Stable Diffusion 模型介紹

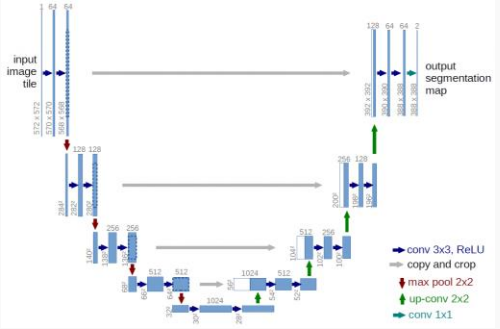
鄧子洋 TENG, TZU-YANG

01
PART

架構

Stable Diffusion 訓練流程介紹

1. VAE Encoder (圖像x 經過壓縮成 latent z, downsampling) (Why? 降低計算資源)
2. Diffusion Process (Forward process, 壓縮過後的圖像z 經過t次加噪, 形成z_t)
3. Denoising U-Net (逐漸去噪, 預測乾淨的latent, 其實這過程是預測latent中的噪聲)
4. VAE Decoder (將最後已經去噪的latent特徵 還原成真實圖片, 生成最後圖像)



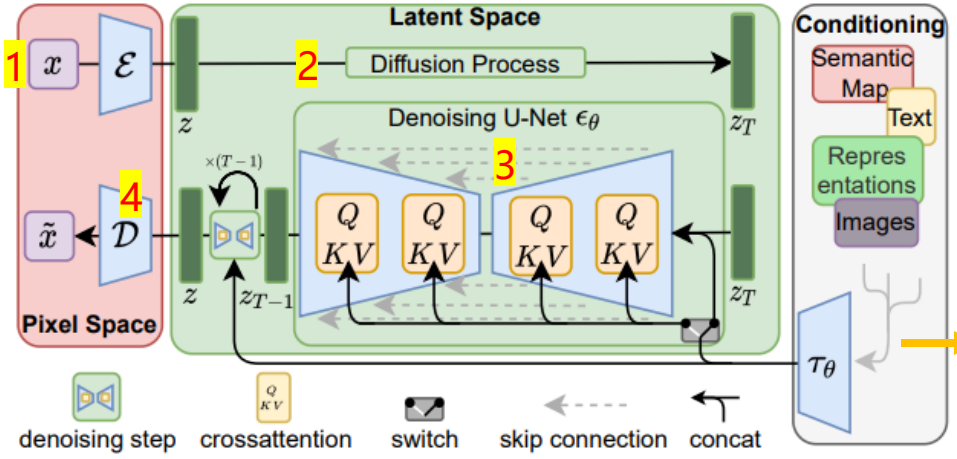


Figure 3. We condition LDMs either via concatenation or by a more general cross-attention mechanism. See Sec. 3.3

Contrastive Language-Image Pre-Training(CLIP)

- 400 million image-text pairs training

Text prompt 經過 CLIP text-encoder 轉為 embedding，再透過crossattention傳入U-Net 使得模型可以根據使用者輸入描述, 生成對應風格圖片

Guidance Scale (透過調整有text與無text圖片的噪聲比, 來調整噪聲, 進而控制結果 (還有分正,負的提示詞)

02
PART

Diffusion剖析

What is Diffusion Model ?

- 核心:
  - 在latent空間加躁, 訓練用U-Net去噪
- Forward process (diffusion process) : 逐步加入雜訊，使圖像變 (圖片->雜訊)
- Reverse process : 逐漸去除噪聲, 還原真實圖片 (雜訊->圖片)

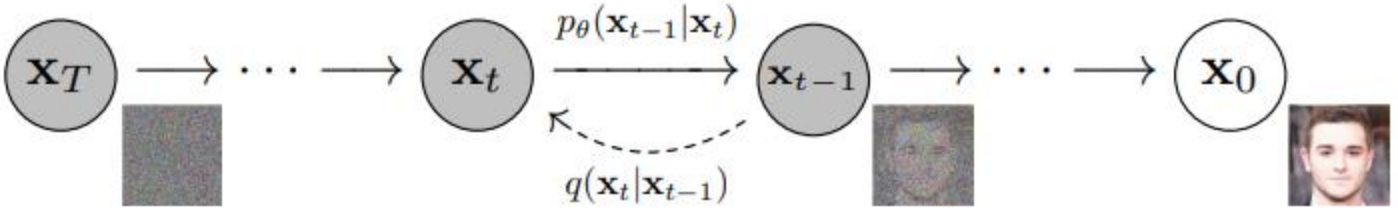


Figure 2: The directed graphical model considered in this work.

What is Diffusion Model ?

- 其實Diffusion Model是在預測噪聲, 而非預測圖片!!!! (學會去噪)
  - 透過 帶有雜訊的圖片 減去 預測的噪聲 來取得原始圖片應該長怎樣
- Inference 其實就是將噪聲圖片 透過迭代的去除雜訊 最後生成期望的圖片

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

Noise predictor

帶有雜訊的圖片

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** $\mathbf{x}_0$

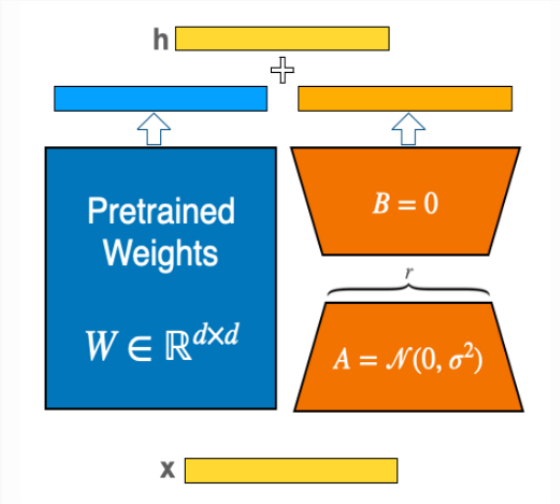
LoRA微調

Why use LoRA?

- 因為原始模型大的可怕, 且訓練好的模型大多是通才（擅於生成各種語意的圖像）
- 因此，PEFT 微調模型的技術就演變而來, 讓模型變成專才, 學會某種我們期望的特徵
  - Dreambooth 及 LoRA 則是常見的方法
- Dreambooth
  - 在少量訓練基礎上, 還原原始圖像的特徵, 但會調整所有參數, 時間較長
  - 微調訓練出來的模型比較強大, 較能學會到細節的圖像特徵, 模型較大
- LoRA（Low-Rank-Adaption）
  - 在原始的計算層中, 多插入計算層, 不影響原始模型參數
  - 冷凍原始模型參數, 透過訓練新的模型參數
  - 利用降維度的訓練, 只訓練較少的參數, 模型較小

How ?

- Diffusion 當中的核心是 U-Net, 裡面最重要的部分是: Attetion, Convolution 層
- Attetion層當中有包含了Cross-attetion, 這涉及到了prompt的效果
  - 也就是noise predictor 如何透過 prompt-embedding 去改變它所預測的噪聲
- LoRA 透過修改cross-attention的參數, 讓模型在特定prompt下產生特定風格
- 特點:
  - 計算量降低, 同時保有模型原本的能力
  - 透過更新少量 權重矩陣 內部參數, 進而改變模型的原先矩陣



How does LoRA work?

- **做法:**
  - **假使原先cross-attention是一個超大矩陣(1000*2000), 訓練參數量= 2百萬個參數**
  - **那現在我們只訓練A (1000*2), B (2*2000)矩陣, 則訓練量=6000個參數即可**

原本的參數更新方法為：

$$h = W \times x \text{(注意：} h \text{和} x \text{是} d \text{維的向量，} W \in d \times d \text{的矩陣)}$$

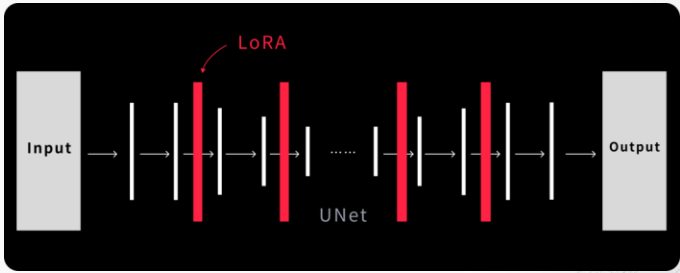
故每次都需要更新 W矩陣內$d \times d$個參數。但在 LoRA 的方法中，我們將W矩陣固定，並使用如下更新方式，利用W'取代原先的W：

$$W' = W + \Delta(W), \Delta(W) = A \times B$$

在這裡，矩陣 A 的維度為 $d \times r$，矩陣 B 的維度為 $r \times d$。通過矩陣乘法$A \times B$，相乘的結果為:

$$A^{d \times r} \times B^{r \times d} = \Delta W^{d \times d}$$

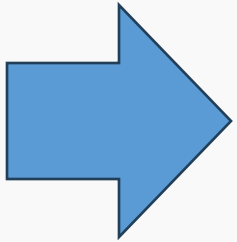
我們透過將參數空間降維，然後再提升回原來的維度。這樣，我們雖然犧牲了一部分資訊，但依然能夠進行有效的微調。



04
PART

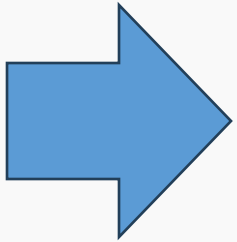
測試LoRA結果

測試LoRA – 川菜回鍋肉



測試LoRA – 握手 (perfect hand, detailed hand)

donald trump shake hand with obama which focus on perfect hand.



THANKS