Imagen

A Preprint

1 Main

Imagen structure consists of:

- Frozen T5-XXL(a very large pre-trained language model that is used as a text encoder within the Imagen model)
- Diffusion model with 64 ×64 shape
- Diffusion model with output shape 256×256
- Diffusion model with output shape 1024×1024

2 Efficient U-Net

- In our setup, we use a larger number of residual blocks at lower resolutions—for example, 8 residual blocks—compared to the typical 2–3 blocks used in standard U-Net architectures. We observe that scaling the skip connections by a factor of $1/\sqrt{2}$, following the approach in, substantially accelerates convergence during training.
- In conventional U-Net architectures, the downsampling operation follows the convolution layers in the downsampling block, while in the upsampling block, the upsampling precedes the convolution. We reverse this order in both downsampling and upsampling blocks. This change significantly improves the forward pass speed of the U-Net, and we find that it does not negatively affect performance.

U-net is also modified using Cross-Attention in order to make better conditioning. They suggest concatenating embedding sequence and the text to the key-value pairs of each self-attention layer in the base 64×64 and $64 \times 64 \rightarrow 256 \times 256$ models. For their $256 \times 256 \rightarrow 1024 \times 1024$ model, since there is no self attention layers, authors suggest adding explicit cross-attention layers to attend over the text embeddings.

3 Diffusion

Diffusion models can be viewed as latent variable models where the latent variables $z = \{z_t \mid t \in [0, 1]\}$ evolve according to a forward process $q(z \mid x)$, initialized at data $x \sim p(x)$. This forward process is a Gaussian process with a Markov structure. Specifically, it satisfies:

$$q(z_t \mid x) = \mathcal{N}(z_t; \alpha_t x, \sigma_t^2 I), \quad q(z_t \mid z_s) = \mathcal{N}\left(z_t; \frac{\alpha_t}{\alpha_s} z_s, \sigma_{t|s}^2 I\right),$$

where $0 \leq s < t \leq 1$, and the variance is given by $\sigma_{t|s}^2 = (1 - e^{\lambda_t - \lambda_s})\sigma_t^2$. The parameters α_t and σ_t define a differentiable noise schedule such that the log signal-to-noise ratio $\lambda_t = \log(\alpha_t^2/\sigma_t^2)$ decreases as t increases, leading to $q(z_1) \approx \mathcal{N}(0, I)$.

To generate data, the model learns the reverse of this process. This reversal is framed as a denoising problem: given $z_t \sim q(z_t \mid x)$, the model learns to predict an estimate $\hat{x}_\theta(z_t, \lambda_t, c) \approx x$, where c is an optional conditioning variable (such as a text embedding or a low-resolution image) paired with x.

Training is performed by minimizing a weighted squared error:

$$\mathbb{E}_{\epsilon,t}\left[w(\lambda_t)\left\|\hat{x}_\theta(z_t,\lambda_t,c)-x\right\|^2\right],$$

where $t\sim\mathcal{U}([0,1])$, $\epsilon\sim\mathcal{N}(0,I)$, and $z_t=\alpha_t x+\sigma_t\epsilon$. This formulation can be interpreted as optimizing a variational lower bound or as a form of denoising score matching.

A common parameterization used in practice involves predicting ϵ instead of x, where

$$\hat{x}_\theta(z_t,\lambda_t,c)=\frac{z_t-\sigma_t\epsilon_\theta(z_t,\lambda_t,c)}{\alpha_t}.$$

The model is trained using a squared error loss on ϵ_θ with t sampled according to a cosine schedule. This results in a scaled estimate of the gradient of the log-density:

$$\epsilon_\theta(z_t,\lambda_t,c)\approx-\sigma_t\nabla_{z_t}\log p(z_t\mid c).$$

To generate a sample, one begins with $z_1\sim\mathcal{N}(0,I)$ and applies either a stochastic ancestral sampler or a deterministic update rule such as DDIM. The DDIM update is given by:

$$z_s=\alpha_s\hat{x}_\theta(z_t,\lambda_t,c)+\frac{\sigma_s}{\sigma_t}\left(z_t-\alpha_t\hat{x}_\theta(z_t,\lambda_t,c)\right),$$

where $s<t$ are chosen from a uniform sequence decreasing from 1 to 0.

The ancestral sampler can be derived from the reverse of the forward process:

$$q(z_s\mid z_t,x)=\mathcal{N}(z_s;\tilde{\mu}_{s|t}(z_t,x),\tilde{\sigma}^2_{s|t}I),$$

with

$$\tilde{\mu}_{s|t}(z_t,x)=e^{\lambda_t-\lambda_s}\frac{\alpha_s}{\alpha_t}z_t+(1-e^{\lambda_t-\lambda_s})\alpha_s x,\quad\tilde{\sigma}^2_{s|t}=(1-e^{\lambda_t-\lambda_s})\sigma^2_s.$$

The corresponding stochastic update rule is:

$$z_s=\tilde{\mu}_{s|t}(z_t,\hat{x}_\theta(z_t,\lambda_t,c))+\sqrt{\tilde{\sigma}^2_{s|t}(1-\gamma)+\sigma^2_{t|s}\gamma}\cdot\epsilon,$$

where $\epsilon\sim\mathcal{N}(0,I)$, and γ controls the level of stochasticity in the generation process.

4 Upsampling

It is also important to note that authors train 3 different diffusion models:

- 64×64
- 256×256
- 1024×1024

Each time since shape 256 they use low resolution image is condition for higher one which improves results and saves memory.

Classifier-Free Guidance

The model is trained to handle both conditional and unconditional generation tasks, and guidance is applied by interpolating between these two outputs at inference time.

During training, the model learns to predict the noise added during the forward diffusion process. Let x_t be a noisy image at timestep t, and let c be an optional conditioning input (e.g., a text prompt). The model is trained to predict the original noise ϵ from x_t, either with conditioning c or without it. With some probability, the conditioning input is dropped during training, allowing the model to learn both conditional and unconditional denoising:

$$\mathbb{E}_{x,\epsilon,t,c}\left[\|\epsilon_\theta(x_t,t,c)-\epsilon\|^2\right],$$

where c is either the actual condition or a null input $\varnothing$.

At inference time, both the conditional and unconditional predictions are computed:

$$\epsilon_{\text{cond}}=\epsilon_\theta(x_t,t,c),\quad \epsilon_{\text{uncond}}=\epsilon_\theta(x_t,t,\varnothing).$$

These are combined using a guidance scale $w \geq 1$ to obtain the guided noise estimate:

$$\epsilon_{\text{guided}}=\epsilon_{\text{uncond}}+w\cdot(\epsilon_{\text{cond}}-\epsilon_{\text{uncond}}).$$

This equation increases the influence of the conditioning input, encouraging the model to generate outputs that better match the desired condition. The choice of w controls the strength of the guidance: $w = 1$ corresponds to standard conditional sampling, while higher values of w increase the emphasis on the condition at the potential cost of sample diversity.

Classifier-free guidance is simple to implement, works for any conditioning type, and has been shown to significantly improve alignment between generated outputs and conditioning inputs.