UTEC
UNIVERSIDAD DE INGENIERÍA Y TECNOLOGÍA

> Reinventa el mundo <

Sesión 5.1

Diffusion models

DDPM, LDM, LCM, DiT

TRANSFORMATEC

UTEC
UNIVERSIDAD DE INGENIERÍA Y TECNOLOGÍA

1.

Diffusion *Models*

TRANSFORMATEC

Diffusion *process*

Es un proceso estocástico en el que una variable aleatoria evoluciona de manera continua en el tiempo y su cambio está gobernado por ecuaciones diferenciales estocásticas. Se utilizan para modelar sistemas dinámicos en los que hay incertidumbre o fluctuaciones aleatorias.

$$dX_t = \mu(X_t, t)dt + \sigma(X_t, t)dW_t$$

Ecuación de Itô

donde:

- X_t es el valor del proceso en el tiempo t.
- $\mu(X_t, t)$ es la **función de deriva** (controla la tendencia del proceso).
- $\sigma(X_t, t)$ es la **difusión** (controla la variabilidad).
- W_t es un **proceso de Wiener**.

Kiyosi Itô

Diffusion *process*

Es un proceso estocástico en el que una variable aleatoria evoluciona de manera continua en el tiempo y su cambio está gobernado por ecuaciones diferenciales estocásticas. Se utilizan para modelar sistemas dinámicos en los que hay incertidumbre o fluctuaciones aleatorias.

$$dX_t = \mu(X_t, t)dt + \sigma(X_t, t)dW_t$$

Ecuación de Itô

Un proceso estocástico W_t es un proceso de Wiener si cumple las siguientes condiciones:

1. $W_0 = 0$ casi seguramente ($P = 1$).
2. Incrementos independientes: Para cualquier conjunto de tiempos $t_1 < t_2 < t_3 < \cdots < t_n$, los incrementos $W_{t_2} - W_{t_1}, W_{t_3} - W_{t_2}$, son independientes entre sí.
3. Incrementos gaussianos: Para cualquier $s < t$, el incremento $W_t - W_s$ sigue una distribución normal con media cero y varianza $t - s$:

$$W_t - W_s \sim \mathcal{N}(0, t - s)$$

4. Trayectorias continuas: W_t es una función continua en t con probabilidad 1, aunque sus trayectorias son irregulares (no diferenciables en casi ningún punto).

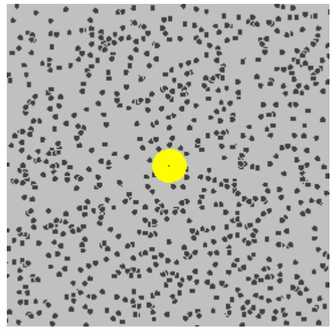
Kiyosi Itô

Diffusion *process*

Es un proceso estocástico en el que una variable aleatoria evoluciona de manera continua en el tiempo y su cambio está gobernado por ecuaciones diferenciales estocásticas. Se utilizan para modelar sistemas dinámicos en los que hay incertidumbre o fluctuaciones aleatorias.

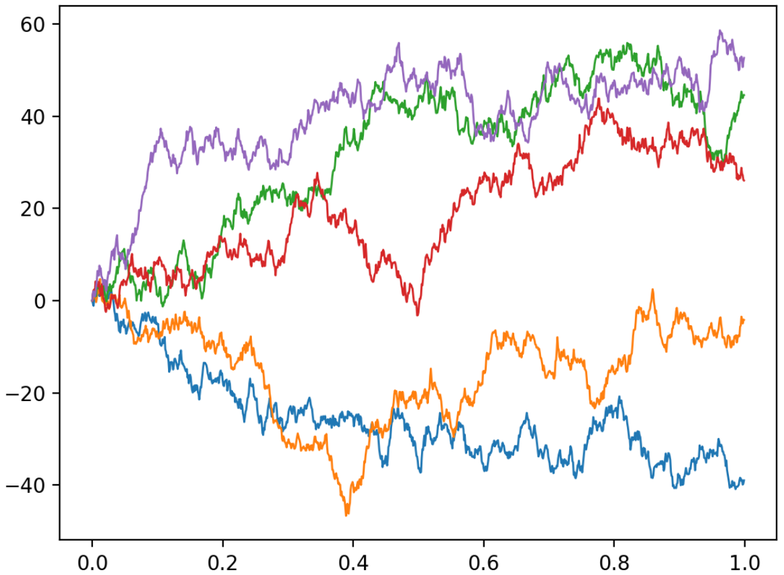
$$dX_t = \mu(X_t, t)dt + \sigma(X_t, t)dW_t$$

Ecuación de Itô



Brownian motion

Sampled paths of brownian motion



Diffusion *process*

Se considera el stochastic differential equation (SDE) en tiempo continuo:

$$dx = -\frac{1}{2}\beta(t)xdt + \sqrt{\beta(t)}dW_t$$

donde:

- $\beta(t)$ es una función o schedule que, en la práctica, se evalúa en pasos discretos: $\beta_t = \beta(t)$ (se supone pasos muy pequeños).
- dW_t es el incremento de un proceso de Wiener (con $dW_t \sim \mathcal{N}(0, dt)$).

Diffusion *process*

Se considera el stochastic differential equation (SDE) en tiempo continuo:

$$dx = -\frac{1}{2}\beta(t)xdt + \sqrt{\beta(t)}dW_t$$

donde:

- $\beta(t)$ es una función o schedule que, en la práctica, se evalúa en pasos discretos: $\beta_t = \beta(t)$ (se supone pasos muy pequeños).
- dW_t es el incremento de un proceso de Wiener (con $dW_t \sim \mathcal{N}(0, dt)$).

Esta SDE tiene dos propiedades deseadas:

1. **Disminución de la magnitud de la entrada:** La parte determinista $-\frac{1}{2}\beta(t)xdt$ reduce la magnitud de x.
2. **Adición controlada de ruido:** La parte estocástica $\sqrt{\beta(t)}dW_t$ añade ruido gaussiano con varianza proporcional a $\beta(t)$.

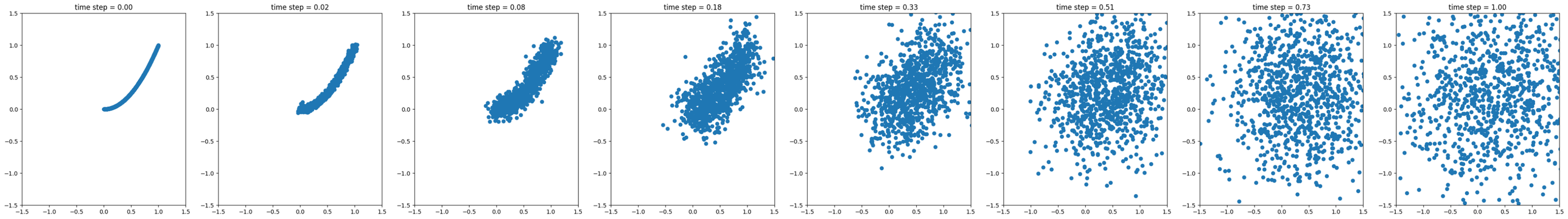
Diffusion *process*

Se considera el stochastic differential equation (SDE) en tiempo continuo:

$$dx = -\frac{1}{2}\beta(t)xdt + \sqrt{\beta(t)}dW_t$$

donde:

- $\beta(t)$ es una función o schedule que, en la práctica, se evalúa en pasos discretos: $\beta_t = \beta(t)$ (se supone pasos muy pequeños).
- dW_t es el incremento de un proceso de Wiener (con $dW_t \sim \mathcal{N}(0, dt)$).



Diffusion *process*

Se considera el stochastic differential equation (SDE) en tiempo continuo:

$$dx = -\frac{1}{2}\beta(t)xdt + \sqrt{\beta(t)}dW_t$$

Diffusion *process*

Se considera el stochastic differential equation (SDE) en tiempo continuo:

$$dx = -\frac{1}{2}\beta(t)xdt + \sqrt{\beta(t)}dW_t$$

La solución en el intervalo $[t, t+1]$ es:

$$x_{t+1} = x_t \exp\left(-\frac{1}{2}\beta_t\right) + \sqrt{1-\exp(-\beta_t)}\, z_t, \quad \text{donde } z_t \sim \mathcal{N}(0, I).$$

Ya que β_t es pequeño, podemos aproximar: $\exp\left(-\frac{1}{2}\beta_t\right) \approx \sqrt{1-\beta_t}$ $\sqrt{1-\exp(-\beta_t)} \approx \sqrt{\beta_t}$

Entonces obtenemos:

$$x_{t+1} = \sqrt{1-\beta_t}x_t + \sqrt{\beta_t}z_t, \quad z_t \sim \mathcal{N}(0, I)$$

Diffusion probabilistic models

Forward Process: Se define como un proceso de ruido aditivo gaussiano con reducción progresiva de señal.

$$x_t = \sqrt{1-\beta_t}x_{t-1} + \sqrt{\beta_t}z_t, \quad z_t \sim \mathcal{N}(0, I)$$

Jascha Sohl-Dickstein et al. (2015) "Deep Unsupervised Learning using Nonequilibrium Thermodynamics". International conference on machine learning (pp. 2256-2265). PMLR.

Diffusion probabilistic models

Forward Process: Se define como un proceso de ruido aditivo gaussiano con reducción progresiva de señal.

$$x_t = \sqrt{1-\beta_t}x_{t-1} + \sqrt{\beta_t}z_t, \quad z_t \sim \mathcal{N}(0, I)$$

Reverse Process: Se aprende una distribución gaussiana para reconstruir los datos originales.

$$x_{t-1} = \frac{1}{\sqrt{1-\beta_t}}(x_t - \sigma_t z_t)$$

Jascha Sohl-Dickstein et al. (2015) "Deep Unsupervised Learning using Nonequilibrium Thermodynamics". International conference on machine learning (pp. 2256-2265). PMLR.

Diffusion probabilistic models

Forward Process: Se define como un proceso de ruido aditivo gaussiano con reducción progresiva de señal.

$$x_t = \sqrt{1-\beta_t}x_{t-1} + \sqrt{\beta_t}z_t, \quad z_t \sim \mathcal{N}(0, I)$$

Se define como una cadena de Markov en la que los datos se corrompen progresivamente:

$$q(x_t \mid x_{t-1}) = \mathcal{N}\left(x_t; \sqrt{1-\beta_t}x_{t-1}, \beta_t I\right)$$

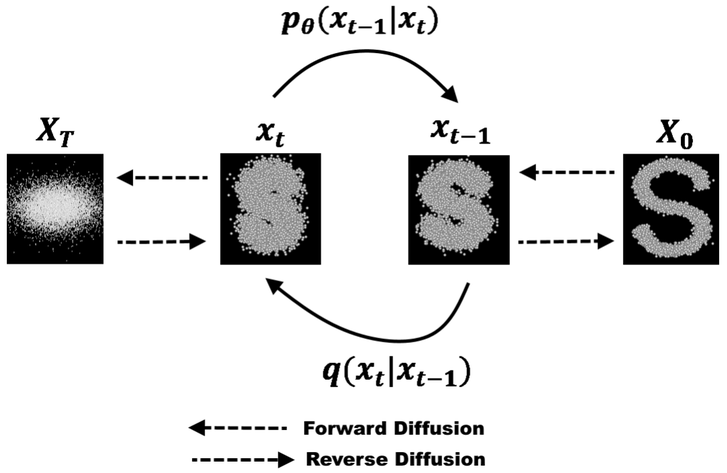
Reverse Process: Se aprende una distribución gaussiana para reconstruir los datos originales.

$$x_{t-1} = \frac{1}{\sqrt{1-\beta_t}}(x_t - \sigma_t z_t)$$

También se formula como una cadena de Markov pero en dirección contraria:

$$p_\theta(x_{t-1} \mid x_t) = \mathcal{N}(x_{t-1}; \boldsymbol{\mu_\theta(x_t, t)}, \Sigma)$$

Red Neuronal

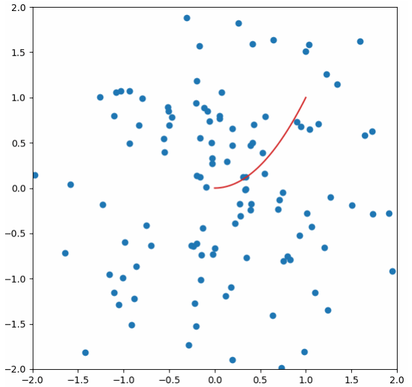


Jascha Sohl-Dickstein et al. (2015) "Deep Unsupervised Learning using Nonequilibrium Thermodynamics". International conference on machine learning (pp. 2256-2265). PMLR.

Diffusion probabilistic models

Entrenamiento Se basa en minimizar la divergencia KL entre:

- La verdadera distribución inversa $q(\, x_{t-1} \mid x_t, x_0 \,)$ (obtenida por el forward process).
- La distribución modelada por la red $p_\theta(\, x_{t-1} \mid x_t \,)$.



El objetivo final es maximizar la log-verosimilitud de los datos x_0, lo que equivale a minimizar el ELBO:

$$\log p\,(x_0) \geq E_q\left[-\sum_{t=1}^{T} D_{\mathrm{KL}}\big(q(x_{t-1}|x_t,x_0)||p_\theta(x_{t-1}|x_t)\big)\right]$$

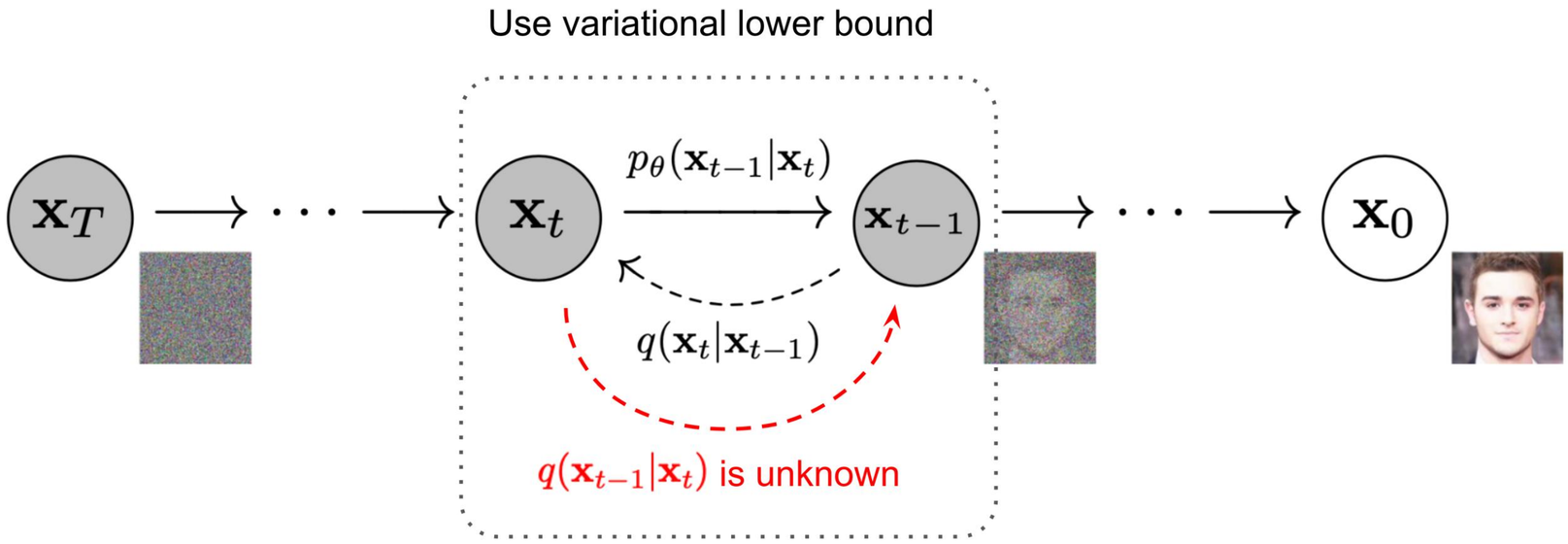
Es decir, se entrena la red neuronal $\mu_\theta(x_t,t)$ para que la distribución inversa estimada $p_\theta(x_{t-1}|x_t)$ se acerque lo más posible a la distribución teórica $q(x_{t-1}|x_t,x_0)$.

$$\mathcal{L}_{ELBO} = \sum_{t=1}^{T} D_{\mathrm{KL}}\big(q(x_{t-1}|x_t,x_0)||p_\theta(x_{t-1}|x_t)\big)$$

Este criterio obliga a la red neuronal a aprender la media correcta $\mu_\theta(x_t,t)$ para revertir el proceso de difusión.

Jascha Sohl-Dickstein et al. (2015) "Deep Unsupervised Learning using Nonequilibrium Thermodynamics". International conference on machine learning (pp. 2256-2265). PMLR.

Diffusion proba*bilistic models*



Jascha Sohl-Dickstein et al. (2015) "Deep Unsupervised Learning using Nonequilibrium Thermodynamics". International conference on machine learning (pp. 2256-2265). PMLR.

DDPM

(Denoising Diffusion Probabilistic Models)

Forward Process: Se define: $q(x_t \mid x_{t-1}) = \mathcal{N}\left(x_t; \sqrt{1-\beta_t}x_{t-1}, \beta_t I\right)$ donde $x_t = \sqrt{1-\beta_t}x_{t-1} + \sqrt{\beta_t}z_t, \quad z_t \sim \mathcal{N}(0, I)$

Aquí β_t controla la cantidad de ruido que se añade en cada paso.

Jonathan Ho et al. propusieron reformular la ecuación:

$$x_t = \sqrt{\bar{\alpha}_t}\, x_0 + \sqrt{1-\bar{\alpha}_t}\, \epsilon, \quad \epsilon \sim \mathcal{N}(0, I) \qquad \text{donde:} \quad \bar{\alpha}_t = \prod_{s=1}^{t} \beta_s$$

Jonathan Ho et al. (2020) "Denoising Diffusion Probabilistic Models".
Advances in neural information processing systems, 33, 6840-6851.

DDPM

(Denoising Diffusion Probabilistic Models)

Forward Process: Se define: $q(x_t \mid x_{t-1}) = \mathcal{N}\left(x_t; \sqrt{1-\beta_t}x_{t-1}, \beta_t I\right)$ donde $x_t = \sqrt{1-\beta_t}x_{t-1} + \sqrt{\beta_t}z_t, \quad z_t \sim \mathcal{N}(0, I)$

Aquí β_t controla la cantidad de ruido que se añade en cada paso.

Jonathan Ho et al. propusieron reformular la ecuación:

$$x_t = \sqrt{\bar{\alpha}_t}\, x_0 + \sqrt{1-\bar{\alpha}_t}\, \epsilon, \quad \epsilon \sim \mathcal{N}(0, I) \qquad \text{donde:} \quad \bar{\alpha}_t = \prod_{s=1}^{t} \beta_s$$

Reverse Process: En lugar de modelar directamente $p_\theta(x_{t-1}|x_t)$, Ho et al. reformularon el problema y propusieron entrenar la red neuronal para predecir el ruido ϵ en la ecuación del forward process.

La red neuronal aprende una función $\hat{\epsilon}_\theta(x_t, t)$ que estima el ruido presente en x_t: $\epsilon \approx \hat{\epsilon}_\theta(x_t, t)$

Entonces, el reverse process se define como:

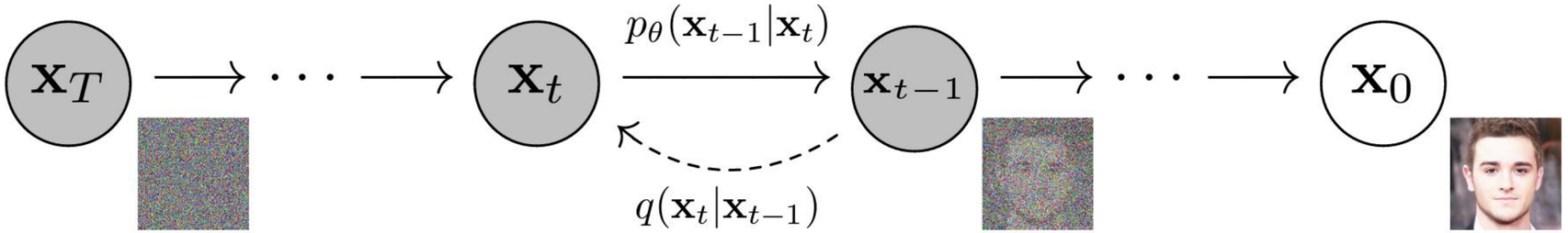
$$x_{t-1} = \frac{1}{\sqrt{1-\beta_t}}\left(x_t - \beta_t \hat{\epsilon}_\theta(x_t, t)\right) + \sigma_t z_t, \quad z_t \sim \mathcal{N}(0, I)$$

donde σ_t controla el ruido en la reconstrucción. $\sigma_t^2 = \frac{1-\bar{\alpha}_{t-1}}{1-\bar{\alpha}_t}\beta_t$

Jonathan Ho et al. (2020) "Denoising Diffusion Probabilistic Models". Advances in neural information processing systems, 33, 6840-6851.

DDPM

(Denoising Diffusion Probabilistic Models)



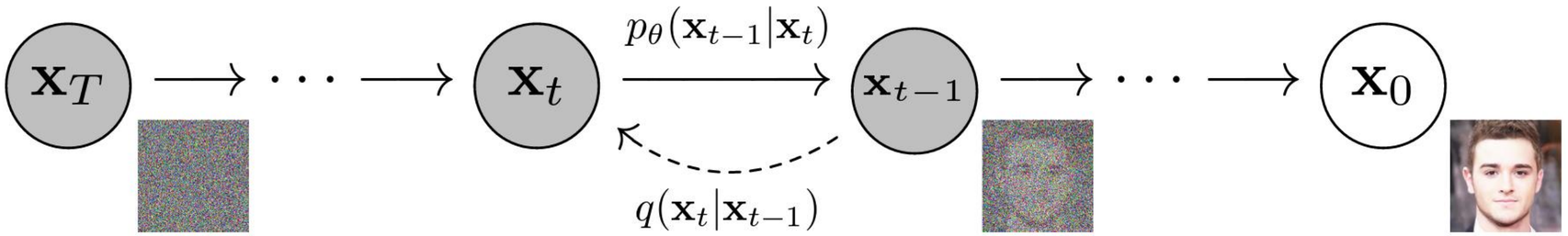
Entrenamiento: Dado que sabemos el ruido original ϵ que se usó en el forward process, podemos entrenar la red neuronal para predecir dicho ruido con una pérdida de error cuadrático medio (MSE):

$$\mathcal{L} = \mathbb{E}_{x_{\theta,\epsilon,t}}[||\epsilon - \hat{\epsilon}_\theta(x_t, t)||^2]$$

Jonathan Ho et al. (2020) "Denoising Diffusion Probabilistic Models". Advances in neural information processing systems, 33, 6840-6851.

DDPM

(Denoising Diffusion Probabilistic Models)



Algorithm 1 Training

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

Algorithm 2 Sampling

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

Jacob Devlin et al. (2018) "BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding". Proceedings of naacL-HLT (Vol. 1, p. 2).

UTEC
UNIVERSIDAD DE INGENIERÍA Y TECNOLOGÍA

2.

DALL-*E*

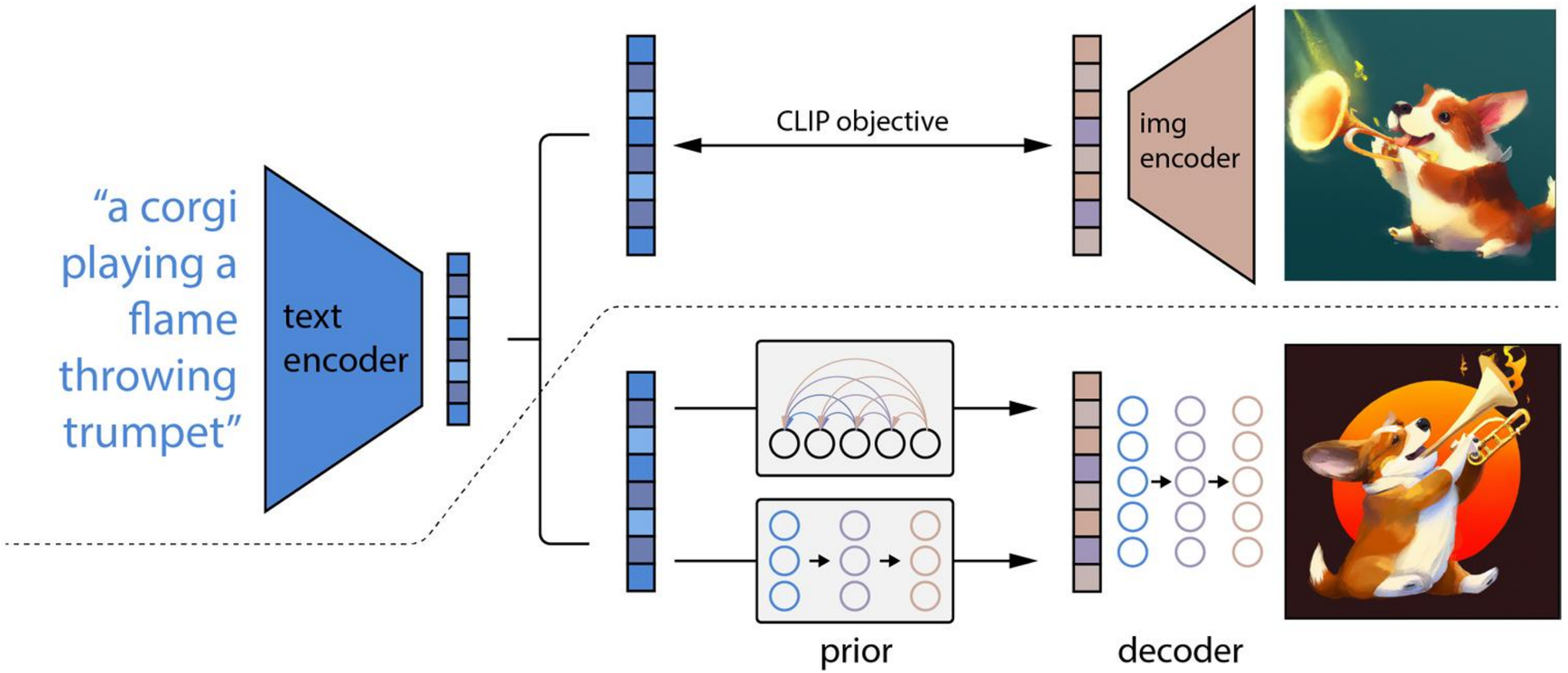
TRANSFORMATEC

GLIDE

(Guided Language to Image Diffusion for Generation and Editing)

Alex Nichol et al. (2022) "GLIDE: Towards Photorealistic Image Generation and Editing with Text-Guided Diffusion Models". arXiv preprint arXiv:2112.10741.

Dall-*E 2*



Aditya Ramesh et al. (2022) "Hierarchical Text-Conditional Image Generation with CLIP Latents". arXiv preprint arXiv:2204.06125, 1(2), 3.

UTEC
UNIVERSIDAD DE INGENIERÍA Y TECNOLOGÍA

3.

Latent *diffusion*

TRANSFORMATEC

LDM

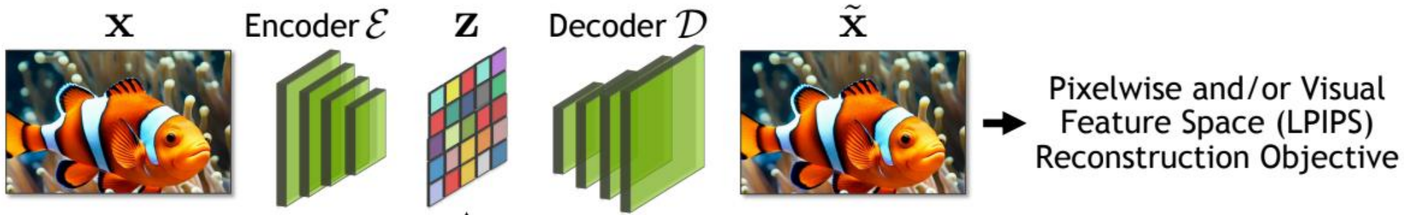
(Latent diffusion models)

Map Data into Compressed Latent Space. Train Diffusion Model efficiently in Latent Space.

- Stage 1:

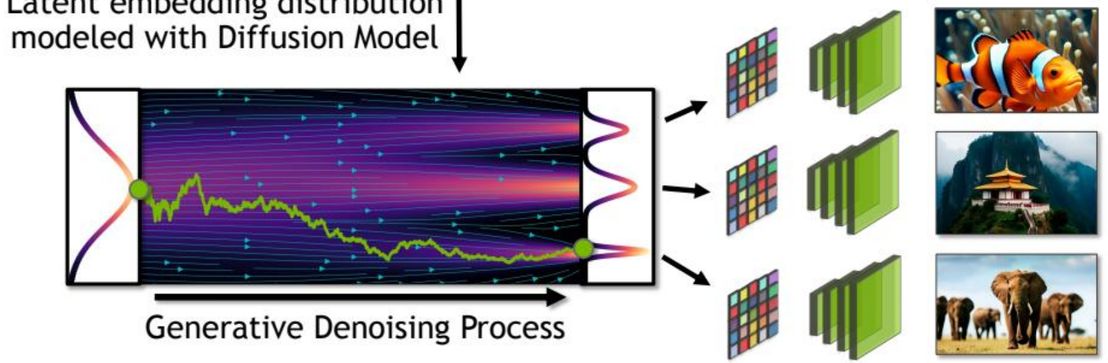
Train Autoencoder

$$\tilde{\mathbf{x}} = \mathcal{D}(\mathcal{E}(\mathbf{x}))$$



- Stage 2:

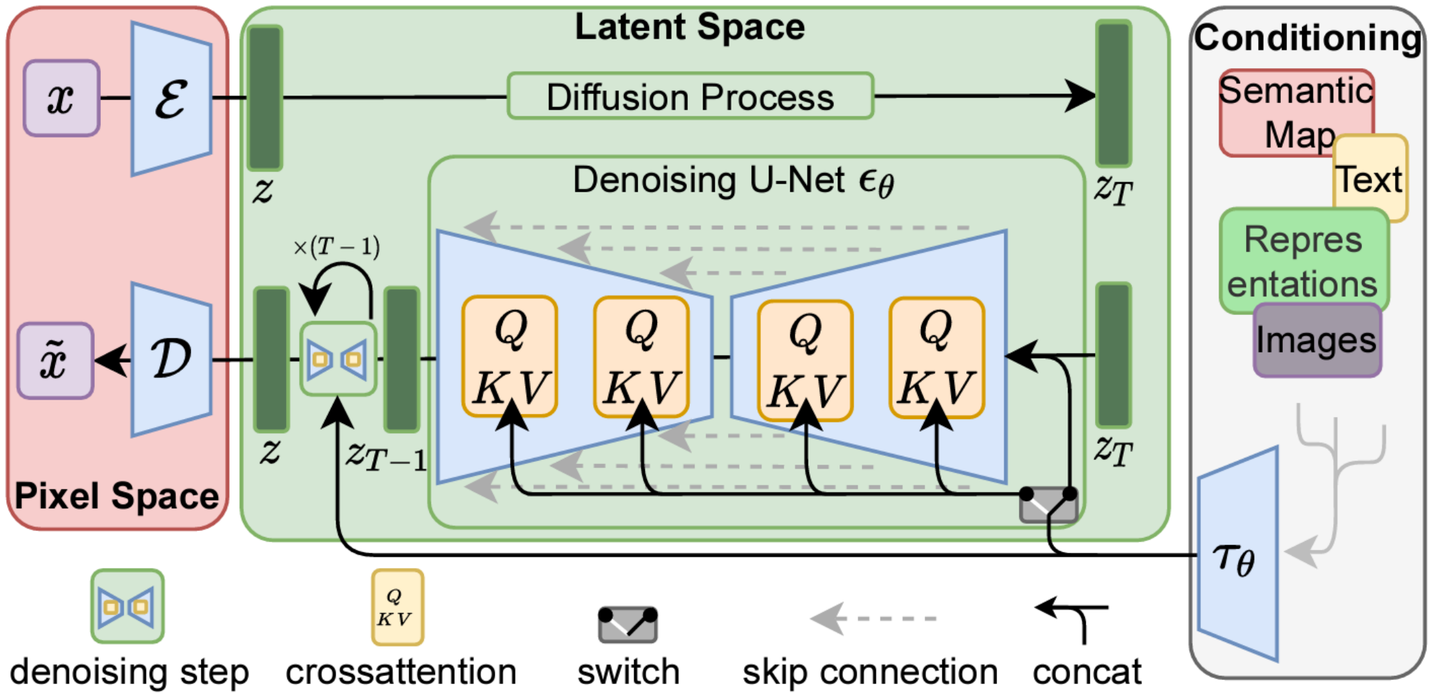
Train **Latent** Diffusion Model



Robin Rombach et al. (2022) "High-Resolution Image Synthesis with Latent Diffusion Models".
Proceedings of the IEEE/CVF conference on computer vision and pattern recognition (pp. 10684-10695).

LDM

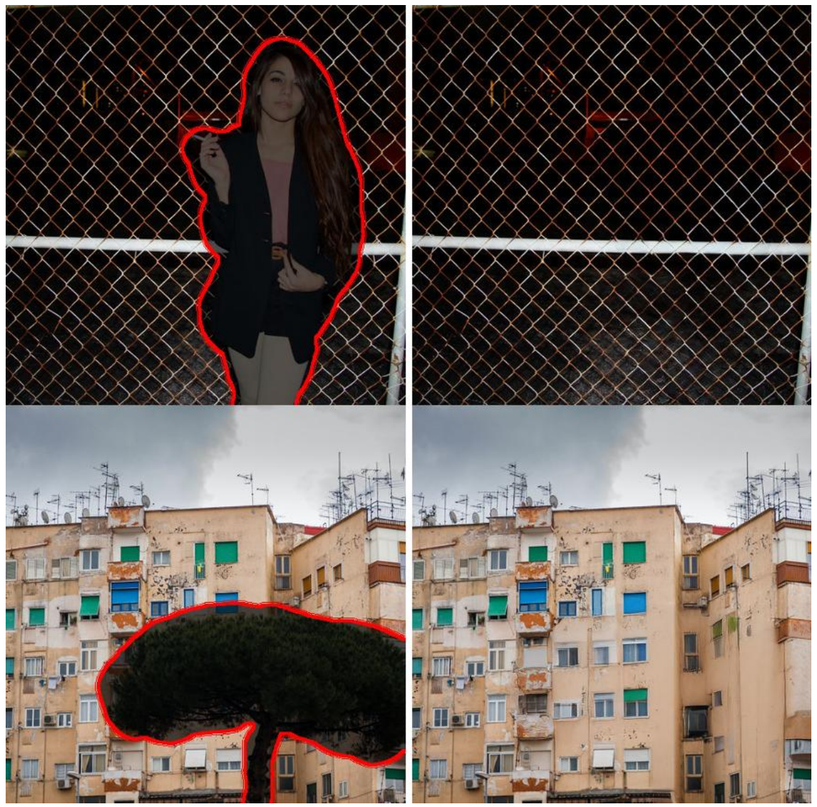
(Latent diffusion models)



$$L_{LDM} := \mathbb{E}_{\mathcal{E}(x),y,\epsilon\sim\mathcal{N}(0,1),t}\left[\|\epsilon - \epsilon_\theta(z_t, t, \tau_\theta(y))\|_2^2\right]$$

Robin Rombach et al. (2022) "High-Resolution Image Synthesis with Latent Diffusion Models". Proceedings of the IEEE/CVF conference on computer vision and pattern recognition (pp. 10684-10695).

LDM
(Latent diffusion models)



Robin Rombach et al. (2022) "High-Resolution Image Synthesis with Latent Diffusion Models". Proceedings of the IEEE/CVF conference on computer vision and pattern recognition (pp. 10684-10695).

LCM

(Latent Consistency Models)

Latent Consistency Models se basan en la idea de destilar el proceso inverso iterativo de DDPM en una única (o muy pocas) transformación(s) directa(s) que mapeen una muestra ruidosa en el espacio latente a la versión limpia.

Reconstrucción:

Dada una muestra z_t obtenida del forward process (en el espacio latente)

$$z_t = \sqrt{\bar{\alpha}_t} z_0 + \sqrt{1 - \bar{\alpha}_t}\epsilon, \quad \epsilon \sim \mathcal{N}(0, I)$$

el modelo debe predecir el latente original z_0. Es decir, queremos que $f_\theta(z_t, t) \approx z_0$

Simian Luo et al. (2023) "Latent Consistency Models: Synthesizing High-Resolution Images with Few-Step Inference". arXiv preprint arXiv:2310.04378.

LCM

(Latent Consistency Models)

Latent Consistency Models se basan en la idea de destilar el proceso inverso iterativo de DDPM en una única (o muy pocas) transformación(s) directa(s) que mapeen una muestra ruidosa en el espacio latente a la versión limpia.

Reconstrucción:

Dada una muestra z_t obtenida del forward process (en el espacio latente)

$$z_t = \sqrt{\bar{\alpha}_t} z_0 + \sqrt{1 - \bar{\alpha}_t}\epsilon, \quad \epsilon \sim \mathcal{N}(0, I)$$

el modelo debe predecir el latente original z_0. Es decir, queremos que $f_\theta(z_t, t) \approx z_0$

Consistency Models:

La idea es que, para cualquier par de tiempos t y s con $s < t$ (diferentes niveles de ruido), el mapeo debe ser consistente. Es decir, si definimos

$$z_t = \sqrt{\bar{\alpha}_t} z_0 + \sqrt{1 - \bar{\alpha}_t}\epsilon \quad \text{y} \quad z_s = \sqrt{\bar{\alpha}_s} z_0 + \sqrt{1 - \bar{\alpha}_s}\epsilon'$$

se desea que $f_\theta(z_t, t) \approx f_\theta(z_s, s)$.

En particular, fijando $s = 0$, la propiedad de consistencia nos dice que para cualquier t: $f_\theta(z_t, t) \approx z_0$

Simian Luo et al. (2023) "Latent Consistency Models: Synthesizing High-Resolution Images with Few-Step Inference". arXiv preprint arXiv:2310.04378.

LCM

(Latent Consistency Models)

Latent Consistency Models se basan en la idea de destilar el proceso inverso iterativo de DDPM en una única (o muy pocas) transformación(s) directa(s) que mapeen una muestra ruidosa en el espacio latente a la versión limpia.

Reconstruction loss:

A partir z_t (obtenido a partir de z_0 mediante el forward process) se espera predecir z_0:

$$\mathcal{L}_{rec} = E_{z_0,t,\epsilon}\left[\left\| f_\theta\left(\sqrt{\bar{\alpha}_t} z_0 + \sqrt{1-\bar{\alpha}_t}\epsilon, t\right) - z_0 \right\|^2\right]$$

Simian Luo et al. (2023) "Latent Consistency Models: Synthesizing High-Resolution Images with Few-Step Inference". arXiv preprint arXiv:2310.04378.

LCM

(Latent Consistency Models)

Latent Consistency Models se basan en la idea de destilar el proceso inverso iterativo de DDPM en una única (o muy pocas) transformación(s) directa(s) que mapeen una muestra ruidosa en el espacio latente a la versión limpia.

Reconstruction loss:

A partir z_t (obtenido a partir de z_0 mediante el forward process) se espera predecir z_0:

$$\mathcal{L}_{rec} = E_{z_0,t,\epsilon}\left[\left\|f_\theta\left(\sqrt{\bar{\alpha}_t}z_0 + \sqrt{1-\bar{\alpha}_t}\epsilon, t\right) - z_0\right\|^2\right]$$

Consistency loss:

Buscamos que el mapeo sea invariante al nivel de ruido, es decir, que la predicción del modelo sea la misma si se evalúa en diferentes tiempos. Formalmente, para dos tiempos t y s (con $s < t$) definimos:

$$\mathcal{L}_{cons} = E_{z_0,t,s,\epsilon,\epsilon'}\left[\left\|f_\theta\left(\sqrt{\bar{\alpha}_t}z_0 + \sqrt{1-\bar{\alpha}_t}\epsilon, t\right) - f_\theta\left(\sqrt{\bar{\alpha}_s}z_0 + \sqrt{1-\bar{\alpha}_s}\epsilon', s\right)\right\|^2\right]$$

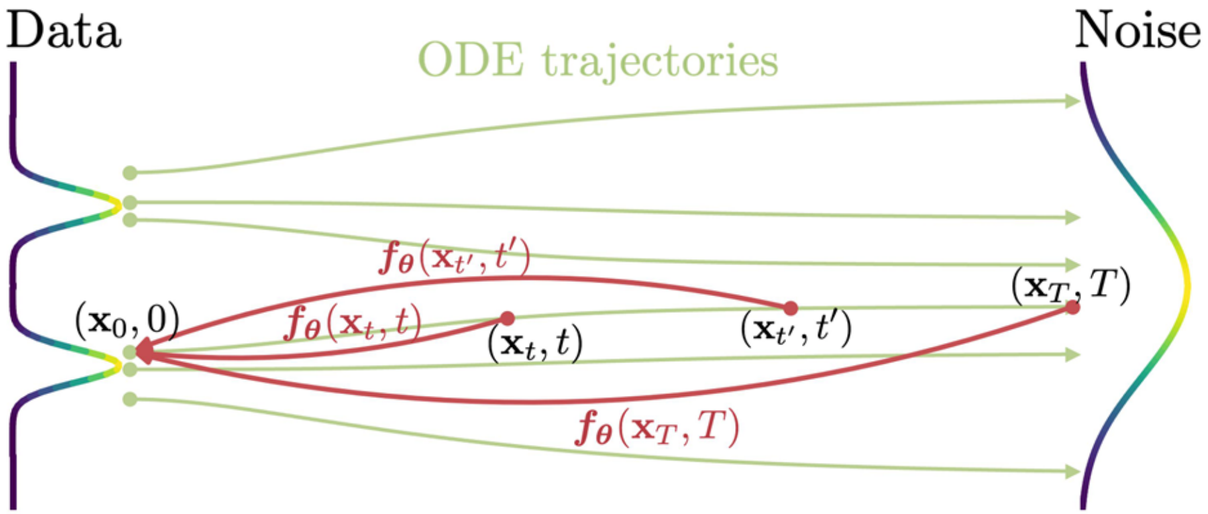
Esta pérdida fuerza a que, sin importar el nivel t en el que se observe la muestra ruidosa, el modelo produzca la misma estimación de z_0.

Total loss:

$$\mathcal{L} = \mathcal{L}_{rec} + \lambda\mathcal{L}_{cons}$$

Simian Luo et al. (2023) "Latent Consistency Models: Synthesizing High-Resolution Images with Few-Step Inference". arXiv preprint arXiv:2310.04378.

LCM
(Latent Consistency Models)



Simian Luo et al. (2023) "Latent Consistency Models: Synthesizing High-Resolution Images with Few-Step Inference". arXiv preprint arXiv:2310.04378.

UTEC
UNIVERSIDAD DE INGENIERÍA Y TECNOLOGÍA

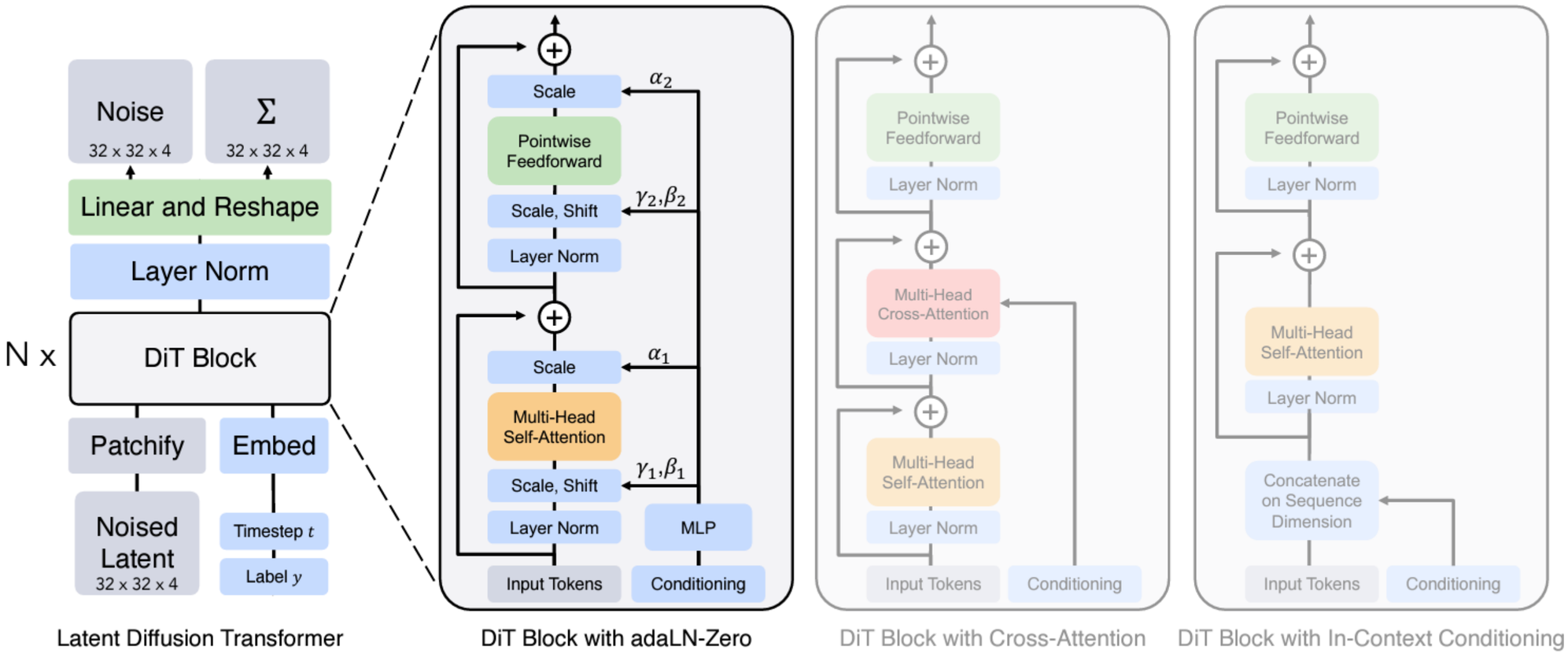
4.

Diffusion *Transformer*

TRANSFORMATEC

DiT

(Diffusion Transformer)



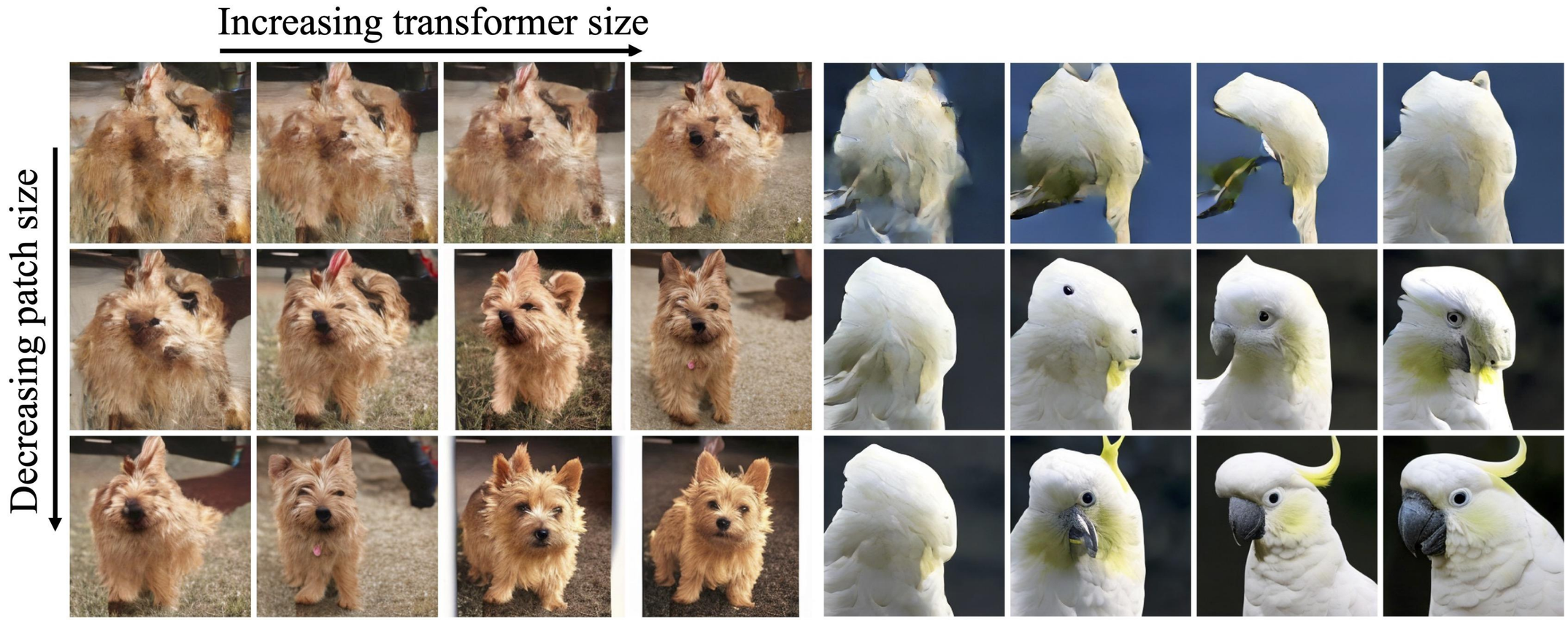
Latent Diffusion Transformer | DiT Block with adaLN-Zero | DiT Block with Cross-Attention | DiT Block with In-Context Conditioning

TRANSFORMATEC

William Peebles and Saining Xie (2023) "Scalable Diffusion Models with Transformers". Proceedings of the IEEE/CVF International Conference on Computer Vision (pp. 4195-4205).

DiT

(Diffusion Transformer)



William Peebles and Saining Xie (2023) "Scalable Diffusion Models with Transformers".
Proceedings of the IEEE/CVF International Conference on Computer Vision (pp. 4195-4205).

DiT

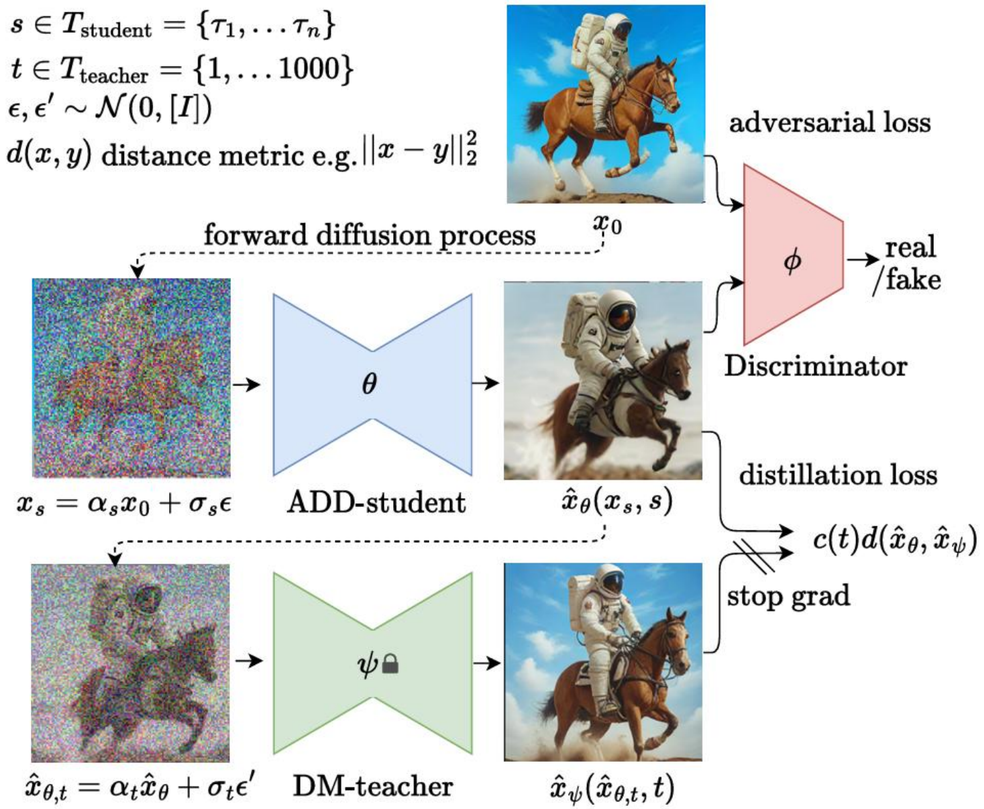
(Diffusion Transformer)

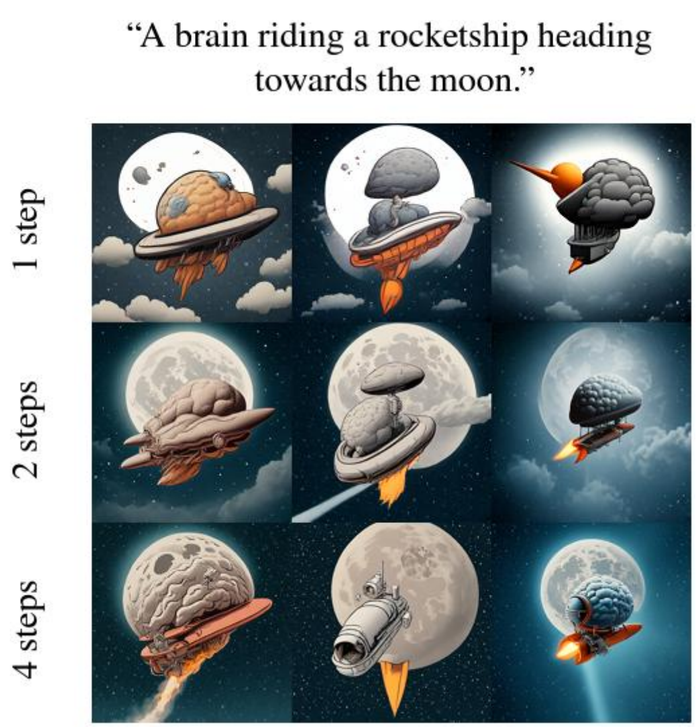
Walking through DiT-XL/2's (512x512) learned label embedding space

William Peebles and Saining Xie (2023) "Scalable Diffusion Models with Transformers".
Proceedings of the IEEE/CVF International Conference on Computer Vision (pp. 4195-4205).

ADD

(Adversarial Diffusion Distillation)

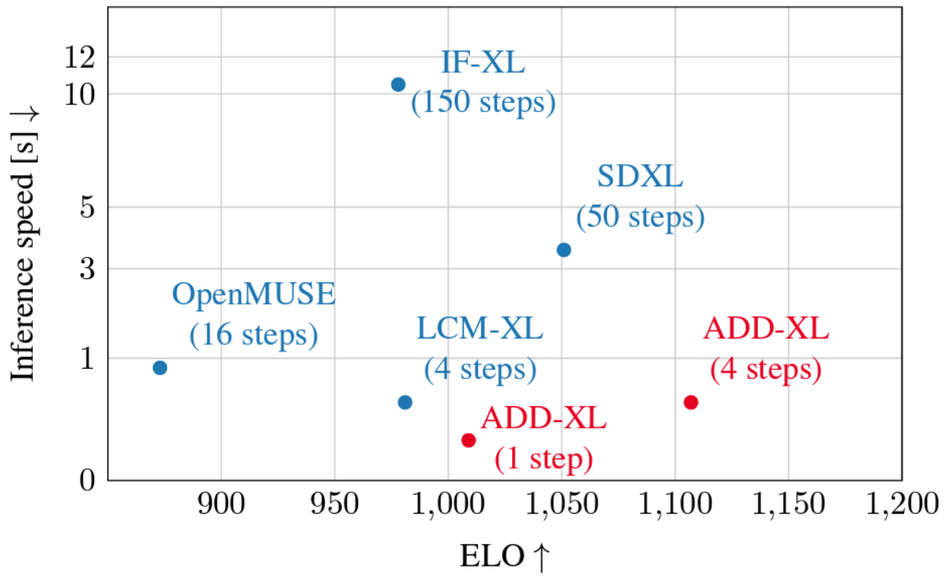




Axel Sauer et al. (2023) "Adversarial Diffusion Distillation". Proceedings of naacL-HLT (Vol. 1, p. 2).

ADD

(Adversarial Diffusion Distillation)



Axel Sauer et al. (2023) "Adversarial Diffusion Distillation". Proceedings of naacL-HLT (Vol. 1, p. 2).

UTEC
UNIVERSIDAD DE INGENIERÍA Y TECNOLOGÍA

> Reinventa el mundo <

GRACIAS

Victor Flores Benites

TRANSFORMATEC