Latent Diffusion Model

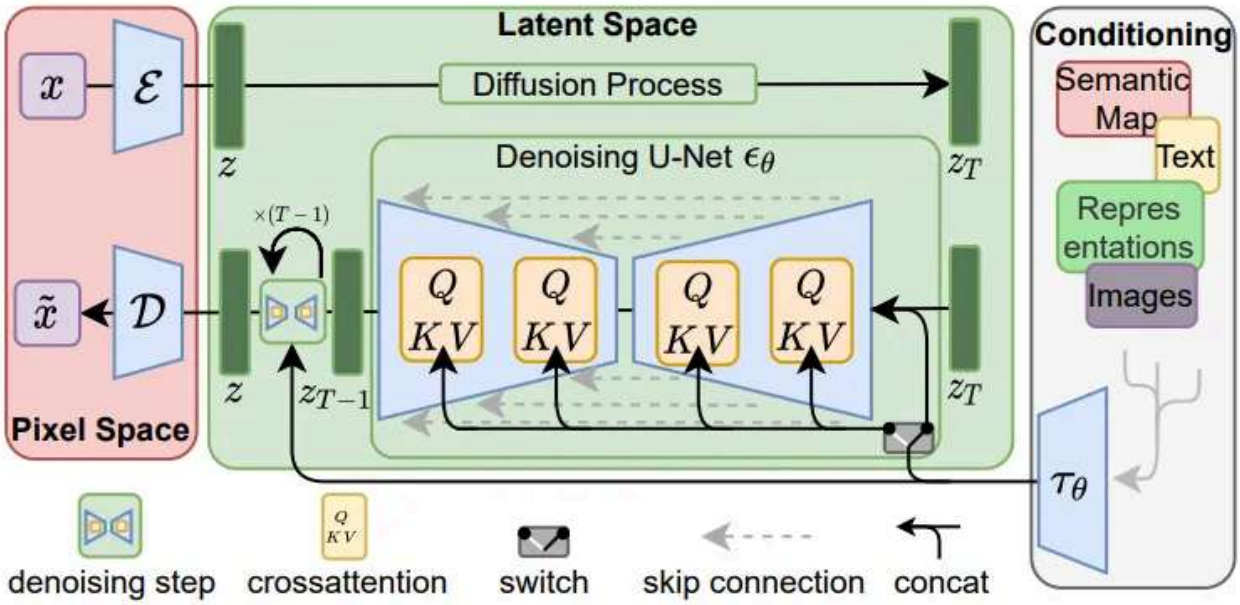
R Rich
Just now

#LDMs

一般的DMs(diffusion Model)所需的算力是需要非常大的，如GPU、Memory或訓練時的parameters，也要耗費非常多的時間。

因此Latent Diffusion Model想要把diffusion的步驟在Latent Space上操作同時也能有更有彈性的加入conditions，藉此降低整體的算力和花費。

> *LDMs*



分為Pixel Space、Latent Space、Conditioning 三個部分

一開始圖片先透過encoder轉換成在Latent Space的向量Z(Latent Vector)，之後再做Diffusion Process也就是加噪音(gaussian)，得到一個含噪音的新向量ZT(Noise Latent Vector)。之後再一步一步Denoising(Reverse Diffusion Step)得到結果的Latent Vector，再透過decoder成結果圖(原圖)，而在denosing的過程中可能會加一些condition。

Latent Space 操作

當輸入圖片透過VAE(encoder)成Latent Vector後，會開始加入gaussian noise(t1~tN)noise慢慢變多也就是ZT，之後再透過Attention U-Net預測此noise vector(ti)的noise，之後再相減就能得到noise vector(ti-1)，重複此步驟直到t0(noise完全去除)，再decoder回去原圖。

Condition

Condition(text, image, etc)會透過Condition Encoder 成condition embedding 再結合到Attention U-Net(cross attention)

Loss function不含condition

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

Loss function含condition

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

Diffusion Model在Pixel Space的Loss Function

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

> *Cross Attention*

$$\text{Attention}(Q, K, V) = \text{softmax}\left(\frac{QK^T}{\sqrt{d}}\right) \cdot V$$

$$Q = W_Q^{(i)} \cdot \varphi_i(z_t),\ K = W_K^{(i)} \cdot \tau_\theta(y),\ V = W_V^{(i)} \cdot \tau_\theta(y)$$

Q是在Latent Space Vector的image做線性轉換，K 、V則是condition embedding做線性變換，W是線性變換的矩陣，藉此方式就能將condition加入其中。