rete neurale per approssimare  $q(\mathbf{x}_{t-1}|\mathbf{x}_t)$ .

---

<sup>2</sup>Risultato analogo sussiste *mutatis mutandis* per  $q(\mathbf{x}_{t-1})$



**Figura 3.4:** Processo di diffusione inversa. Fonte: [12].

Il processo di diffusione inversa, quindi, è definito come un catena markoviana di transizioni gaussiane i cui parametri (i.e. media e varianza) sono appresi da una rete neurale. L'assunzione di gaussianità per le suddette transizioni è supportata dall'osservazione che, qualora i  $\beta_t$  abbiano entità contenuta, le transizioni di entrambi i processi di diffusione (diretta e inversa) esibiscono la stessa forma funzionale [7]. Matematicamente risulta che:

$$p_{\boldsymbol{\theta}}(\mathbf{x}_{0:T}) \triangleq p(\mathbf{x}_T) \prod_{t=1}^T p_{\boldsymbol{\theta}}(\mathbf{x}_{t-1}|\mathbf{x}_t) \text{ Processo di diffusione inversa} \quad (3.19)$$

$$p_{\boldsymbol{\theta}}(\mathbf{x}_{t-1}|\mathbf{x}_t) \triangleq \mathcal{N}(\mathbf{x}_{t-1}; \boldsymbol{\mu}_{\boldsymbol{\theta}}(\mathbf{x}_t, t), \boldsymbol{\Sigma}_{\boldsymbol{\theta}}(\mathbf{x}_t, t)) \text{ Generica transizione} \quad (3.20)$$

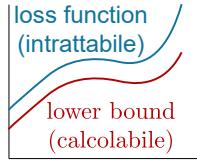
dove:

- $p(\mathbf{x}_T) = q(\mathbf{x}_T) = \mathcal{N}(\mathbf{x}_T; \mathbf{0}, \mathbf{I})$  dal momento che il punto di arrivo della diffusione in avanti costituisce la base di partenza del processo di diffusione inversa.
- il pedice  $\boldsymbol{\theta}$  denota il vettore dei *parametri* della rete neurale, aggiornati con la tecnica del gradiente discendente stocastico.
- $\boldsymbol{\mu}_{\boldsymbol{\theta}}(\mathbf{x}_t, t), \boldsymbol{\Sigma}_{\boldsymbol{\theta}}(\mathbf{x}_t, t)$  sono rispettivamente media e varianza che, come suggerisce la notazione (i.e il pedice  $\boldsymbol{\theta}$ ), sono appresi dalla rete neurale.

Ricapitolando, il processo di diffusione inversa consiste nel rimuovere iterativamente (anziché in un singolo passo come le GAN [1]), servendosi di una rete neurale, il rumore dall'immagine  $\mathbf{x}_T$ . Tale processo di diffusione inversa è imbastito sull'immagine terminale della diffusione in avanti e, al decrescere di  $t$  da  $T$  a 0, si propaga nella direzione opposta a quest'ultima tramite una catena markoviana di transizioni (3.20).

### 3.2.3 Addestramento

Da quanto detto nel paragrafo precedente, *si ricorre ad una rete neurale* per approssimare la media e la varianza delle transizioni (3.16) intrattabili della diffusione inversa.



**Figura 3.5:** *Lower bound* di una funzione. Fonte [18].

Nella scelta della funzione di costo da minimizzare per allenare la rete neurale, la *ratio* perseguita è la seguente. Dal momento che si possono ravvisare delle affinità tra il processo di diffusione inversa e il decoder di un *autoencoder variazionale* (VAE) (si veda B.1), è legittimo avvalersi della medesima funzione di costo usata nei VAE: il logaritmo della funzione di verosimiglianza cambiato di segno (*Negative Log-Likelihood*) (A.2.2). Pertanto, la funzione di costo da minimizzare è:

$$L = -\log(p_{\theta}(\mathbf{x}_0)) \quad (3.21)$$

Tuttavia, esplicitando la funzione di verosimiglianza  $p_{\theta}(\mathbf{x}_0)$

$$p_{\theta}(\mathbf{x}_0) = \int p_{\theta}(\mathbf{x}_{0:T}) d\mathbf{x}_{1:T} \quad (3.22)$$

si desume che  $p_{\theta}(\mathbf{x}_0)$  (e quindi  $L$ ) è *intrattabile*, essendo l'integrale nella (3.22) esteso ad uno spazio (di pixel) ad alta dimensionalità. Si ricorre, quindi, al tipico *pattern* di approssimare una funzione intrattabile con un suo *lower bound* (Figura 3.5) che ammetta, invece, un'espressione in forma chiusa. Invocando, ancora una volta, l'affinità intercorrente tra il processo di diffusione inversa e il decoder di un VAE, si può ricorrere al *lower bound varazionale* (VLB) (o *lower bound dell'evidenza* (ELBO)) che, declinato al contesto dei modelli di diffusione, è:

$$\log(p_{\theta}(\mathbf{x}_0)) \geq \underbrace{\mathbb{E}_q \left[ \log \frac{p_{\theta}(\mathbf{x}_{0:T})}{q(\mathbf{x}_{1:T}|\mathbf{x}_0)} \right]}_{VLB} \quad (3.23)$$

(si veda Appendice B per la derivazione dettagliata), e, come si vedrà nel prosieguo, risulterà essere trattabile. Pertanto, minimizzare la funzione di costo  $L$  significa minimizzare il VLB cambiato di segno:

$$L = -\log(p_{\theta}(\mathbf{x}_0)) \leq L_{vlb} \triangleq \mathbb{E}_q \left[ -\log \frac{p_{\theta}(\mathbf{x}_{0:T})}{q(\mathbf{x}_{1:T}|\mathbf{x}_0)} \right] \quad (3.24)$$

*Osservazione.* A rigore l'uso del pedice *vlb* in  $L_{vlb}$  è un abuso di notazione, dal momento che  $L_{vlb}$  non è un lower bound ma è un *upper bound* della funzione di costo  $L$ . Tuttavia, si predilige tale notazione  $L_{vlb}$  per essere consistenti con la letteratura.

A valle di una serie di calcoli che sono riportati in Appendice B si perviene al seguente risultato:

$$L_{vbl} = L_0 + \sum_{t=2}^T L_{t-1} + L_T \quad (3.25)$$

dove:

- $L_0 = -\log(p_{\theta}(\mathbf{x}_0|\mathbf{x}_1))$ . Ho et al. [12] hanno delegato ad un apposito decoder l'approssimazione di tale termine che, pertanto, può essere trascurato nella fase di addestramento della rete neurale.
- $L_{t-1} = D_{KL}(q(\mathbf{x}_{t-1}|\mathbf{x}_t, \mathbf{x}_0) \| p_{\theta}(\mathbf{x}_{t-1}|\mathbf{x}_t))$ . L'obiettivo è quello di addestrare la rete neurale, cosicché  $p_{\theta}(\mathbf{x}_{t-1}|\mathbf{x}_t)$  approssimi quanto più fedelmente la distribuzione  $q(\mathbf{x}_{t-1}|\mathbf{x}_t, \mathbf{x}_0)$  in modo da minimizzare il termine  $L_{t-1}$  (ricorrendo all'usuale interpretazione della *divergenza di Kullback-Leibler* (Appendice B, B.20) come una misura della distanza tra le distribuzioni).
- $L_T = D_{KL}(q(\mathbf{x}_T|\mathbf{x}_0) \| p(\mathbf{x}_T))$  è un termine che può essere *ignorato* durante la fase di addestramento, dal momento che non annovera alcun parametro che possa essere appreso dalla rete neurale. Infatti  $p(\mathbf{x}_T) = q(\mathbf{x}_T) \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$  e  $q(\mathbf{x}_T|\mathbf{x}_0)$  è fissato una volta scelti gli iperparametri  $\beta_t$  (3.15).

*Osservazione.* Definendo uno schema di diffusione opportuno (i.e. i valori dei  $\beta_t$ ) e scegliendo l'iperparametro  $T$  sufficientemente elevato, entrambe le distribuzioni  $q(\mathbf{x}_T|\mathbf{x}_0)$  e  $p(\mathbf{x}_T)$  approssimano una gaussiana standard. Quindi, stante l'interpretazione della divergenza di Kullback-Leibler come una misura della distanza tra tali distribuzioni, il termine  $L_T = D_{KL}(q(\mathbf{x}_T|\mathbf{x}_0) \| p(\mathbf{x}_T))$  tende a zero.

Dunque, affinché  $L_{vbl}$  risulti trattabile è sufficiente che  $L_{t-1}$  ammetta un'espressione in forma chiusa.

Restringendo l'attenzione al termine  $L_{t-1}$ , in Appendice B si mostra che sussistono le seguenti considerazioni:

- poiché:

$$q(\mathbf{x}_{t-1}|\mathbf{x}_t, \mathbf{x}_0) = \mathcal{N}(\mathbf{x}_{t-1}; \tilde{\boldsymbol{\mu}}_t(\mathbf{x}_t, \mathbf{x}_0), \tilde{\beta}_t \mathbf{I}) \quad (3.26)$$

$$p_{\theta}(\mathbf{x}_{t-1}|\mathbf{x}_t) = \mathcal{N}(\mathbf{x}_{t-1}; \boldsymbol{\mu}_{\theta}(\mathbf{x}_t, t), \boldsymbol{\Sigma}_{\theta}(\mathbf{x}_t, t)) \quad (3.27)$$

segue che  $L_{t-1} = D_{KL}(q(\mathbf{x}_{t-1}|\mathbf{x}_t, \mathbf{x}_0) \| p_{\theta}(\mathbf{x}_{t-1}|\mathbf{x}_t))$  è la divergenza di Kullback-Leibler tra due gaussiane, e in quanto tale può essere espresso in forma chiusa.

- nella (3.27), sebbene in linea di principio si possa addestrare la rete neurale ad apprendere sia  $\mu_\theta(\mathbf{x}_t, t)$  che  $\Sigma_\theta(\mathbf{x}_t, t)$ , Ho et al. [12] hanno fissato la varianza  $\Sigma_\theta(\mathbf{x}_t, t) = \beta_t \mathbf{I}$ . Quindi nella fase di addestramento la rete neurale deve apprendere la sola media  $\mu_\theta(\mathbf{x}_t, t)$ .
- anziché addestrare la rete neurale ad apprendere, in ogni passo di diffusione inversa, la media  $\mu_\theta(\mathbf{x}_t, t)$ , si mostra che la rete può equivalentemente essere addestrata a predire il rumore  $\epsilon_\theta(\mathbf{x}_t, t)$  che deve essere rimosso in ogni step della diffusione inversa per giungere all'immagine di partenza.

Stante i suddetti risultati, in Appendice B si mostra che ognuno degli  $L_{t-1}$  nella (3.25) assume la seguente forma semplificata:

$$L_{t-1}^{\text{simple}} = \|\boldsymbol{\epsilon} - \underbrace{\epsilon_\theta(\sqrt{\bar{\alpha}_t} \mathbf{x}_0 + \sqrt{1 - \bar{\alpha}_t} \boldsymbol{\epsilon}, t)}_{\mathbf{x}_t}\|^2 + C \quad (3.28)$$

dove  $C$  è una costante che non dipende da  $\theta$  e, quindi, può essere ignorata durante l'addestramento.

In definitiva, partendo da una funzione di costo  $L = -\log(p_\theta(\mathbf{x}_0))$  intrattabile si è giunti alla (3.28), in cui ognuno degli  $L_{t-1}$  è, banalmente, l'*errore quadratrico medio* tra il rumore  $\boldsymbol{\epsilon}$ , aggiunto nella diffusione in avanti, e il rumore *stimato* della rete neurale  $\epsilon_\theta(\mathbf{x}_t, t)$ .

### Architettura della rete neurale e algoritmo di addestramento

Dal momento che la rete neurale deve fornire in uscita un'immagine (i.e. la stima del rumore), che è un *tensore* avente le stesse dimensioni dell'immagine rumorosa in ingresso alla rete, Ho et al. [12] hanno implementato la suddetta rete neurale con una U-Net, in virtù della sua peculiarità di produrre un'immagine con le medesime dimensioni dell'immagine in ingresso. In Appendice C sono riportati ulteriori dettagli sulla U-Net utilizzata in un DDPM.

---

#### Algoritmo 1 Addestramento. Fonte [12]

---

```

1: repeat
2:    $\mathbf{x}_0 \sim q(\mathbf{x}_0)$ 
3:    $t \sim \text{Uniform}(\{2, \dots, T\})$ 
4:    $\boldsymbol{\epsilon} \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$ 
5:    $\mathbf{x}_t = \sqrt{\bar{\alpha}_t} \mathbf{x}_0 + \sqrt{1 - \bar{\alpha}_t} \boldsymbol{\epsilon}$ 
6:   Take gradient descent step on
7:      $\nabla_\theta \|\boldsymbol{\epsilon} - \epsilon_\theta(\mathbf{x}_t, t)\|^2$ 
8: until converged

```

---

For each training step:

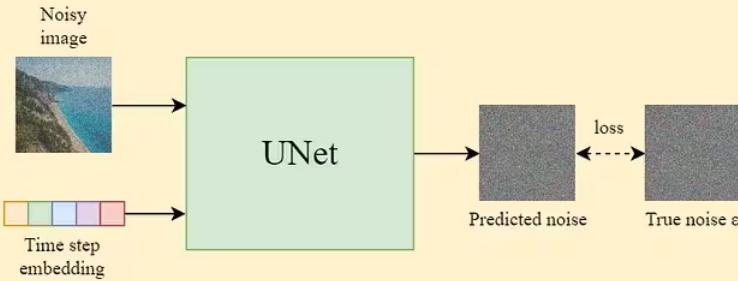
1. Randomly select a time step & encode it



2. Add noise to image

A diagram illustrating the addition of Gaussian noise. It shows three images: a "Noisy image" (a photo of a coastal scene), an "Original image" (the same scene), and a "Gaussian noise" (a dark gray square). Below them is the equation:  $x_t = \sqrt{\bar{a}_t} x_0 + \sqrt{1 - \bar{a}_t} \varepsilon$ . To the right, the parameters are defined:  $\varepsilon \sim \mathcal{N}(0, 1)$ ,  $\alpha_t = 1 - \beta_t$ , and  $\bar{\alpha}_t = \prod_{i=1}^t \alpha_i$ .

3. Train the UNet



**Figura 3.6:** Addestramento della rete neurale. Fonte [18].

In Figura 3.6 è illustrato il meccanismo sotteso ad un passo dell'algoritmo di addestramento (Algoritmo 1) della rete neurale. Nell'Algoritmo 1:

- si considera un'immagine  $\mathbf{x}_0$ , nel dataset di addestramento, caratterizzata da una distribuzione incognita e arbitrariamente complessa  $q(\mathbf{x}_0)$  (riga 2).
- si seleziona *a caso* (ciò giustifica il ricorso alla distribuzione uniforme) un passo temporale  $t$  (riga 3).
- si campiona del rumore  $\varepsilon$  da una gaussiana standard (riga 4).

- si applica il processo di diffusione in avanti nella sua versione ottimizzata (3.12), per ottenere l'immagine rumorosa  $\mathbf{x}_t$  a partire da  $\mathbf{x}_0$  (riga 5).
- si minimizzano i termini  $L_{t-1}^{\text{simple}} = \|\boldsymbol{\epsilon} - \boldsymbol{\epsilon}_{\theta}(\mathbf{x}_t, t)\|^2$  con la tecnica del gradiente discendente stocastico (riga 7).

*Osservazione.* Si noti come la riformulazione (3.12) del processo di diffusione in avanti permetta di scegliere *a caso* il passo temporale  $t$ , e quindi il termine  $L_{t-1}$  da ottimizzare, con il gradiente discendente, nella fase di addestramento.

### 3.2.4 Generazione di immagini

Ultimato l'addestramento della U-Net, si possono sfruttare le transizioni (3.20) della catena markoviana della diffusione inversa per rimuovere iterativamente il rumore.

In particolare, poiché la versione approssimata  $\boldsymbol{\mu}_{\theta}(\mathbf{x}_t, t)$ , dalla rete neurale, della media (B.30)  $\tilde{\boldsymbol{\mu}}_t(\mathbf{x}_t)$  è:

$$\boldsymbol{\mu}_{\theta}(\mathbf{x}_t, t) = \frac{1}{\sqrt{\alpha_t}} \left( \mathbf{x}_t - \frac{1 - \alpha_t}{\sqrt{1 - \bar{\alpha}_t}} \boldsymbol{\epsilon}_{\theta}(\mathbf{x}_t, t) \right) \quad (3.29)$$

combinando la (3.29) con la (3.20), e sfruttando l'artificio di riparametrizzazione (A.25), risulta che

$$\mathbf{x}_{t-1} = \frac{1}{\sqrt{\alpha_t}} \left( \mathbf{x}_t - \frac{1 - \alpha_t}{\sqrt{1 - \bar{\alpha}_t}} \boldsymbol{\epsilon}_{\theta}(\mathbf{x}_t, t) \right) + \sqrt{\beta_t} \boldsymbol{\epsilon} \quad (3.30)$$

dove  $\boldsymbol{\epsilon}$  è una gaussiana standard. Applicando ricorsivamente la formula (3.30) è possibile risalire, partendo da  $\mathbf{x}_T$  (che è indistinguibile dal rumore puro), all'immagine di partenza  $\mathbf{x}_0$  (Figura 3.7).

*Osservazione.* A scopo puramente chiarificatore, si consideri, ad esempio, il passo  $t = 50$ . Nella diffusione inversa il rumore  $\boldsymbol{\epsilon}_{\theta}(\mathbf{x}_{50}, 50)$ , stimato dalla Unet, è il rumore che sottratto a  $\mathbf{x}_{50}$  restituisce  $\mathbf{x}_0$  non  $\mathbf{x}_{49}$ . Tuttavia, da un'attenta analisi della formula (3.30), ciò che si sottrae a  $\mathbf{x}_{50}$ , per ottenere  $\mathbf{x}_{49}$ , è una versione scalata, del fattore  $(1 - \alpha_t)/\sqrt{1 - \bar{\alpha}_t}$ , di tale rumore.

*Osservazione.* Avendo Ho et al. [12] scelto uno schema di diffusione lineare, cosicché i  $\beta_t$  si incrementano linearmente, e considerando le definizioni (3.4) di  $\alpha_t$  e  $\bar{\alpha}_t$ , dalla formula (3.30) si evince che il primo passo della diffusione inversa sottrae da  $\mathbf{x}_T$  soltanto una piccola parte del rumore  $\boldsymbol{\epsilon}_{\theta}(\mathbf{x}_t, t)$  stimato dalla U-Net. Viceversa, i passi di diffusione inversa successivi sottraggono da  $\mathbf{x}_t$  aliquote del rumore stimato dalla rete neurale via via sempre più grandi.

## Reverse Diffusion / Denoising / Sampling

## 1. Sample a Gaussian noise

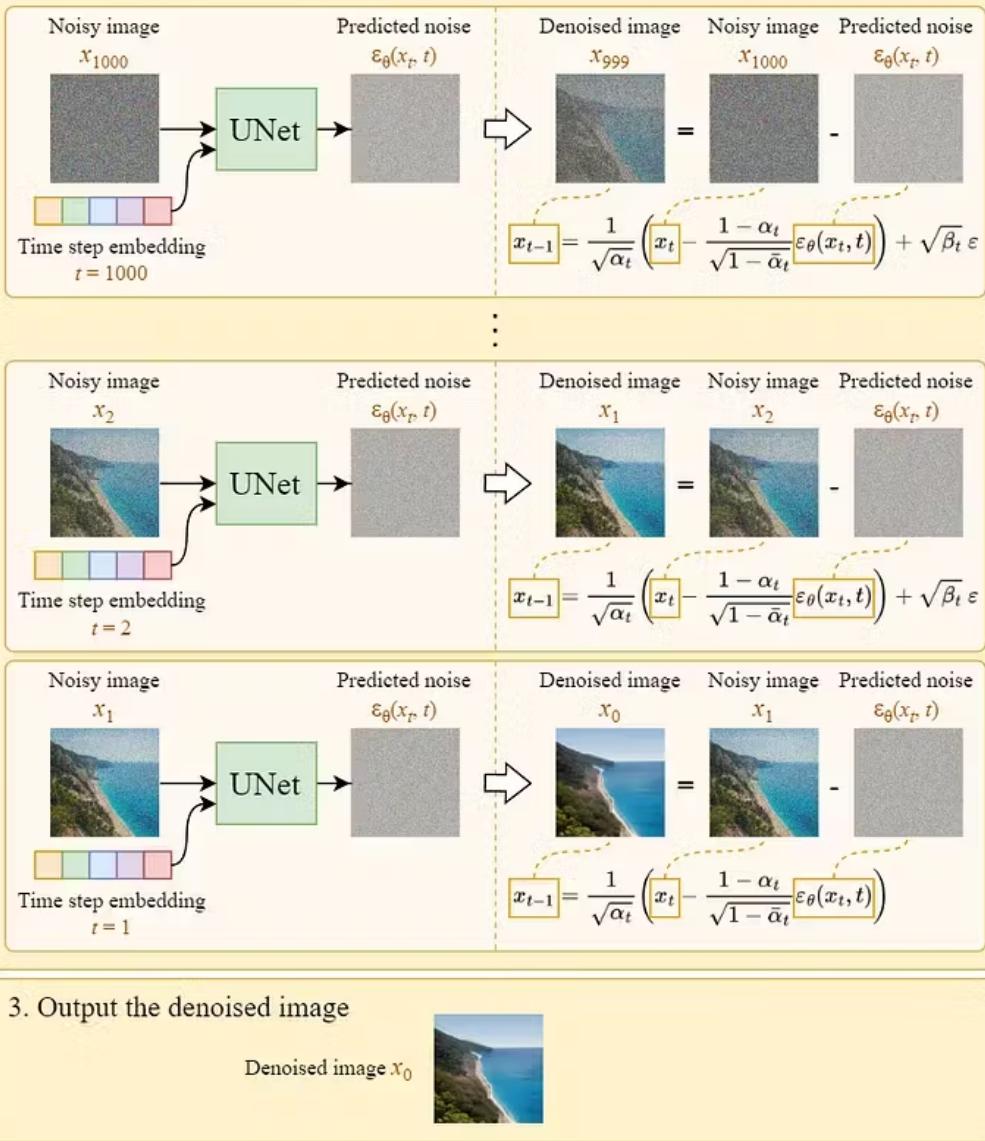
$$x_T \sim N(0, I)$$

E.g.  $T = 1000$ 

$$x_{1000} \sim N(0, I)$$



## 2. Iteratively denoise the image



**Figura 3.7:** Generazione di immagini con il processo di diffusione inversa.  
Fonte [18].

---

**Algoritmo 2** Generazione di immagini. Fonte [12]

---

```

1:  $\mathbf{x}_T \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$ 
2: for  $t = T, \dots, 1$  do
3:    $\epsilon \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$  if  $t > 1$ , else  $\epsilon = \mathbf{0}$ 
4:    $\mathbf{x}_{t-1} = \frac{1}{\sqrt{\alpha_t}} \left( \mathbf{x}_t - \frac{1-\alpha_t}{\sqrt{1-\alpha_t}} \epsilon_{\theta}(\mathbf{x}_t, t) \right) + \sqrt{\beta_t} \epsilon$ 
5: end for
6: return  $\mathbf{x}_0$ 

```

---

### 3.2.5 Prestazioni

A conclusione di tale capitolo vengono riportate le prestazioni del DDPM implementato da Ho et al. [12], misurate ricorrendo a due notorie metriche, ampiamente usate in letteratura, per valutare la qualità delle immagini prodotte da modelli generativi. Entrambe le due suddette metriche si avvalgono di una rete *Inception-v3* [21], un classificatore di immagini pre-addestrato su ImageNet<sup>3</sup>. In particolare:

- la metrica IS (*Inception Score*) valuta la sola distribuzione delle immagini generate dal modello: ad un punteggio IS elevato corrisponde una migliore *qualità* e *varietà* delle immagini generate dal modello.
- la metrica FID (*Fréchet Inception Distance*) confronta le distribuzioni delle immagini generate dal modello con quelle delle immagini del dataset di addestramento. Tale metrica, dal momento che “*cattura la somiglianza delle immagini generate con quelle reali, meglio di quanto non faccia la metrica IS*” [11], è lo standard *de facto* per valutare la qualità delle immagini prodotte dai modelli generativi. Più basso è il punteggio FID, più la distribuzione delle immagini generate si avvicina alla distribuzione delle immagini del dataset di addestramento.

Come già accennato in precedenza, Ho et al. [12] hanno scelto  $T = 1000$  e hanno proposto, nel processo di diffusione in avanti, per uno schema di diffusione *lineare*, con i  $\beta_t$  che si incrementano linearmente da  $\beta_1 = 10^{-4}$  a  $\beta_T = 0.02$ . Gli iperparametri  $\beta_t$ , che implementano lo schema di diffusione, sono stati scelti molto più piccoli rispetto ai valori dei pixel delle immagini di addestramento normalizzati tra  $[-1, 1]$ , cosicché le transizioni delle catene markoviane dei processi di diffusione diretta e inversa esibiscano approssimativamente la medesima forma funzionale (distribuzione gaussiana) mantenendo, al contempo, il rapporto segnale rumore a valle della diffusione in avanti,

---

<sup>3</sup><https://www.image-net.org/>

| Model                                   | IS                                | FID         |
|---|-----------------------------------|-------------|
| <b>Conditional</b>                      |                                   |             |
| EBM [11]                                | 8.30                              | 37          |
| JEM [17]                                | 8.76                              | 38.4        |
| BigGAN [3]                              | 9.22                              | 14.73       |
| StyleGAN2 + ADA (v1) [29]               | <b>10.06</b>                      | <b>2.67</b> |
| <b>Unconditional</b>                    |                                   |             |
| Gated PixelCNN [59]                     | 4.60                              | 65.93       |
| PixelIQN [43]                           | 5.29                              | 49.46       |
| EBM [11]                                | 6.78                              | 38.2        |
| NCSNv2 [56]                             |                                   | 31.75       |
| NCSN [55]                               | $8.87 \pm 0.12$                   | 25.32       |
| SNGAN [39]                              | $8.22 \pm 0.05$                   | 21.7        |
| SNGAN-DDLS [4]                          | $9.09 \pm 0.10$                   | 15.42       |
| StyleGAN2 + ADA (v1) [29]               | <b><math>9.74 \pm 0.05</math></b> | 3.26        |
| Ours ( $L$ , fixed isotropic $\Sigma$ ) | $7.67 \pm 0.13$                   | 13.51       |
| <b>Ours</b> ( $L_{\text{simple}}$ )     | $9.46 \pm 0.11$                   | <b>3.17</b> |

**Figura 3.8:** Prestazioni DDPM. Fonte [12]

il più contenuto possibile ( $L_T = D_{KL}(q(\mathbf{x}_T|\mathbf{x}_0) \parallel \mathcal{N}(\mathbf{0}, \mathbf{I})) \approx 10^{-5}$  bit per dimensione [12]).

Come si evince dalla Figura 3.8, con un punteggio FID<sup>4</sup> di 3.17, il modello incondizionato addestrato da Ho et al. [12] sul dataset CIFAR-10<sup>5</sup> ottiene una migliore qualità dei campioni rispetto alla maggior parte dei modelli in letteratura, inclusa la classe dei modelli condizionali.

---

<sup>4</sup>i punteggi IS e FID riportati in Figura 3.8 sono stati calcolati su 50000 campioni del dataset di addestramento CIFAR-10.

<sup>5</sup>Il dataset CIFAR-10 consiste di 60000 immagini  $32 \times 32$  a colori suddivise in 10 classi, con 6000 immagini per classe. Tale dataset annovera 50000 immagini per l'addestramento e 10000 per la fase di testing.

# Capitolo 4

## Conclusioni

In questa trattazione si è dettagliato il meccanismo sotteso ai modelli probabilistici di diffusione del rumore (DDPM), che hanno rappresentato un’innovazione nel contesto della modellazione generativa, rivaleggiando con le GAN (Dhariwal e Nichol, 2021 [6]) nella generazione di immagini. Dalla discussione del capitolo precedente emerge che il punto di forza dei DDPM è insito nell’alto grado di flessibilità offerto dagli stessi: l’unico requisito è che l’input e l’output della U-Net (Appendice C) abbiano la stessa dimensionalità.

L’articolo pionieristico di Ho et al. (2020, [12]) ha gettato le basi teoretiche per lo sviluppo di tecnologie quali Stable Diffusion<sup>1</sup>, Imagen<sup>2</sup>, DALL-E 2<sup>3</sup>, il cui funzionamento sottende l’impiego di varianti del modello di diffusione descritto in questa tesi.

Sebbene le suddette tecnologie abbiano rivoluzionato l’ambito della generazione di immagini mediante l’uso dell’intelligenza artificiale, esse sollevano anche importanti implicazioni etiche.

Un primo problema risiede nel fatto che i modelli generativi di cui si avvalgono tali tecnologie riflettono i pregiudizi (*bias*) impliciti nelle immagini del dataset di addestramento. Eloquente, a tal proposito, è la notizia, risalente al settembre 2022, in cui si riporta che “*OpenAI ha confermato a The Verge che DALL-E inserisce silentemente frasi nei prompt degli utenti per mitigare l’impatto dei bias insiti nelle immagini del dataset; ad esempio, le frasi «uomo di colore» e «donna asiatica» vengono inserite nei prompt che non specificano sesso o razza”* [4].

Infine, un’altra questione, molto delicata sotto il profilo etico, è il potenziale uso improprio di immagini generate dalle suddette tecnologie. In Figura 4.1a è riportata l’immagine, successivamente rimossa, pubblicata da *Amnesty*

---

<sup>1</sup><https://stability.ai/>

<sup>2</sup><https://imagen.research.google/>

<sup>3</sup><https://openai.com/dall-e-2>



- (a) Immagini pubblicate da *Amnesty International*.  
Fonte: <https://twitter.com>.

- (b) Immagini reperibili nel servizio immagini stock di *Adobe*.  
Fonte: [23].

**Figura 4.1:** Immagini create dall'intelligenza artificiale.

*International* sul social media X, delle proteste colombiane del 2021, mentre nel servizio di immagini stock di *Adobe* si possono reperire foto, generate ricorrendo all'IA, concernenti il conflitto israelo-palestinese (Figura 4.1b).

Benché le immagini in Figura 4.1 siano corredate dall'informazione di essere state create avvalendosi dell'intelligenza artificiale, qualora, ad esempio, venissero immesse nel circuito dei mezzi d'informazione, sprovviste di segnalazioni esplicite sulla loro natura (da un'attenta analisi della Figura 4.1a risaltano i *colori invertiti della bandiera colombiana*), potrebbero essere impiegate per plagiare, subdolamente, il segmento con meno attitudine allo spirito critico dell'opinione pubblica. Ma questa è un'altra storia.

# Appendice A

## Nozioni di probabilità e statistica

In tale appendice vengono richiamati concetti di teoria della probabilità (Paragrafo A.1) e di statistica (Paragrafo A.2), ricorrenti nei capitoli di questa trattazione.

### A.1 Elementi di teoria della probabilità

#### A.1.1 Distribuzioni di probabilità

Si consideri un vettore aleatorio<sup>1</sup>  $\mathbf{X} = [X_1, X_2, \dots, X_N]^T$ .

**Definizione** (CDF congiunta). Si definisce *funzione di distribuzione cumulativa* (CDF) congiunta di  $\mathbf{X}$  la funzione:

$$F_{\mathbf{X}}(\mathbf{x}): \mathbf{x} \in \mathbb{R}^N \longrightarrow P(\{X_1 \leq x_1\}, \dots, \{X_N \leq x_N\}) \quad (\text{A.1})$$

Se le componenti del vettore  $\mathbf{X}$  sono tutte *v.a. discrete*, allora la caratterizzazione viene spesso effettuata introducendo la *funzione masse di probabilità* (*pmf*) congiunta.

**Definizione** (*pmf* congiunta).

$$p_{\mathbf{X}}(\mathbf{x}): \mathbf{x} = (x_1, \dots, x_N) \in \prod_{i=1}^N \mathcal{A}_i \longrightarrow P\left(\bigcap_{i=1}^N \{X_i = x_i\}\right) \quad (\text{A.2})$$

dove  $\mathcal{A}_i$  è l'alfabeto della componente  $i$ -esima.

---

<sup>1</sup>T è l'operatore di *trasposizione*

Analogamente, nel caso in cui le componenti di  $\mathbf{X}$  siano *v.a. continue* si ricorre alla seguente caratterizzazione alternativa.

**Definizione** (*pdf* congiunta). Si definisce *pdf* congiunta di  $\mathbf{X}$  la derivata mista, di ordine  $N$ , della CDF congiunta rispetto a tutte le variabili:

$$f_{\mathbf{X}}(\mathbf{x}) \triangleq \frac{\partial^N F_{\mathbf{X}}(x_1, x_2, \dots, x_N)}{\partial x_1 \dots \partial x_N} \quad (\text{A.3})$$

*Osservazione* (Marginalizzazione). Dalla *pdf* congiunta di un vettore aleatorio si può desumere la *pdf* congiunta di un qualsiasi sottovettore, in particolare le *pdf* delle singole variabili aleatorie (*pdf* marginali). Risulta che:

$$f_{X_i}(x_i) = \int_{-\infty}^{+\infty} f_{\mathbf{X}}(x_1, \dots, x_N) dx_1 \dots, dx_{i-1}, dx_{i+1}, \dots, dx_N \quad (\text{A.4})$$

La *ratio* è quella di integrare la *pdf* congiunta rispetto alle variabili che non interessano ai fini del calcolo della *pdf* marginale.

*Osservazione* (Notazione). Nel prosieguo si utilizzerà la dizione “*distribuzione*” e la notazione  $p_{\mathbf{X}}(\mathbf{x})$  per riferirsi sia alla *pmf* che alla *pdf* indistintamente. Coerentemente con la maggior parte della letteratura, si delega al contesto l’attribuzione della corretta *semantica* alla notazione.

## Distribuzioni condizionate

**Definizione.** Siano  $\mathbf{X}$  e  $\mathbf{Y}$  due vettori aleatori. Risulta che:

$$p_{\mathbf{X}|\mathbf{Y}}(\cdot|\mathbf{y}): \mathbf{x} \in \mathbb{R}^N \longrightarrow \frac{p_{\mathbf{XY}}(\mathbf{x}, \mathbf{y})}{p_{\mathbf{Y}}(\mathbf{y})} \quad (\text{A.5})$$

Di seguito vengono riportate la regola della catena (*chain rule*) (A.6) e la *legge di Bayes* (A.7):

$$p_{\mathbf{XY}}(\mathbf{x}, \mathbf{y}) = p_{\mathbf{X}|\mathbf{Y}}(\mathbf{x}|\mathbf{y})p_{\mathbf{Y}}(\mathbf{y}) = p_{\mathbf{Y}|\mathbf{X}}(\mathbf{y}|\mathbf{x})p_{\mathbf{X}}(\mathbf{x}) \quad (\text{A.6})$$

$$p_{\mathbf{X}|\mathbf{Y}}(\mathbf{x}|\mathbf{y}) = p_{\mathbf{Y}|\mathbf{X}}(\mathbf{y}|\mathbf{x}) \frac{p_{\mathbf{X}}(\mathbf{x})}{p_{\mathbf{Y}}(\mathbf{y})} \quad (\text{A.7})$$

### A.1.2 Caratterizzazione sintetica di vettori aleatori

#### Media statistica

**Definizione.** Si definisce *media statistica* di un vettore aleatorio  $\mathbf{X}$ :

$$\mathbb{E}[\mathbf{X}] \triangleq \begin{bmatrix} \mathbb{E}[X_1] \\ \vdots \\ \mathbb{E}[X_N] \end{bmatrix} \in \mathbb{R}^N \quad (\text{A.8})$$

dove

$$\mathbb{E}[X_i] \triangleq \begin{cases} \int_{-\infty}^{+\infty} x_i p(x_i) dx_i & \text{se } X_i \text{ è una v.a. continua} \\ \sum_{x_j \in \mathcal{A}_{X_i}} x_j p(x_i = x_j) & \text{se } X_i \text{ è una v.a. discreta} \end{cases} \quad (\text{A.9})$$

#### Covarianza e varianza

**Definizione** (Covarianza). La *covarianza* tra due vettori aleatori  $\mathbf{X}, \mathbf{Y}$  è così definita:

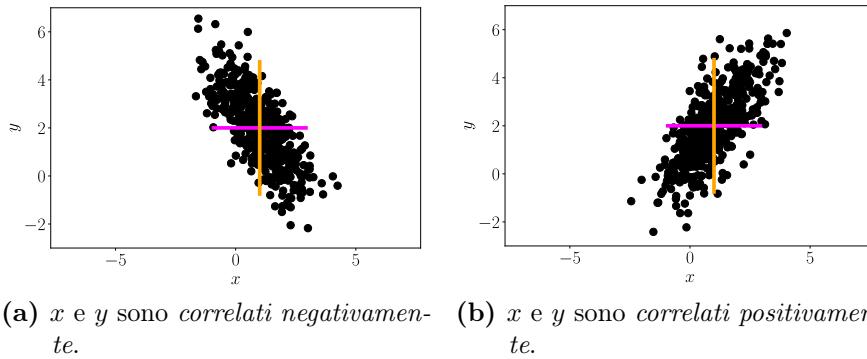
$$\text{Cov}[\mathbf{X}, \mathbf{Y}] \triangleq \mathbb{E}[(\mathbf{X} - \boldsymbol{\mu}_{\mathbf{X}})(\mathbf{Y} - \boldsymbol{\mu}_{\mathbf{Y}})^T] \quad (\text{A.10})$$

dove  $\boldsymbol{\mu}_{\mathbf{X}} = \mathbb{E}[\mathbf{X}]$  e  $\boldsymbol{\mu}_{\mathbf{Y}} = \mathbb{E}[\mathbf{Y}]$ .

Particolarizzando la suddetta definizione al caso in cui si consideri lo stesso vettore aleatorio  $\mathbf{X}$  in entrambi gli argomenti della covarianza (A.10), si ottiene la *varianza* di  $\mathbf{X}$ .

**Definizione** (Varianza). Sia  $\mathbf{X}$  un vettore aleatorio avente media  $\boldsymbol{\mu}$ . Si definisce *varianza* di  $\mathbf{X}$ :

$$\begin{aligned} \text{Var}[\mathbf{X}] &= \text{Cov}[\mathbf{X}, \mathbf{X}] \\ &= \mathbb{E}[(\mathbf{X} - \boldsymbol{\mu})(\mathbf{X} - \boldsymbol{\mu})^T] \\ &= \begin{bmatrix} \text{Cov}[X_1, X_1] & \text{Cov}[X_1, X_2] & \dots & \text{Cov}[X_1, X_N] \\ \text{Cov}[X_2, X_1] & \text{Cov}[X_2, X_2] & \dots & \text{Cov}[X_2, X_N] \\ \vdots & \vdots & \ddots & \vdots \\ \text{Cov}[X_N, X_1] & \dots & \dots & \text{Cov}[X_N, X_N] \end{bmatrix} \quad (\text{A.11}) \end{aligned}$$



**Figura A.1:** Due dataset bidimensionali con medie e varianze uguali (linee colorate), ma aventi covarianze diverse. Fonte [5].

### Proprietà notevoli di media e varianza

Siano  $a, b \in \mathbb{R}$  e  $\mathbf{X}, \mathbf{Y}$  due vettori aleatori. Sussistono i seguenti risultati:

$$\mathbb{E}[a\mathbf{X} + b\mathbf{Y}] = a\mathbb{E}[\mathbf{X}] + b\mathbb{E}[\mathbf{Y}] \quad \text{Linearità della media} \quad (\text{A.12})$$

$$\mathbb{E}[a\mathbf{X} + b] = a\mathbb{E}[\mathbf{X}] + b \quad (\text{A.13})$$

$$\mathbb{V}\text{ar}[a\mathbf{X} + b] = a^2\mathbb{V}\text{ar}[\mathbf{X}] \quad (\text{A.14})$$

$$\mathbb{V}\text{ar}[a\mathbf{X} + b\mathbf{Y}] = a^2\mathbb{V}\text{ar}[\mathbf{X}] + b^2\mathbb{V}\text{ar}[\mathbf{Y}] + 2ab\text{Cov}[\mathbf{X}, \mathbf{Y}] \quad (\text{A.15})$$

### A.1.3 Indipendenza statistica

**Definizione** (Indipendenza statistica). Due vettori aleatori  $\mathbf{X}, \mathbf{Y}$  sono *statisticamente indipendenti* se e solo se

$$p(\mathbf{x}, \mathbf{y}) = p(\mathbf{x})p(\mathbf{y}) \quad (\text{A.16})$$

Intuitivamente, due vettori aleatori  $\mathbf{X}$  e  $\mathbf{Y}$  sono indipendenti se la conoscenza dei valori assunti da uno dei due non dà nessuna informazione sui valori assunti dall'altro.

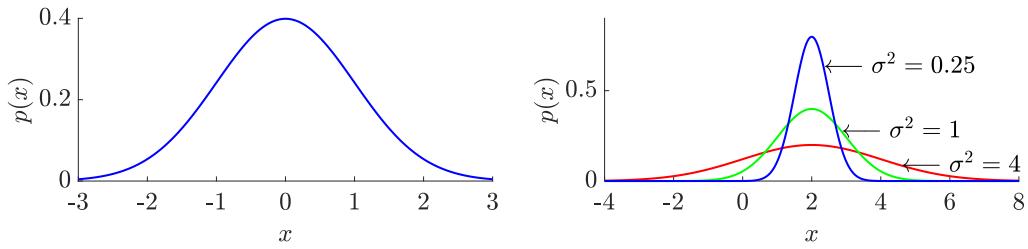
Qualora i vettori  $\mathbf{X}$  e  $\mathbf{Y}$  siano statisticamente indipendenti, risulta che:

$$p(\mathbf{y}|\mathbf{x}) = p(\mathbf{y}) \quad (\text{A.17})$$

$$p(\mathbf{x}|\mathbf{y}) = p(\mathbf{x}) \quad (\text{A.18})$$

$$\mathbb{V}\text{ar}[\mathbf{X} + \mathbf{Y}] = \mathbb{V}\text{ar}[\mathbf{X}] + \mathbb{V}\text{ar}[\mathbf{Y}] \quad (\text{A.19})$$

$$\text{Cov}[\mathbf{X}, \mathbf{Y}] = \mathbf{0} \quad (\text{A.20})$$



(a)  $pdf$  di una variabile aleatoria gaussiana standard,  $X \sim \mathcal{N}(0, 1)$ . (b)  $pdf$  di variabili gaussiane con  $\mu = 2$  e  $\sigma^2 = 4, 1, 0.25$

**Figura A.2:** Distribuzioni gaussiane unidimensionali

### A.1.4 Distribuzione gaussiana

#### Variabile aleatoria gaussiana

**Definizione.** Una variabile aleatoria continua  $X$  dicesi *gaussiana*, o *normale*, se è descritta dalla seguente  $pdf$ :

$$p(x) = \frac{1}{\sqrt{2\pi\sigma^2}} \exp\left(-\frac{(x-\mu)^2}{2\sigma^2}\right) \quad (\text{A.21})$$

Sinteticamente si scrive

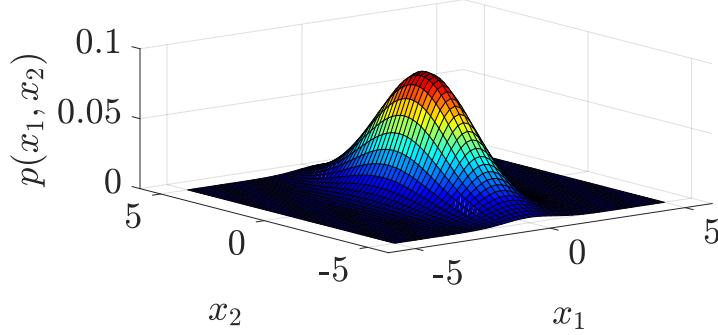
$$X \sim \mathcal{N}(\mu, \sigma^2)$$

per evidenziare come la  $pdf$  di una *v.a.* gaussiana sia completamente determinata qualora vengano assegnati i due parametri seguenti:

- $\mu = \mathbb{E}[X] \in \mathbb{R}$ , la *media statistica* della *v.a.*  $X$ .
- $\sigma^2 = \text{Var}[X]$ , la *varianza* della *v.a.*  $X$ .

Si noti (Figura A.2) la peculiare forma, tipica di una “campana”, esibita dalla  $pdf$  di una gaussiana (*bell-shaped pdf*). Dalla Figura A.2 è possibile desumere il significato dei due suddetti parametri:

- $\mu$  è un *fattore di posizione* che costituisce il centro di simmetria della densità di probabilità. Modifiche apportate a tale parametro si riflettono in traslazioni orizzontali rigide della  $pdf$ .
- $\sigma^2$  è un *fattore di scala* che regola la larghezza della campana attorno alla media  $\mu$ : all'aumentare di  $\sigma^2$  la campana si allarga e diventa più schiacciata (per preservare l'unitarietà dell'area sottesa), mentre al diminuire di  $\sigma^2$  corrisponde una campana più stretta e alta (Figura A.2).



**Figura A.3:** Distribuzione gaussiana bidimensionale

### Vettore aleatorio gaussiano

**Definizione.** Un vettore aleatorio  $\mathbf{X} = [X_1, \dots, X_N]^T \in \mathbb{R}^N$  dicesi *gaussiano*, o *normale*, qualora abbia la seguente *pdf*:

$$p(\mathbf{x}) = \frac{1}{(2\pi)^{N/2} |\Sigma|^{1/2}} \exp\left(-\frac{1}{2}(\mathbf{x} - \boldsymbol{\mu})^T \Sigma^{-1} (\mathbf{x} - \boldsymbol{\mu})\right) \quad (\text{A.22})$$

dove  $\mathbf{x} = [x_1, \dots, x_N]^T \in \mathbb{R}^N$  e  $|\Sigma| = \det(\Sigma)$  è il determinante della matrice  $\Sigma$ .

Sinteticamente si scrive

$$\mathbf{X} \sim \mathcal{N}(\boldsymbol{\mu}, \Sigma) \quad \text{o} \quad p(\mathbf{x}) = \mathcal{N}(\mathbf{x}; \boldsymbol{\mu}, \Sigma)$$

per evidenziare come la *pdf* del vettore aleatorio  $X$  sia completamente determinata qualora vengano assegnati:

- il *vettore delle medie*  $\boldsymbol{\mu} = \mathbb{E}[\mathbf{X}] = (\mathbb{E}[X_1], \mathbb{E}[X_2], \dots, \mathbb{E}[X_N])^T$ .
- la *matrice di covarianza*  $\Sigma = \mathbb{E}[(\mathbf{X} - \boldsymbol{\mu})(\mathbf{X} - \boldsymbol{\mu})^T] \in \mathbb{R}^{N \times N}$ .

*Osservazione* (Combinazione lineare di vettori gaussiani indipendenti). Siano  $\mathbf{X} \sim \mathcal{N}(\boldsymbol{\mu}_{\mathbf{X}}, \Sigma_{\mathbf{X}})$  e  $\mathbf{Y} \sim \mathcal{N}(\boldsymbol{\mu}_{\mathbf{Y}}, \Sigma_{\mathbf{Y}})$  due vettori aleatori gaussiani *indipendenti*. Siano, inoltre,  $\mathbf{A}$  e  $\mathbf{B}$  due matrici. Sussiste il seguente risultato:

$$(\mathbf{AX} + \mathbf{BY}) \sim \mathcal{N}(\mathbf{A}\boldsymbol{\mu}_{\mathbf{X}} + \mathbf{B}\boldsymbol{\mu}_{\mathbf{Y}}, \mathbf{A}\Sigma_{\mathbf{X}}\mathbf{A}^T + \mathbf{B}\Sigma_{\mathbf{Y}}\mathbf{B}^T) \quad (\text{A.23})$$

da cui si evince che la combinazione lineare di vettori aleatori gaussiani indipendenti è ancora un vettore aleatorio gaussiano. La (A.23) declinata al caso in cui le suddette matrici siano degli scalari diventa:

$$(a\mathbf{X} + b\mathbf{Y}) \sim \mathcal{N}(a\boldsymbol{\mu}_{\mathbf{X}} + b\boldsymbol{\mu}_{\mathbf{Y}}, a^2\Sigma_{\mathbf{X}} + b^2\Sigma_{\mathbf{Y}}) \quad (\text{A.24})$$

### Trasformazione affine di una v.a. gaussiana

Una variabile aleatoria gaussiana  $X \sim \mathcal{N}(\mu, \sigma^2)$  può essere sempre riguardata come una *trasformazione affine* (cascata di cambiamento di scala e traslazione) di una *gaussiana standard*  $X_0 \sim \mathcal{N}(0, 1)$

$$X = \sigma X_0 + \mu, \quad \sigma > 0, \mu \in \mathbb{R} \quad (\text{A.25})$$

### A.1.5 Catena Markoviana

Siano  $\{\mathbf{X}_i\}_{i \in \mathbb{N} \cup 0}$  una sequenza di vettori aleatori aventi tutti lo stesso alfabeto  $\mathcal{A}_{\mathbf{X}_i} = \mathcal{X}$ .

**Definizione** (Proprietà markoviana). La sequenza  $\{\mathbf{X}_i\}$  è una *catena markoviana* se sussiste la seguente proprietà:

$$\begin{aligned} p(\mathbf{x}_n | \mathbf{x}_{n-1}, \dots, \mathbf{x}_0) &= p(\mathbf{x}_n | \mathbf{x}_{n-1}) \\ \forall n \in \mathbb{N}, \forall (\mathbf{x}_0, \dots, \mathbf{x}_n) \in \mathcal{X}^n \end{aligned} \quad (\text{A.26})$$

*Osservazione* (Interpretazione). Interpretando  $n - 1$  come l'istante presente,  $n - j$  (con  $j \geq 2$ ) come il passato e  $n$  come futuro, la proprietà markoviana (A.26) asserisce che, in una catena markoviana, il futuro è indipendente dal passato dato il presente [3].

*Osservazione.* Avvalendosi della proprietà markoviana, la distribuzione congiunta di  $\mathbf{X}_0, \mathbf{X}_1, \dots, \mathbf{X}_n$  ammette la seguente riformulazione:

$$p(\mathbf{x}_0, \dots, \mathbf{x}_n) = p(\mathbf{x}_n | \mathbf{x}_{n-1}, \dots, \mathbf{x}_0) p(\mathbf{x}_{n-1} | \mathbf{x}_{n-2}, \dots, \mathbf{x}_0) \dots \quad (\text{A.27})$$

$$\dots p(\mathbf{x}_1 | \mathbf{x}_0) p(\mathbf{x}_0) \quad (\text{A.28})$$

$$= p(\mathbf{x}_n | \mathbf{x}_{n-1}) p(\mathbf{x}_{n-1} | \mathbf{x}_{n-2}) \dots p(\mathbf{x}_1 | \mathbf{x}_0) p(\mathbf{x}_0) \quad (\text{A.29})$$

$$= p(\mathbf{x}_0) \prod_{j=1}^n p(\mathbf{x}_j | \mathbf{x}_{j-1}) \quad (\text{A.30})$$

dove

- nella (A.27) si è ricorsi alla *chain rule* (A.6).
- nel passaggio (A.28) alla (A.29) si è sfruttata la proprietà markoviana (A.26).

## A.2 Problema della stima dei parametri

### A.2.1 Modellazione parametrica

In statistica, un modello parametrico è una famiglia di distribuzioni  $p_{\boldsymbol{\theta}}(\mathbf{x})$  descritta da un numero finito di *parametri*  $\boldsymbol{\theta} \in \Theta \subseteq \mathbb{R}^k$ , con  $\Theta$  detto *spazio dei parametri*.

### A.2.2 Funzione di verosimiglianza

Sia  $\mathbf{X}$  un vettore aleatorio e  $\mathbf{x} \in \mathbb{R}^N$  una sua possibile realizzazione.

**Definizione.** Si definisce funzione di verosimiglianza (*likelihood*) la funzione:

$$\boldsymbol{\theta}_i \in \Theta \longmapsto \mathcal{L}_{\mathbf{x}}(\boldsymbol{\theta}_i) = p(\mathbf{x} \mid \boldsymbol{\theta} = \boldsymbol{\theta}_i) = p_{\boldsymbol{\theta}_i}(\mathbf{x}) \quad (\text{A.31})$$

*Osservazione.* Si potrebbe essere erroneamente indotti a ritenere che  $\boldsymbol{\theta}$ , dal momento che si trova a destra del condizionamento in  $p(\mathbf{x} \mid \boldsymbol{\theta} = \boldsymbol{\theta}_i)$  (A.31), debba essere considerato come osservato e fissato: tale interpretazione è errata. Come suggerisce la notazione,  $\mathcal{L}_{\mathbf{x}}(\boldsymbol{\theta})$  è una funzione del parametro  $\boldsymbol{\theta}$ , che varia, mentre il dato  $\mathbf{x}$  è fissato. Inoltre la funzione di verosimiglianza (A.31) non è una distribuzione di probabilità in  $\boldsymbol{\theta}$ .

*Osservazione.* Allorché risulti chiaro dal contesto, si scrive semplicemente  $\mathcal{L}(\boldsymbol{\theta})$ , omettendo il pedice  $\mathbf{x}$  che si riferisce ai dati osservati.

*Osservazione (Interpretazione).* La distribuzione  $p(\mathbf{x} \mid \boldsymbol{\theta} = \boldsymbol{\theta}_i)$  modella l'incertezza dei dati  $\mathbf{x}$  per una certa istanza  $\boldsymbol{\theta}_i$  del parametro  $\boldsymbol{\theta}$ . Dunque, avvalendosi della funzione di verosimiglianza, per dei dati fissati  $\mathbf{x}$ , è possibile capire, al variare del parametro  $\boldsymbol{\theta}$ , qual'è l'istanza del parametro che ha generato i dati con maggiore probabilità. In un'ottica complementare, riguardando i dati come fissati (poiché osservati), e variando il parametro  $\boldsymbol{\theta}$ , la funzione di verosimiglianza  $\mathcal{L}(\boldsymbol{\theta})$  misura quanto è probabile una particolare scelta di  $\boldsymbol{\theta}$  per le osservazioni  $\mathbf{x}$  [5].

Se si dispone di un dataset  $\mathcal{X} = \{\mathbf{x}_1, \dots, \mathbf{x}_M\}$  di osservazioni tra loro indipendenti allora, in virtù della (A.16), la funzione di verosimiglianza è:

$$\mathcal{L}_{\mathcal{X}}(\boldsymbol{\theta}) = p(\mathcal{X} \mid \boldsymbol{\theta}) = p_{\boldsymbol{\theta}}(\mathbf{x}_1, \dots, \mathbf{x}_M) = \prod_{\mathbf{x} \in \mathcal{X}} p_{\boldsymbol{\theta}}(\mathbf{x}) \quad (\text{A.32})$$

In letteratura si considera il *logaritmo* della funzione di verosimiglianza cambiato di segno<sup>2</sup> (*negative log-likelihood*):

$$\ell_{\mathbf{x}}(\boldsymbol{\theta}) = -\log(\mathcal{L}_{\mathbf{x}}(\boldsymbol{\theta})) = -\log(p_{\boldsymbol{\theta}}(\mathbf{x})) \quad (\text{A.33})$$

Riformulando la (A.32) con il logaritmo, risulta che:

$$\ell_{\mathcal{X}}(\boldsymbol{\theta}) = -\log(\mathcal{L}_{\mathcal{X}}(\boldsymbol{\theta})) = -\log\left(\prod_{\mathbf{x} \in \mathcal{X}} p_{\boldsymbol{\theta}}(\mathbf{x})\right) = \sum_{\mathbf{x} \in \mathcal{X}} -\log p_{\boldsymbol{\theta}}(\mathbf{x}) \quad (\text{A.34})$$

### A.2.3 Metodo della massima verosimiglianza

Nel *metodo della massima verosimiglianza* (MLE) si massimizza la funzione di verosimiglianza per trovare un modello che ben si adatti ai dati di interesse:

$$\hat{\boldsymbol{\theta}} = \operatorname{argmax}_{\boldsymbol{\theta} \in \Theta} \mathcal{L}_{\mathbf{x}}(\boldsymbol{\theta}) \quad (\text{A.35})$$

Poiché le reti neurali *minimizzano* una funzione di costo, la (A.35) può essere riformulata minimizzando la *negative log-likelihood*:

$$\hat{\boldsymbol{\theta}} = \operatorname{argmin}_{\boldsymbol{\theta}} (\ell_{\mathbf{x}}(\boldsymbol{\theta})) = \operatorname{argmin}_{\boldsymbol{\theta}} (-\log p_{\boldsymbol{\theta}}(\mathbf{x})) \quad (\text{A.36})$$

Il minimizzare la *negative log-likelihood* ha un duplice beneficio [5]:

- il prodotto delle  $M$  probabilità nella (A.32) ( $M$  è la cardinalità del dataset) è soggetto a problemi di *underflow* numerico, poiché non si possono rappresentare accuratamente numeri molto piccoli come  $10^{-256}$ . Il logaritmo ovvia a tale problema trasformando i prodotti in somme.
- trasformando i prodotti in somme mediante l'uso del logaritmo, il gradiente dei termini nella (A.34) è la somma dei gradienti dei singoli addendi. In tal modo si evita di ricorrere all'applicazione reiterata della regola del prodotto per calcolare il gradiente del prodotto di  $M$  termini.

In questa trattazione si investigano i modelli generativi che si avvalgono del metodo di massima verosimiglianza, dove i parametri  $\boldsymbol{\theta}$  sono i pesi delle reti neurali del modello. L'obiettivo è quello di trovare i valori dei suddetti parametri che massimizzano la verosimiglianza (o equivalentemente, minimizzano la *negative log-likelihood*) di osservare i dati nel dataset. Tuttavia, per problemi ad alta dimensionalità dei dati coinvolti, non è possibile, generalmente, calcolare direttamente  $p_{\boldsymbol{\theta}}(\mathbf{x})$  che risulta intrattabile (si veda (3.22)).

---

<sup>2</sup>il cambiamento di segno è imputabile alla convenzione vigente nella letteratura sull'ottimizzazione numerica, in cui si predilige lo studio della minimizzazione di funzioni.

## Appendice B

# Derivazione funzione di costo DDPM

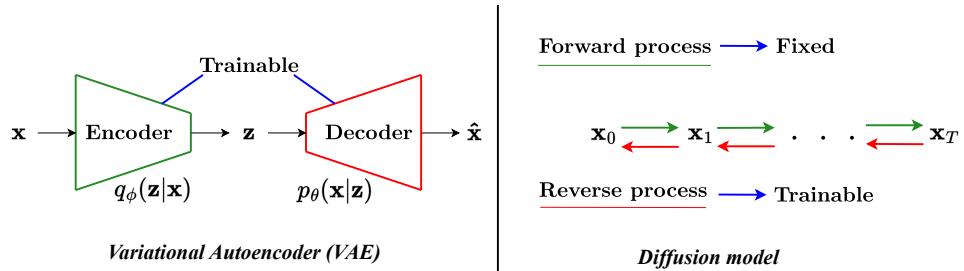
Riguardando l'immagine  $\mathbf{x}_0$  come un dato osservabile e  $\mathbf{x}_1, \dots, \mathbf{x}_T$  come variabili *latenti*, si ravvisa un'affinità con l'autoencoder variazionale (VAE) (Figura B.1).

Similitudini:

- il processo di diffusione in avanti può essere riguardato come l'equivalente dell'encoder di un VAE, che converte i dati osservabili in variabili latenti.
- il processo di diffusione inversa può essere riguardato come l'equivalente del decoder di un VAE, che, partendo dalle variabili latenti, fornisce in uscita i dati.

Differenze:

- l'encoder e il decoder di un VAE sono due reti neurali che possono essere addestrate congiuntamente, laddove il processo di diffusione in avanti in un DDPM è *fisso* (i.e non annovera parametri che possono essere



**Figura B.1:** VAE e modello di diffusione a confronto. Fonte [15].

appresi addestrando un'opportuna rete neurale). Soltanto il processo di diffusione inversa si avvale di una rete neurale.

Nel paper originale sui VAE (Kingma e Welling [13]), dette  $\mathbf{x}$  una variabile osservabile e  $\mathbf{z}$  una variabile latente, si perviene al seguente risultato:

$$\log p_{\theta}(\mathbf{x}) \geq \underbrace{\mathbb{E}_{q_{\phi}(\mathbf{z}|\mathbf{x})}[\log p_{\theta}(\mathbf{x}|\mathbf{z})] - D_{KL}(q_{\phi}(\mathbf{z}|\mathbf{x}) \| p_{\theta}(\mathbf{z}))}_{\text{Lower Bound Variazionale (ELBO)}} \quad (\text{B.1})$$

Declinando la (B.1) al contesto dei modelli di diffusione, con  $\mathbf{x}_0$  e  $\mathbf{x}_{1:T}$  che assurgono rispettivamente a dato osservabile e variabili latenti, si ha che [15]:

$$\log p_{\theta}(\mathbf{x}) \geq \mathbb{E}_{q(\mathbf{x}_{1:T}|\mathbf{x}_0)}[\log p_{\theta}(\mathbf{x}_0|\mathbf{x}_{1:T})] - D_{KL}(q(\mathbf{x}_{1:T}|\mathbf{x}_0) \| p_{\theta}(\mathbf{x}_{1:T})) \quad (\text{B.2})$$

$$= \mathbb{E}_{q(\mathbf{x}_{1:T}|\mathbf{x}_0)}[\log p_{\theta}(\mathbf{x}_0|\mathbf{x}_{1:T})] - \mathbb{E}_{q(\mathbf{x}_{1:T}|\mathbf{x}_0)}\left[\log \frac{q(\mathbf{x}_{1:T}|\mathbf{x}_0)}{p_{\theta}(\mathbf{x}_{1:T})}\right] \quad (\text{B.3})$$

$$= \mathbb{E}_q[\log p_{\theta}(\mathbf{x}_0|\mathbf{x}_{1:T})] - \mathbb{E}_q\left[\log \frac{q(\mathbf{x}_{1:T}|\mathbf{x}_0)}{p_{\theta}(\mathbf{x}_{1:T})}\right] \quad (\text{B.4})$$

$$= \mathbb{E}_q\left[\log p_{\theta}(\mathbf{x}_0|\mathbf{x}_{1:T}) - \log \frac{q(\mathbf{x}_{1:T}|\mathbf{x}_0)}{p_{\theta}(\mathbf{x}_{1:T})}\right] \quad (\text{B.5})$$

$$= \mathbb{E}_q\left[\log p_{\theta}(\mathbf{x}_0|\mathbf{x}_{1:T}) + \log \frac{p_{\theta}(\mathbf{x}_{1:T})}{q(\mathbf{x}_{1:T}|\mathbf{x}_0)}\right] \quad (\text{B.6})$$

$$= \mathbb{E}_q\left[\log \frac{p_{\theta}(\mathbf{x}_0|\mathbf{x}_{1:T}) p_{\theta}(\mathbf{x}_{1:T})}{q(\mathbf{x}_{1:T}|\mathbf{x}_0)}\right] \quad (\text{B.7})$$

$$= \mathbb{E}_q\left[\log \frac{p_{\theta}(\mathbf{x}_{0:T})}{q(\mathbf{x}_{1:T}|\mathbf{x}_0)}\right] \quad (\text{B.8})$$

dove nel passaggio dalla (B.2) alla (B.3) si è sfruttata la definizione di *divergenza di Kullback-Leibler*, dalla (B.4) alla (B.5) ci si è avvalsi della linearità dell'operatore di media statistica, e infine delle proprietà dei logaritmi.

Pertanto risulta che

$$\log p_{\theta}(\mathbf{x}) \geq \mathbb{E}_q\left[\log \frac{p_{\theta}(\mathbf{x}_{0:T})}{q(\mathbf{x}_{1:T}|\mathbf{x}_0)}\right] \quad (\text{B.9})$$

$$\implies -\log p_{\theta}(\mathbf{x}) \leq \mathbb{E}_q\left[-\log \frac{p_{\theta}(\mathbf{x}_{0:T})}{q(\mathbf{x}_{1:T}|\mathbf{x}_0)}\right] \triangleq L_{vlb} \quad (\text{B.10})$$

Sostituendo le definizioni (3.1) e (3.19) delle distribuzioni  $q(\mathbf{x}_{1:T}|\mathbf{x}_0)$  e  $p_{\theta}(\mathbf{x}_{0:T})$  e usando le proprietà dei logaritmi:

$$\begin{aligned} L_{vlb} &= \mathbb{E}_q\left[-\log p(\mathbf{x}_T) - \log \frac{\prod_{t=1}^T p_{\theta}(\mathbf{x}_{t-1}|\mathbf{x}_t)}{\prod_{t=1}^T q(\mathbf{x}_t|\mathbf{x}_{t-1})}\right] \\ &= \mathbb{E}_q\left[-\log p(\mathbf{x}_T) - \log \prod_{t=1}^T \frac{p_{\theta}(\mathbf{x}_{t-1}|\mathbf{x}_t)}{q(\mathbf{x}_t|\mathbf{x}_{t-1})}\right] \end{aligned} \quad (\text{B.11})$$

Usando la proprietà dei logaritmi per cui il logaritmo di un prodotto è la somma dei logaritmi si ha:

$$L_{vlb} = \mathbb{E}_q \left[ -\log p(\mathbf{x}_T) - \sum_{t=2}^T \log \frac{p_{\theta}(\mathbf{x}_{t-1}|\mathbf{x}_t)}{q(\mathbf{x}_t|\mathbf{x}_{t-1})} - \log \frac{p_{\theta}(\mathbf{x}_0|\mathbf{x}_1)}{q(\mathbf{x}_1|\mathbf{x}_0)} \right] \quad (\text{B.12})$$

Ricorrendo all'assunzione di Markov (A.1.5), per cui l'ulteriore condizionamento a  $\mathbf{x}_0$  è pleonastico, e alla legge di Bayes (A.7):

$$q(\mathbf{x}_t|\mathbf{x}_{t-1}) = q(\mathbf{x}_t|\mathbf{x}_{t-1}, \mathbf{x}_0) \quad (\text{B.13})$$

$$q(\mathbf{x}_t|\mathbf{x}_{t-1}, \mathbf{x}_0) = q(\mathbf{x}_{t-1}|\mathbf{x}_t, \mathbf{x}_0) \frac{q(\mathbf{x}_t|\mathbf{x}_0)}{q(\mathbf{x}_{t-1}|\mathbf{x}_0)} \quad (\text{B.14})$$

*Osservazione.* Il condizionamento rispetto a  $\mathbf{x}_0$  è essenziale:  $q(\mathbf{x}_{t-1}|\mathbf{x}_t, \mathbf{x}_0)$  ammette un'espressione in forma chiusa, laddove  $q(\mathbf{x}_{t-1}|\mathbf{x}_t)$  è intrattabile.

Sostituendo il combinato disposto della (B.13) e della (B.14) nella (B.12):

$$\mathbb{E}_q \left[ -\log p(\mathbf{x}_T) - \underbrace{\sum_{t=2}^T \log \left( \frac{p_{\theta}(\mathbf{x}_{t-1}|\mathbf{x}_t)}{q(\mathbf{x}_{t-1}|\mathbf{x}_t, \mathbf{x}_0)} \cdot \frac{q(\mathbf{x}_{t-1}|\mathbf{x}_0)}{q(\mathbf{x}_t|\mathbf{x}_0)} \right)}_{\text{B.15}} - \log \frac{p_{\theta}(\mathbf{x}_0|\mathbf{x}_1)}{q(\mathbf{x}_1|\mathbf{x}_0)} \right]$$

Focalizzandosi sulla sommatoria centrale nella (B.15) si ha che:

$$\begin{aligned} & - \sum_{t=2}^T \log \left( \frac{p_{\theta}(\mathbf{x}_{t-1}|\mathbf{x}_t)}{q(\mathbf{x}_{t-1}|\mathbf{x}_t, \mathbf{x}_0)} \cdot \frac{q(\mathbf{x}_{t-1}|\mathbf{x}_0)}{q(\mathbf{x}_t|\mathbf{x}_0)} \right) \\ &= - \sum_{t=2}^T \log \frac{p_{\theta}(\mathbf{x}_{t-1}|\mathbf{x}_t)}{q(\mathbf{x}_{t-1}|\mathbf{x}_t, \mathbf{x}_0)} - \sum_{t=2}^T \log \frac{q(\mathbf{x}_{t-1}|\mathbf{x}_0)}{q(\mathbf{x}_t|\mathbf{x}_0)} \end{aligned} \quad (\text{B.16})$$

Isolando il secondo termine di sommatoria, risulta che:

$$\begin{aligned} & - \sum_{t=2}^T \log \frac{q(\mathbf{x}_{t-1}|\mathbf{x}_0)}{q(\mathbf{x}_t|\mathbf{x}_0)} \\ &= - \sum_{t=2}^T \log q(\mathbf{x}_{t-1}|\mathbf{x}_0) + \sum_{t=2}^T \log q(\mathbf{x}_t|\mathbf{x}_0) \\ &= - \sum_{t=1}^{T-1} \log q(\mathbf{x}_t|\mathbf{x}_0) + \sum_{t=2}^T \log q(\mathbf{x}_t|\mathbf{x}_0) \\ &= - \log q(\mathbf{x}_1|\mathbf{x}_0) - \sum_{t=2}^{T-1} \log q(\mathbf{x}_t|\mathbf{x}_0) + \sum_{t=2}^{T-1} \log q(\mathbf{x}_t|\mathbf{x}_0) + \log q(\mathbf{x}_T|\mathbf{x}_0) \\ &= - \log q(\mathbf{x}_1|\mathbf{x}_0) + \log q(\mathbf{x}_T|\mathbf{x}_0) \end{aligned} \quad (\text{B.17})$$

Sostituendo il combinato disposto della (B.16) e della (B.17) nella (B.15) si ottiene che:

$$L_{vlb} = \mathbb{E}_q \left[ -\log p(\mathbf{x}_T) - \sum_{t=2}^T \log \frac{p_{\theta}(\mathbf{x}_{t-1}|\mathbf{x}_t)}{q(\mathbf{x}_{t-1}|\mathbf{x}_t, \mathbf{x}_0)} \right. \\ \left. - \log q(\mathbf{x}_1|\mathbf{x}_0) + \log q(\mathbf{x}_T|\mathbf{x}_0) - \log \frac{p_{\theta}(\mathbf{x}_0|\mathbf{x}_1)}{q(\mathbf{x}_1|\mathbf{x}_0)} \right] \quad (\text{B.18})$$

da cui, per le regole dei logaritmi:

$$L_{vlb} = \mathbb{E}_q \left[ -\log \frac{p(\mathbf{x}_T)}{q(\mathbf{x}_T|\mathbf{x}_0)} - \sum_{t=2}^T \log \frac{p_{\theta}(\mathbf{x}_{t-1}|\mathbf{x}_t)}{q(\mathbf{x}_{t-1}|\mathbf{x}_t, \mathbf{x}_0)} - \log p_{\theta}(\mathbf{x}_0|\mathbf{x}_1) \right] \quad (\text{B.19})$$

Avvalendosi della definizione di *divergenza di Kullback-Leibler*:

$$D_{KL}(p_1(x) \| p_2(x)) \triangleq \int_{-\infty}^{+\infty} p_1(x) \log \frac{p_1(x)}{p_2(x)} dx \\ = \mathbb{E}_{x \sim p_1(x)} \left[ \log \frac{p_1(x)}{p_2(x)} \right] \\ = \mathbb{E}_{p_1} \left[ -\log \frac{p_2(x)}{p_1(x)} \right] \quad (\text{B.20})$$

si perviene, infine, al seguente risultato:

$$L_{vlb} = \underbrace{-\log p_{\theta}(\mathbf{x}_0|\mathbf{x}_1)}_{L_0} \\ + \sum_{t=2}^T \underbrace{D_{KL}(q(\mathbf{x}_{t-1}|\mathbf{x}_t, \mathbf{x}_0) \| p_{\theta}(\mathbf{x}_{t-1}|\mathbf{x}_t))}_{L_{t-1}} \\ + \underbrace{D_{KL}(q(\mathbf{x}_T|\mathbf{x}_0) \| p(\mathbf{x}_T))}_{L_T} \quad (\text{B.21})$$

## Espressione analitica di $L_{t-1}$

Si dimostra che (Sohl-Dickstein et al. [19]), sebbene  $q(\mathbf{x}_{t-1}|\mathbf{x}_t)$  sia intrattabile, condizionando ulteriormente rispetto a  $\mathbf{x}_0$  si perviene, (si veda [15]), ad una distribuzione gaussiana, che, in quanto tale, è trattabile:

$$q(\mathbf{x}_{t-1}|\mathbf{x}_t, \mathbf{x}_0) = \mathcal{N}(\mathbf{x}_{t-1}; \tilde{\boldsymbol{\mu}}_t(\mathbf{x}_t, \mathbf{x}_0), \tilde{\beta}_t \mathbf{I}) \quad (\text{B.22})$$

dove

$$\tilde{\boldsymbol{\mu}}_t(\mathbf{x}_t, \mathbf{x}_0) = \frac{\sqrt{\alpha_t}(1 - \bar{\alpha}_{t-1})}{1 - \bar{\alpha}_t} \mathbf{x}_t + \frac{\sqrt{\bar{\alpha}_{t-1}}\beta_t}{1 - \bar{\alpha}_t} \mathbf{x}_0 \quad (\text{B.23})$$

$$\tilde{\beta}_t = \frac{1 - \bar{\alpha}_{t-1}}{1 - \bar{\alpha}_t} \beta_t \quad (\text{B.24})$$

in cui  $\alpha_t$  e  $\bar{\alpha}_t$  sono definiti nella (3.4) e, dipendendo solo dagli iperparametri  $\beta_t$ , possono essere pre-calcolati.

Quindi  $\tilde{\beta}_t$  e  $\tilde{\boldsymbol{\mu}}_t(\mathbf{x}_t, \mathbf{x}_0)$  della distribuzione (B.25) sono i parametri che possono essere approssimati dalla media e varianza della distribuzione (B.26):

$$q(\mathbf{x}_{t-1} | \mathbf{x}_t, \mathbf{x}_0) = \mathcal{N}(\mathbf{x}_{t-1}; \tilde{\boldsymbol{\mu}}_t(\mathbf{x}_t, \mathbf{x}_0), \tilde{\beta}_t \mathbf{I}) \quad (\text{B.25})$$

$$p_{\boldsymbol{\theta}}(\mathbf{x}_{t-1} | \mathbf{x}_t) = \mathcal{N}(\mathbf{x}_{t-1}; \boldsymbol{\mu}_{\boldsymbol{\theta}}(\mathbf{x}_t, t), \boldsymbol{\Sigma}_{\boldsymbol{\theta}}(\mathbf{x}_t, t)) \quad (\text{B.26})$$

addestrando un'opportuna rete neurale.

Pertanto,  $L_{t-1} = D_{KL}(q(\mathbf{x}_{t-1} | \mathbf{x}_t, \mathbf{x}_0) \| p_{\boldsymbol{\theta}}(\mathbf{x}_{t-1} | \mathbf{x}_t))$  risulta essere la divergenza di Kullback-Leibler tra due distribuzioni gaussiane e, in quanto tale, ammette un'espressione in forma chiusa.

Sebbene nella (B.26) la rete neurale possa apprendere sia la media che la varianza, Ho et al. [12] hanno rilevato sperimentalmente che l'apprendimento di quest'ultima degrada la qualità dei campioni prodotti dalla rete neurale a valle dell'addestramento. Hanno deciso, quindi, di addestrare la rete neurale ad apprendere la sola media  $\boldsymbol{\mu}_{\boldsymbol{\theta}}(\mathbf{x}_t, t)$ , fissando la varianza  $\boldsymbol{\Sigma}_{\boldsymbol{\theta}}(\mathbf{x}_t, t) = \beta_t \mathbf{I}$ .

Dunque, nella fase di addestramento la rete neurale deve apprendere la media  $\boldsymbol{\mu}_{\boldsymbol{\theta}}(\mathbf{x}_t, t)$ , cosicché la distribuzione (B.28) approssimi quanto più fedelmente possibile la (B.27):

$$q(\mathbf{x}_{t-1} | \mathbf{x}_t, \mathbf{x}_0) = \mathcal{N}(\mathbf{x}_{t-1}; \tilde{\boldsymbol{\mu}}_t(\mathbf{x}_t, \mathbf{x}_0), \tilde{\beta}_t \mathbf{I}) \quad (\text{B.27})$$

$$p_{\boldsymbol{\theta}}(\mathbf{x}_{t-1} | \mathbf{x}_t) = \mathcal{N}(\mathbf{x}_{t-1}; \boldsymbol{\mu}_{\boldsymbol{\theta}}(\mathbf{x}_t, t), \beta_t \mathbf{I}) \quad (\text{B.28})$$

Inoltre, Ho et al. [12] hanno osservato che, avvalendosi dell'artificio di riparametrizzazione (A.25)  $\mathbf{x}_0$  può essere così riguardato:

$$\mathbf{x}_0 = \frac{1}{\sqrt{\bar{\alpha}_t}} (\mathbf{x}_t - \sqrt{1 - \bar{\alpha}_t} \boldsymbol{\epsilon}) \quad (\text{B.29})$$

dove  $\boldsymbol{\epsilon} \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$ . Dal combinato disposto della (B.23) e della (B.29) risulta che  $\tilde{\boldsymbol{\mu}}_t$  dipende solo da  $\mathbf{x}_t$ :

$$\tilde{\boldsymbol{\mu}}_t(\mathbf{x}_t) = \frac{1}{\sqrt{\alpha_t}} \left( \mathbf{x}_t - \frac{1 - \alpha_t}{\sqrt{1 - \bar{\alpha}_t}} \boldsymbol{\epsilon} \right) \quad (\text{B.30})$$

Quindi in ogni passo della diffusione inversa la rete neurale deve approssimare la media nella (B.30). Tuttavia, poiché  $\mathbf{x}_t$  è disponibile come input del generico passo di diffusione inversa e gli  $\alpha_t$  sono noti a monte della fase di training, allora apprendere la suddetta media si traduce nell'apprendere il rumore  $\epsilon_t$  additivo che è stato aggiunto nel corrispondente passo di diffusione in avanti:

$$\boldsymbol{\mu}_{\theta}(\mathbf{x}_t, t) = \frac{1}{\sqrt{\alpha_t}} \left( \mathbf{x}_t - \frac{1 - \alpha_t}{\sqrt{1 - \bar{\alpha}_t}} \boldsymbol{\epsilon}_{\theta}(\mathbf{x}_t, t) \right) \quad (\text{B.31})$$

Quindi il termine  $L_{t-1}$  risulta essere:

$$L_{t-1} = D_{KL} \left( \mathcal{N} \left( \mathbf{x}_{t-1}; \tilde{\boldsymbol{\mu}}_t(\mathbf{x}_t, \mathbf{x}_0), \tilde{\beta}_t \mathbf{I} \right) \parallel \mathcal{N} \left( \mathbf{x}_{t-1}; \boldsymbol{\mu}_{\theta}(\mathbf{x}_t, t), \beta_t \mathbf{I} \right) \right) \quad (\text{B.32})$$

dove  $\boldsymbol{\mu}_{\theta}(\mathbf{x}_t, t)$  è quella della (B.31).

Ma la (B.32) è la divergenza di Kullback-Leibler tra due distribuzioni gaussiane e, a valle di alcuni calcoli (si veda [15]), si perviene alla seguente riformulazione per  $L_{t-1}$ :

$$L_{t-1} = \mathbb{E}_{\mathbf{x}_0, \epsilon} \left[ \underbrace{\frac{(1 - \alpha_t)^2}{2\alpha_t \beta_t (1 - \bar{\alpha}_t)}}_{\text{termine di scalatura}} \left\| \boldsymbol{\epsilon} - \boldsymbol{\epsilon}_{\theta} \left( \underbrace{\sqrt{\bar{\alpha}_t} \mathbf{x}_0 + \sqrt{1 - \bar{\alpha}_t} \boldsymbol{\epsilon}}_{\mathbf{x}_t}, t \right) \right\|^2 \right] \quad (\text{B.33})$$

Infine Ho et al. [12] hanno osservato sperimentalmente che, ignorando il termine di scalatura nella (B.33), si ottengono prestazioni migliori a valle dell'addestramento del modello.

Pertanto, ognuno degli  $L_{t-1}$  nella (B.21) esibisce la seguente espressione semplificata:

$$L_{t-1}^{\text{simple}} = \left\| \boldsymbol{\epsilon} - \boldsymbol{\epsilon}_{\theta} \left( \underbrace{\sqrt{\bar{\alpha}_t} \mathbf{x}_0 + \sqrt{1 - \bar{\alpha}_t} \boldsymbol{\epsilon}}_{\mathbf{x}_t}, t \right) \right\|^2 + C \quad (\text{B.34})$$

dove  $C$  è una costante che non dipende da  $\theta$  e, quindi, può essere ignorata durante l'addestramento.

# Appendice C

## U-Net

La U-Net è stata introdotta nel 2015 (Ronneberger et al., [17]), e costituiva, allora, lo stato dell’arte nell’ambito della segmentazione delle immagini mediche.

Si ribadisce che la scelta di una U-Net, nel paper di Ho et al. [12], coniuga l’esigenza che le immagini a monte e a valle della rete neurale abbiano le stesse dimensioni, con la peculiarità della U-Net di produrre un’immagine con le medesime dimensioni dell’immagine in ingresso.

In Figura C.1 è riportato il diagramma della U-Net in uno step del processo di diffusione inversa. In particolare sono mostrate esplicitamente le dimensioni dell’immagine rumorosa  $\mathbf{x}_t$  nel suo percorso attraverso la rete. Essendo un’immagine a colori, matematicamente  $\mathbf{x}_t$  è un *tensore*.

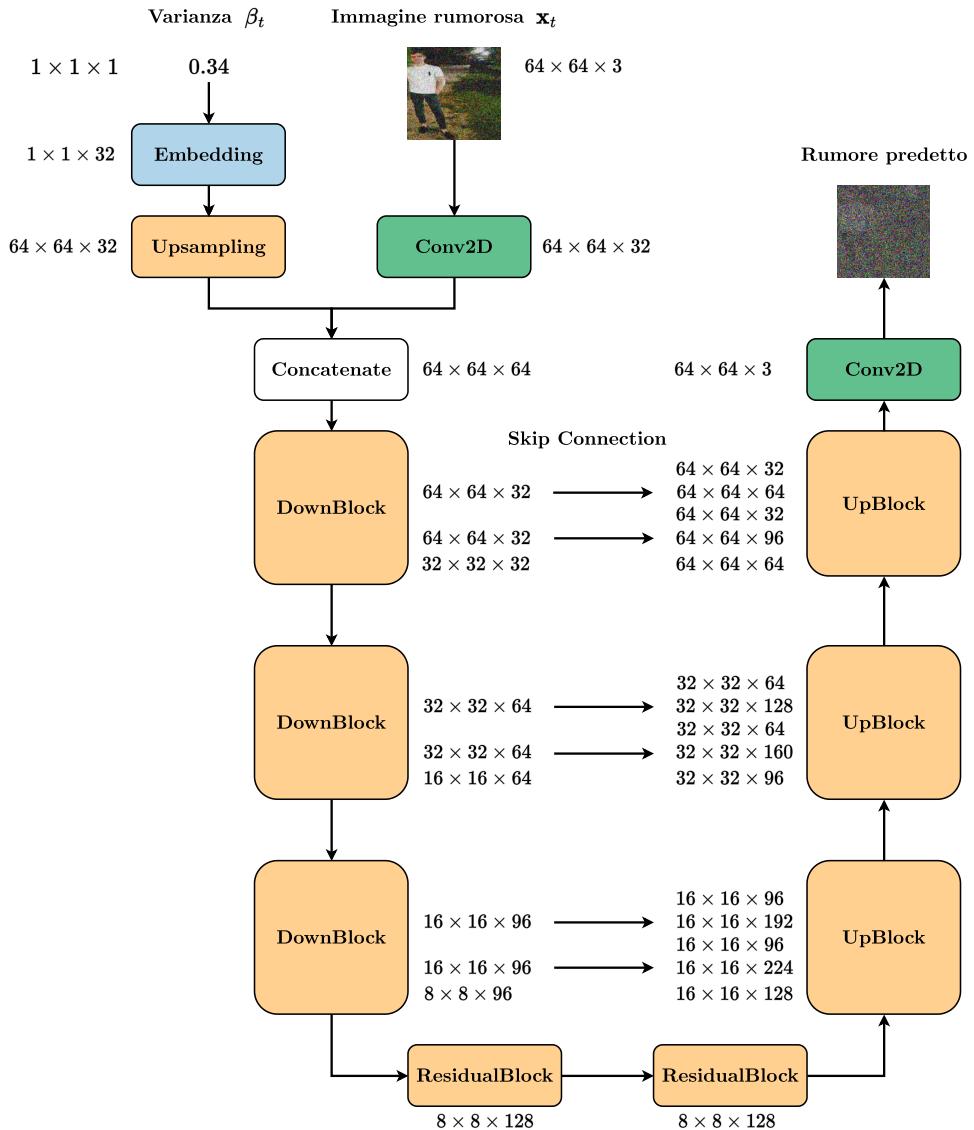
Come si evince dalla Figura C.1, la U-Net è composta da due metà:

- la parte di *downsampling* (a sinistra in Figura C.1), dove l’immagine viene *compressa spazialmente* ma *espansa a livello di canale* (o profondità).
- la parte di *upsampling* (a destra in Figura C.1), dove l’immagine viene *espansa spazialmente* ma *compressa a livello di canale*.

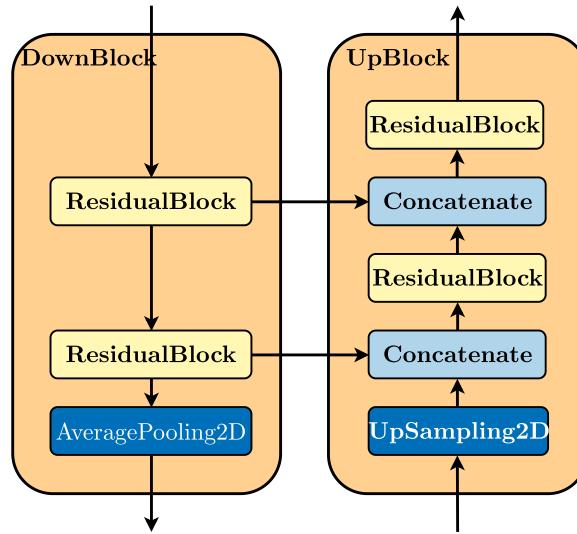
In Figura C.2 sono riportati i dettagli dei blocchi “*DownBlock*” e “*UpBlock*” della Figura C.1.

In particolare, relativamente all’esempio in Figura C.1, ognuno dei blocchi “*DownBlock*” nella parte di *downsampling* incrementa il numero di canali del tensore rappresentante l’immagine mediante l’impiego di due “blocchi residuali” (*ResidualBlock*), applicando anche un layer finale “*AveragePooling2D*” per dimezzare la dimensione dell’immagine (si veda Figura C.2).

Nella porzione di *upsampling* della U-Net, ognuno dei blocchi “*UpBlock*” applica preliminarmente un layer “*UpSampling2D*” atto a raddoppiare le



**Figura C.1:** Diagramma della U-Net in uno step della diffusione inversa.  
 Fonte [8].



**Figura C.2:** Dettagli dei blocchi “*DownBlock*” e “*UpBlock*” della U-Net di Figura C.1. Fonte [8].

dimensioni dell’immagine, ricorrendo all’interpolazione bilineare [8]. Inoltre ognuno degli “*UpBlock*” diminuisce il numero di canali dell’immagine mediante l’impiego di due “*ResidualBlock*”, concatenando, al contempo, gli output dei blocchi “*DownBlock*” per il tramite delle cosiddette “connesioni scorciatoia” (*skip connection*), peculiari della U-Net [8].

# Bibliografia

- [1] Ziyi Chang, George Alex Koulieris e Hubert P. H. Shum. *On the Design Fundamentals of Diffusion Models: A Survey*. 19 Ott. 2023. arXiv: 2306.04542 [cs]. URL: <http://arxiv.org/abs/2306.04542>. preprint.
- [2] Ting Chen. *On the Importance of Noise Scheduling for Diffusion Models*. 21 Mag. 2023. arXiv: 2301.10972 [cs]. URL: <http://arxiv.org/abs/2301.10972>. preprint.
- [3] Ernesto Conte e Carmela Galdi. *Fenomeni aleatori*. Aracne, 1 dic. 2006. 316 pp. ISBN: 978-88-548-0813-3.
- [4] *DALL-E*. In: *Wikipedia*. 2 Dic. 2023. URL: <https://en.wikipedia.org/w/index.php?title=DALL-E&oldid=1187974775>.
- [5] Marc Peter Deisenroth, A. Aldo Faisal e Cheng Soon Ong. *Mathematics for Machine Learning*. Studies in Natural Language Processing. Cambridge ; New York, NY: Cambridge University Press, 23 apr. 2020. 390 pp. ISBN: 978-1-108-67993-0. DOI: 10.1017/9781108679930. URL: <https://www.cambridge.org/highereducation/books/mathematics-for-machine-learning/5EE57FD1CFB23E6EB11E130309C7EF98>.
- [6] Prafulla Dhariwal e Alex Nichol. *Diffusion Models Beat GANs on Image Synthesis*. 1 Giu. 2021. arXiv: 2105.05233 [cs, stat]. URL: <http://arxiv.org/abs/2105.05233>. preprint.
- [7] W. Feller. «On the Theory of Stochastic Processes, with Particular Reference to Applications». In: *Proceedings of the [First] Berkeley Symposium on Mathematical Statistics and Probability*. Vol. 1. University of California Press, 1 gen. 1949, pp. 403–433. URL: <https://projecteuclid.org/ebooks/berkeley-symposium-on-mathematical-statistics-and-probability/Proceedings-of-the-First-Berkeley-Symposium-on-Mathematical-Statistics-and/chapter/On-the-Theory-of-Stochastic-Processes-with-Particular-Reference-to-bsmsp/1166219215>.

- [8] David Foster e Karl Friston. *Generative Deep Learning: Teaching Machines to Paint, Write, Compose, and Play*. 2° edizione. Beijing ; Boston: O'Reilly Media, Inc., 31 mag. 2023. 426 pp. ISBN: 978-1-09-813418-1.
- [9] Ian Goodfellow. *NIPS 2016 Tutorial: Generative Adversarial Networks*. 3 Apr. 2017. arXiv: 1701.00160 [cs]. URL: <http://arxiv.org/abs/1701.00160>. preprint.
- [10] *Google Colaboratory*. URL: <https://colab.research.google.com/>.
- [11] Martin Heusel et al. «GANs Trained by a Two Time-Scale Update Rule Converge to a Local Nash Equilibrium». In: *Advances in Neural Information Processing Systems*. Vol. 30. Curran Associates, Inc., 2017. URL: <https://papers.nips.cc/paper/2017/hash/8a1d694707eb0fe6e65871369074926d-Abstract.html>.
- [12] Jonathan Ho, Ajay Jain e Pieter Abbeel. «Denoising Diffusion Probabilistic Models». In: *Advances in Neural Information Processing Systems*. Vol. 33. Curran Associates, Inc., 2020, pp. 6840–6851. URL: [https://proceedings.neurips.cc/paper\\_files/paper/2020/hash/4c5bcfec8584af0d967f1ab10179ca4b-Abstract.html](https://proceedings.neurips.cc/paper_files/paper/2020/hash/4c5bcfec8584af0d967f1ab10179ca4b-Abstract.html).
- [13] Diederik P. Kingma e Max Welling. *Auto-Encoding Variational Bayes*. 10 Dic. 2022. arXiv: 1312.6114 [cs, stat]. URL: <http://arxiv.org/abs/1312.6114>. preprint.
- [14] *Midjourney*. Midjourney. URL: <https://www.midjourney.com/home/?callbackUrl=%2Fapp%2F>.
- [15] Aakash Kumar Nain. *The Latent: Code the Maths - A Deep Dive into DDPMs*. 2 Set. 2022. URL: <https://magic-with-latents.github.io/latent/posts/ddpms/part3/>.
- [16] Alexander Quinn Nichol e Prafulla Dhariwal. «Improved Denoising Diffusion Probabilistic Models». In: *Proceedings of the 38th International Conference on Machine Learning*. International Conference on Machine Learning. PMLR, 1 lug. 2021, pp. 8162–8171. URL: <https://proceedings.mlr.press/v139/nichol21a.html>.
- [17] Olaf Ronneberger, Philipp Fischer e Thomas Brox. *U-Net: Convolutional Networks for Biomedical Image Segmentation*. 18 Mag. 2015. arXiv: 1505.04597 [cs]. URL: <http://arxiv.org/abs/1505.04597>. preprint.
- [18] Subhradip Roy. *A Beginner's Guide to Diffusion Models: Understanding the Basics and Beyond*. Subhradip Roy's Blog. URL: <https://roysubhradip.hashnode.dev/a-beginners-guide-to-diffusion-models-understanding-the-basics-and-beyond>.

- [19] Jascha Sohl-Dickstein et al. «Deep Unsupervised Learning Using Nonequilibrium Thermodynamics». In: *Proceedings of the 32nd International Conference on Machine Learning*. International Conference on Machine Learning. PMLR, 1 giu. 2015, pp. 2256–2265. URL: <https://proceedings.mlr.press/v37/sohl-dickstein15.html>.
- [20] Yang Song e Stefano Ermon. «Improved Techniques for Training Score-Based Generative Models». In: *Advances in Neural Information Processing Systems*. Vol. 33. Curran Associates, Inc., 2020, pp. 12438–12448. URL: <https://proceedings.neurips.cc/paper/2020/hash/92c3b916311a5517d9290576e3ea37ad-Abstract.html>.
- [21] Christian Szegedy et al. *Rethinking the Inception Architecture for Computer Vision*. 11 Dic. 2015. arXiv: 1512.00567 [cs]. URL: <http://arxiv.org/abs/1512.00567>. preprint.
- [22] Vaibhav Singh. *An In-Depth Guide to Denoising Diffusion Probabilistic Models – From Theory to Implementation*. 6 Mar. 2023. URL: <https://learnopencv.com/denoising-diffusion-probabilistic-models/>.
- [23] Cam Wilson. *Adobe Is Selling Fake AI Images of the War in Israel-Gaza*. Crikey. 1 Nov. 2023. URL: <https://www.crikey.com.au/2023/11/01/israel-gaza-adobe-artificial-intelligence-images-fake-news/>.

# Ringraziamenti

*Il primo pensiero è per mio fratello Lorenzo, al quale esprimo gratitudine e al contempo ammirazione per l'eccezionale forza di spirito mostrata dinanzi alle avversità della vita. Sbarcandosi, come un cireneo, di responsabilità gravose, mi ha consentito di raggiungere la fine di questo percorso. Ringrazio mia madre Rita che ha trovato dentro di sè la forza di andare avanti, nonostante tutta l'oscurità. Ringrazio, inoltre, i Professori Alessandro Bria e Daniele Pinchera per le parole di stima proferte nei miei confronti a valle dei rispettivi esami che, riecheggiando nella mia mente, mi hanno permesso di andare avanti nei momenti più bui. Ringrazio mio zio Giuseppe, per credere sempre in me e nelle mie potenzialità.*

*Ringrazio, infine, tutti i miei amici.*