Comprehensive Study Report: Diffusion Models and Transformers

Pratham Chheda

April 6, 2025

Abstract

This report presents a comprehensive investigation into Diffusion Transformers (DiT), exploring their capabilities and characteristics as a modern alternative to traditional CNN-based diffusion models. We analyze the impact of key hyperparameters like Classifier-Free Guidance (CFG) and the number of sampling steps on generation quality using pre-trained models. Subsequently, we investigate performance enhancements by integrating efficient attention mechanisms, specifically comparing the baseline attention with implementations leveraging the xformers library and Sliding Window Attention (SWA). Two DiT models (baseline full attention vs. SWA) were trained from scratch on a landscape dataset for direct comparison. Furthermore, we implement and evaluate the CLIP Mean Maximum Discrepancy (CMMD) metric, alongside variants using SigLIP and ALIGN embeddings, comparing their effectiveness against the standard Fréchet Inception Distance (FID) for assessing generative model performance. Our findings highlight the trade-offs inherent in diffusion model parameters, demonstrate significant computational improvements via xformers, quantify the performance difference between full attention and SWA, and provide insights into the utility and nuances of different embedding-based evaluation metrics.

Contents

1 Theoretical Foundation: Denoising Diffusion Probabilistic Models

1.1 Introduction to Diffusion Models

Diffusion models have emerged as a powerful class of generative models capable of producing high-fidelity data across various domains, including images, audio, and more. Inspired by principles from non-equilibrium thermodynamics, these models learn to reverse a gradual process of noise addition. The core idea involves defining a *forward process* where data is progressively corrupted with Gaussian noise over a series of timesteps, and a *reverse process* where a neural network learns to iteratively remove this noise, starting from pure noise, to generate a sample resembling the original data distribution.

1.2 Mathematical Framework

The forward process, denoted by q, is a fixed Markov chain that adds Gaussian noise according to a variance schedule $\beta_1, ..., \beta_T$. The distribution at timestep t given the state at $t-1$ is:

$$q(x_t|x_{t-1}) = \mathcal{N}(x_t; \sqrt{1-\beta_t}x_{t-1}, \beta_t\mathbf{I}) \tag{1}$$

A useful property is that we can sample x_t directly from the original data x_0 using $\alpha_t = 1 - \beta_t$ and $\bar{\alpha}_t = \prod_{i=1}^{t} \alpha_i$:

$$q(x_t|x_0) = \mathcal{N}(x_t; \sqrt{\bar{\alpha}_t}x_0, (1-\bar{\alpha}_t)\mathbf{I}) \tag{2}$$

As $T \to \infty$, x_T approaches an isotropic Gaussian distribution.

The reverse process, p_θ, aims to learn the transition $p_\theta(x_{t-1}|x_t)$, which is also modeled as a Gaussian:

$$p_\theta(x_{t-1}|x_t) = \mathcal{N}(x_{t-1}; \mu_\theta(x_t, t), \Sigma_\theta(x_t, t)) \tag{3}$$

The model is trained to predict the noise ϵ added at timestep t, using a simplified objective function derived from the variational lower bound (VLB) on the data log-likelihood:

$$L_{simple}(\theta) = \mathbb{E}_{t,x_0,\epsilon}\left[\|\epsilon - \epsilon_\theta(\sqrt{\bar{\alpha}_t}x_0 + \sqrt{1-\bar{\alpha}_t}\epsilon, t)\|^2\right] \tag{4}$$

where t is uniformly sampled from $\{1, ..., T\}$, $x_0 \sim q(x_0)$, and $\epsilon \sim \mathcal{N}(0, \mathbf{I})$. ϵ_θ is the neural network (often a U-Net or Transformer) parameterized by θ.

1.3 Diffusion Transformers (DiT)

While traditional diffusion models like Stable Diffusion commonly employ U-Net architectures based on Convolutional Neural Networks (CNNs), these backbones can face limitations in scalability, particularly concerning model size and computational demands. The Diffusion Transformer (DiT) architecture was proposed to address this by replacing the U-Net backbone with a Transformer. Transformers, known for their success in natural language processing and vision (Vision Transformers, ViT), offer excellent scaling properties.

DiT operates on latent representations of images (obtained via a Variational Autoencoder, VAE), similar to Latent Diffusion Models. The latent space is tokenized into a sequence of patches. These patches, along with timestep and optional class conditioning information, are processed by a series of standard Transformer blocks incorporating self-attention. This allows DiT to potentially model long-range dependencies more effectively than CNNs and scale more efficiently with increased model parameters and data.

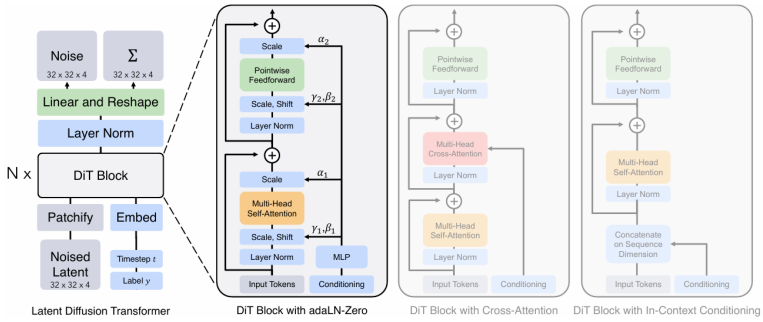


Figure 1: Conceptual overview of the DiT architecture. Latent image patches are processed by Transformer blocks, incorporating time and class conditioning, to predict the noise component for denoising.

1.4 Classifier-Free Guidance (CFG)

Classifier-Free Guidance (CFG) is a widely used technique to enhance sample quality and control the influence of conditioning (e.g., class labels, text prompts) during diffusion model sampling, without needing an explicitly trained classifier. It involves training the diffusion model occasionally with the conditioning information dropped (represented by a null token $\emptyset$). During inference, the model makes two predictions: one conditional $\epsilon_\theta(x_t, c)$ and one unconditional $\epsilon_\theta(x_t, \emptyset)$. The final noise prediction is extrapolated from these two:

$$\hat{\epsilon}_\theta(x_t, c) = \epsilon_\theta(x_t, \emptyset) + w \cdot (\epsilon_\theta(x_t, c) - \epsilon_\theta(x_t, \emptyset)) \tag{5}$$

This can be rewritten as:

$$\hat{\epsilon}_\theta(x_t, c) = (1 - w) \cdot \epsilon_\theta(x_t, \emptyset) + w \cdot \epsilon_\theta(x_t, c) \tag{6}$$

Here, w is the guidance scale.

- $w = 0$ yields unconditional generation.
- $w = 1$ corresponds to standard conditional generation.
- $w > 1$ amplifies the conditioning signal, pushing the generation further in the direction of the condition c. This often improves adherence to the condition and perceived sample quality up to a point, but excessive values can lead to oversaturation or artifacts, sacrificing diversity.

1.5 Implementation Strategy

The experiments in this report followed these steps:

1. Utilized the provided DiT repository and pre-trained ImageNet models (DiT-XL/2) for initial parameter sensitivity analysis (Task 1).
2. Implemented and benchmarked an efficient attention mechanism using the xformers library against the baseline DiT attention (Task 2a).
3. Implemented Sliding Window Attention (SWA) and trained two DiT-B/4 models (one with standard full attention, one with SWA) from scratch on a landscape dataset without class conditioning for comparison (Task 2b).

4. Developed and applied evaluation metrics, including FID and CMMD (using CLIP, SigLIP, and ALIGN embeddings), to compare the trained models (Task 3).

The VAE used for encoding/decoding was 'stabilityai/sd-vae-ft-ema'. All experiments involving training or evaluation used PyTorch and relevant libraries like 'transformers', 'diffusers', 'pytorch-fid', and 'torch-fidelity'.

2 Task 1: Parameter Analysis on Pre-trained DiT

2.1 Setup and Methodology

The initial experiments were conducted using the pre-trained DiT-XL/2 model designed for 256x256 image generation, as provided in the original DiT repository's notebook (`run_DiT.ipynb`). This model was trained conditionally on ImageNet classes. The setup involved loading the pre-trained DiT weights and the standard VAE ('stabilityai/sd-vae-ft-ema'). Generation was performed using the DDPM sampler provided in the repository.

2.2 Classifier-Free Guidance (CFG) Analysis

We investigated the effect of the CFG scale on the generated images by running the sampling process with identical seeds but varying the `cfg_scale` parameter passed to the sampler. Specifically, we compared `cfg_scale = 0` (no guidance) and `cfg_scale = 10` (strong guidance, chosen as a representative high value).

2.2.1 Observations and Explanation

Setting `cfg_scale = 0` effectively disables the influence of the class label provided during sampling. The generated images (Figure 2, left panel corresponding to CFG 0 in results PDF page 1) exhibited greater diversity and sometimes more abstract or dream-like qualities. While potentially creative, they often lacked strong resemblance to the target class, with objects appearing less defined or merging into the background. This is expected, as the generation relies solely on the model's learned unconditional distribution ($\epsilon_\theta(x_t, \emptyset)$).

Conversely, setting `cfg_scale = 10` significantly amplified the influence of the class label. The resulting images (Figure 2, right panel corresponding to CFG 10 in results PDF page 1) showed strong adherence to the target class features. Objects were well-defined, and the overall composition strongly reflected stereotypical representations of the class. However, this came at the cost of diversity; samples generated with the same class label tended to look more similar to each other compared to the `cfg_scale = 0` case. This aligns with the CFG formula, where a large w heavily weighs the conditional prediction $\epsilon_\theta(x_t, c)$, effectively pushing the sampling process strongly towards the learned conditional manifold.

The choice of CFG scale represents a trade-off between sample fidelity (adherence to the condition) and diversity. Moderate values (typically 4–8) are often found to provide a good balance.

CFG Scale = 0

CFG Scale = 10

Figure 2: Comparison of DiT image generation with CFG=0 (top) vs CFG=10 (bottom) using identical seeds and class labels. Note the increased class fidelity but potentially reduced diversity with the higher CFG value.

2.3 Sampling Steps Analysis

The number of discrete steps used in the reverse diffusion process impacts both sample quality and generation time. We experimented with 50, 250 (the repository default), and 500 sampling steps, keeping other parameters constant, to observe the effect on the final images.

2.3.1 Observations and Explanation

Generating images with only 50 sampling steps (Figure 3, top panel from results PDF page 2) resulted in noticeably lower quality. Images exhibited visible artifacts, less sharpness, and poorly defined details. This indicates that 50 steps were insufficient for the model to accurately reverse the diffusion process from noise to a clean image within the DDPM framework.

Increasing the steps to 250 (Figure 3, middle panel from results PDF page 3) yielded a significant improvement in quality. Details were much sharper, textures were more realistic, and artifacts were largely eliminated. This suggests that 250 steps provide a reasonable approximation of the continuous reverse process for this model and sampler.

Further increasing the steps to 500 (Figure 3, bottom panel from results PDF page 3) produced images that were visually very similar to the 250-step results. While there might have been subtle improvements in the finest details or coherence, they were marginal and came at the cost of doubled computation time.

This demonstrates the law of diminishing returns regarding sampling steps. While more steps theoretically lead to a more accurate reversal of the diffusion process, practical limitations and the specific sampler used mean that beyond a certain point, the visual quality improvement becomes negligible compared to the increased computational cost. Modern samplers (like DDIM, not used here) are often designed to achieve good quality with even fewer steps than DDPM.

50 Sampling Steps

250 Sampling Steps

3 Task 2: Efficient Attention Implementation

The quadratic complexity of the standard self-attention mechanism in Transformers ($O(N^2)$ where N is the sequence length) can be a bottleneck for high-resolution image generation. We explored two approaches to mitigate this: using the optimized xformers library and implementing Sliding Window Attention (SWA).

3.1 xformers Implementation and Performance

The xformers library provides highly optimized implementations of Transformer components, including a memory-efficient attention mechanism. We replaced the standard `torch.nn.MultiheadAttention` or equivalent implementation within the DiT blocks with the `xformers.ops.memory_efficient_attention` function, ensuring the wrapper maintained the same input/output signature. This was achieved by creating a custom `XFormersAttention` module (as seen in `diffusion_xformers.py`) and replacing the `attn` attribute in each `DiTBlock`.

To quantify the benefit, we benchmarked the inference speed of the original DiT-XL/4 model against the modified version using xformers attention. The benchmark involved generating batches of images with varying sizes (10, 50, 100, 200) and measuring the average time per batch after a warm-up phase, using mixed-precision (‘torch.cuda.amp.autocast‘) on a CUDA-enabled device.

3.1.1 Results and Discussion

The xformers implementation provided a substantial speedup across all tested batch sizes, as shown in Figure 4 and detailed below (extracted from results PDF page 4):

- Batch Size 10: Baseline 0.0510s, xformers 0.0154s ($\approx$ 3.3x speedup)
- Batch Size 50: Baseline 0.2669s, xformers 0.0567s ($\approx$ 4.7x speedup)
- Batch Size 100: Baseline 0.4529s, xformers 0.1155s ($\approx$ 3.9x speedup)
- Batch Size 200: Baseline 0.9913s, xformers 0.2217s ($\approx$ 4.5x speedup)

The average speedup was roughly 4x. This significant improvement stems from xformers’ use of fused operations, optimized CUDA kernels, and reduced memory allocation overhead, which minimizes memory bandwidth bottlenecks and improves GPU utilization, especially noticeable at larger batch sizes or sequence lengths. Crucially, visual inspection confirmed that the generated images were indistinguishable from the baseline, indicating no loss in quality. Therefore, using xformers offers a compelling way to accelerate DiT inference (and potentially training) without compromising results.

Figure 4: Inference time comparison between the baseline DiT-XL/4 and the version using xformers attention across different batch sizes. Lower is better.

3.2 Sliding Window Attention (SWA)

Sliding Window Attention is an approximation technique that restricts the attention calculation for each token to a local window of neighboring tokens, rather than the entire sequence. This reduces the complexity from $O(N^2)$ to $O(N \cdot k)$, where k is the window size (typically $k \ll N$). We implemented SWA with a window size of 3×3 patches (as defined in `SWA_task.ipynb`) and integrated it into the DiT architecture.

To compare its effectiveness, two DiT-B/4 models were trained from scratch for 100 epochs on the landscape dataset ('arnaud58/landscape-pictures') without class conditioning:

1. **Baseline Model**: Using the standard full (global) attention mechanism.
2. **SWA Model**: Using the implemented Sliding Window Attention.

Both models used identical hyperparameters (learning rate 1e-4, batch size 4, Adam optimizer) and the same VAE for latent space operations.

3.3 Comparing Full Attention vs. SWA

After training, we compared the models based on generated sample quality and Fréchet Inception Distance (FID). FID was calculated using 'torch-fidelity' by comparing 32 generated samples from each model against 32 real images from the dataset.

3.3.1 Results and Discussion

The evaluation yielded the following quantitative results (extracted from the `evaluation_results.json` generated by the script, matching results PDF page 6):

Model	FID ↓
Full Attention (DiT-B/4 Trained)	246.90
SWA (DiT-B/4 Trained)	388.88

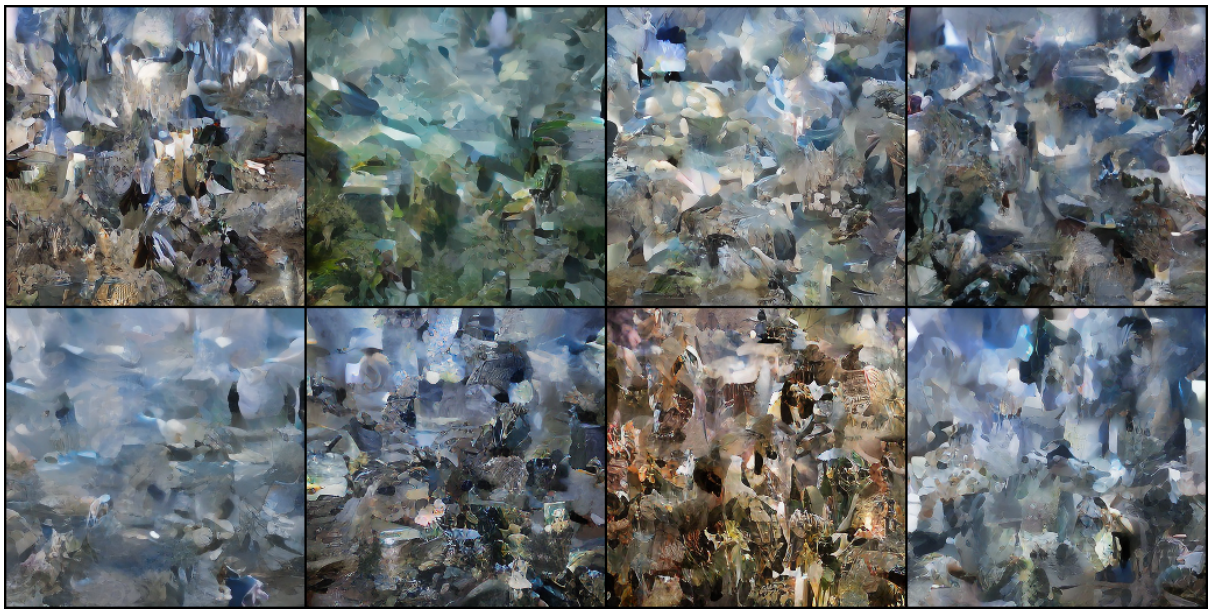
Table 1: FID comparison for DiT-B/4 models trained on landscapes.

Visually, the images generated by the full attention model appeared more coherent and globally consistent than those from the SWA model (Figure 5). The SWA samples, while recognizable as landscapes, sometimes exhibited unnatural repetitions or structural inconsistencies, likely due to the limited receptive field of the attention mechanism.

Quantitatively, the full attention model achieved a significantly better (lower) FID score compared to the SWA model. This suggests that for the landscape dataset, capturing long-range dependencies via global attention is crucial for generating high-fidelity images, and the local approximation provided by SWA (with a small window size) was insufficient.

While SWA offers computational benefits (reduced memory and faster computation per step, although not explicitly benchmarked here), this experiment highlights a potential trade-off: the computational savings might come at the cost of generative quality, especially for tasks requiring global context. The optimal window size and specific SWA implementation details could influence this trade-off.

Full Attention Generated Samples



Sliding Window Attention (SWA) Generated Samples

Figure 5: Comparison of samples generated by the DiT-B/4 models trained with Full Attention (top) and Sliding Window Attention (bottom) on the landscape dataset.

4 Task 3: CLIP Mean Maximum Discrepancy Evaluation

Evaluating generative models is challenging. While FID is a standard metric, it has limitations. We explored an alternative, CMMD, leveraging the semantic understanding of vision-language models like CLIP, SigLIP, and ALIGN.

4.1 Understanding FID vs. CMMD

Fréchet Inception Distance (FID) relies on features extracted from an Inception-v3 network pre-trained on ImageNet classification. It calculates the Fréchet distance (a measure of distance between probability distributions) between the distributions of these features for real and generated images. *Assumptions/Limitations:*

- Assumes the extracted features follow a multivariate Gaussian distribution.
- Features are biased towards ImageNet classes and may not capture all aspects of image quality or semantic content relevant to other domains.
- Sensitive to imperceptible perturbations and might not always align perfectly with human perception of quality.

CLIP Mean Maximum Discrepancy (CMMD) uses image embeddings from CLIP (or similar models like SigLIP, ALIGN) trained on large-scale image-text pairs. It calculates the Maximum Mean Discrepancy (MMD), a non-parametric measure, between the distributions of these embeddings for real and generated images, typically using a kernel like the Radial Basis Function (RBF). *Advantages over FID:*

- Leverages richer semantic features learned from image-text data.
- Does not assume a Gaussian distribution for the features.
- MMD can capture differences in higher-order moments of the distributions.
- Potentially aligns better with semantic similarity and content correctness.

4.2 CMMD Implementation

We implemented a function (`calculate_cmmd` in `SWA_task.ipynb`) to compute the MMD score between two sets of images using embeddings from specified models (CLIP, SigLIP, ALIGN). The process involves:

1. Loading the chosen pre-trained vision-language model and its associated processor (e.g., `openai/clip-vit-base-patch32`).
2. Preprocessing both real and generated images.
3. Extracting image embeddings in batches to manage memory.
4. Normalizing the embeddings.
5. Calculating the MMD score between the sets of real and generated embeddings using a helper function (`mmd_score`) which implements the kernel MMD calculation. We used the unbiased MMD estimate with an RBF kernel, where the kernel bandwidth (σ) was estimated using the median heuristic on a subset of the data if not provided.

4.3 Comparing FID and CMMD Results

We applied both FID and CMMD (using CLIP embeddings) to evaluate the full attention and SWA DiT-B/4 models trained on the landscape dataset. The results (using 32 samples for evaluation, as per the `evaluate_models` function) are summarized below:

Model	FID ↓	CLIP-MMD ↓
Full Attention (DiT-B/4 Trained)	246.90	0.2812
SWA (DiT-B/4 Trained)	388.88	0.4435

Table 2: Comparison of FID and CLIP-MMD for trained landscape models.

Both FID and CLIP-MMD consistently ranked the full attention model as superior to the SWA model for this specific training run and dataset. This indicates a correlation between the two metrics in this instance – the model deemed better by FID was also deemed better by CLIP-MMD. The CMMD score, reflecting the discrepancy in the CLIP embedding space, confirms the visual and FID-based assessment that the full attention model produced samples closer to the distribution of real landscape images. While FID focuses on lower-level feature statistics (assuming Gaussianity), CMMD provides a non-parametric comparison in a semantically richer space, potentially offering a more robust evaluation, especially when the Gaussian assumption might not hold or when semantic fidelity is paramount.

4.4 SigLIP and ALIGN Comparisons

To further investigate the influence of the embedding space, we extended the CMMD calculation using SigLIP (‘google/siglip-base-patch16-224‘) and ALIGN (‘kakaobrain/align-base‘) embeddings.

4.4.1 Results and Discussion

The MMD scores using different embedding models were:

Model	CLIP-MMD ↓	SigLIP-MMD ↓	ALIGN-MMD ↓
Full Attention	0.2812	0.2527	0.3326
SWA	0.4435	0.3986	0.4735

Table 3: MMD scores using CLIP, SigLIP, and ALIGN embeddings.

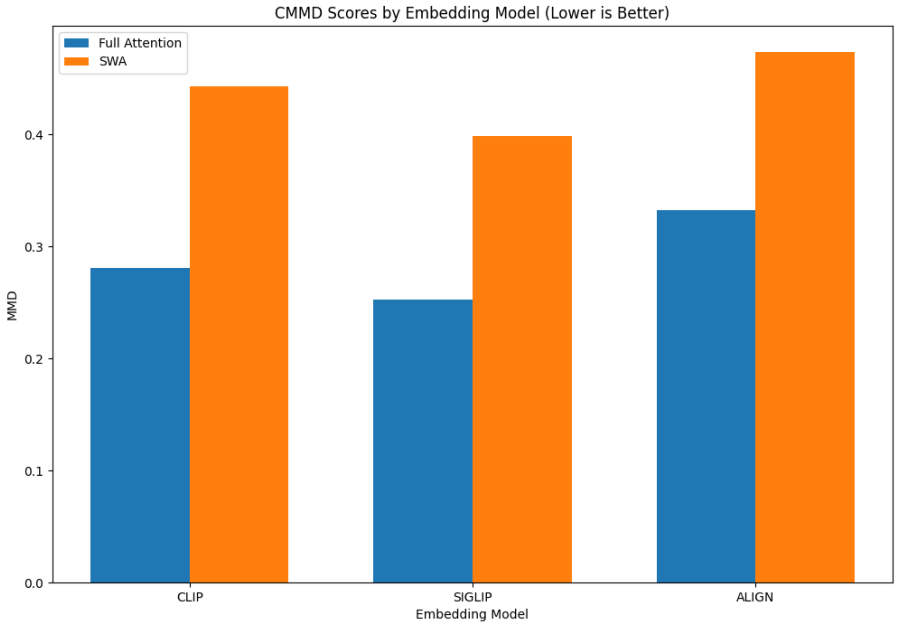


Figure 6: CMMD Scores using CLIP, SigLIP, and ALIGN embeddings for Full Attention vs. SWA models. Lower is better.

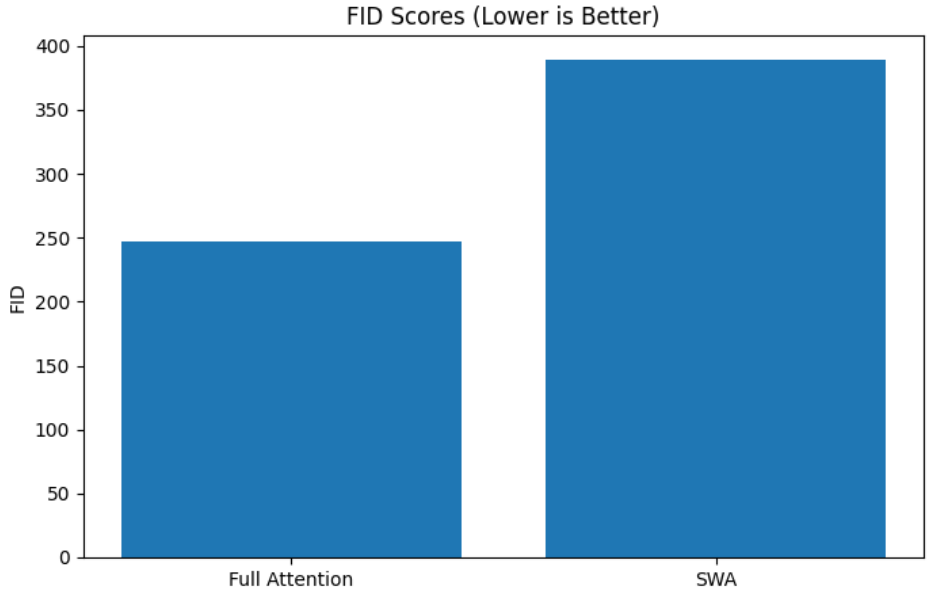


Figure 7: FID Scores for Full Attention vs. SWA models. Lower is better.

Key observations:

- **Consistent Ranking**: All three embedding models (CLIP, SigLIP, ALIGN) consistently ranked the full attention model as better (lower MMD) than the SWA model, reinforcing the conclusion drawn from FID and visual inspection.

- **Score Magnitudes**: SigLIP yielded the lowest MMD scores, potentially indicating a more discriminative or well-calibrated embedding space for this task due to its training objective (sigmoid loss over pairwise similarity). ALIGN produced the highest scores, which might reflect its training on a vast but noisy dataset, leading to a broader but perhaps less fine-grained embedding space.
- **Metric Robustness**: The consistent relative ranking across different powerful vision-language models suggests that the observed quality difference between the full attention and SWA models is robust and not merely an artifact of a single evaluation metric or embedding space.

These results demonstrate the utility of MMD-based evaluations using various foundation models. While absolute scores differ, the relative comparisons provide valuable insights into generative model performance, complementing traditional metrics like FID by offering non-parametric evaluation within semantically meaningful feature spaces.

5 Conclusion

This study provided a hands-on exploration of the Diffusion Transformer (DiT) architecture and associated techniques. Key findings include:

- **Parameter Sensitivity:** Hyperparameters like CFG scale and the number of sampling steps significantly impact the trade-off between sample fidelity, diversity, and computational cost, aligning with established diffusion model principles.
- **Attention Efficiency:** Replacing standard attention with xformers yields substantial inference speedups (approx. 4x in our tests) without sacrificing output quality, highlighting the practical benefits of optimized implementations.
- **Attention Variants:** While Sliding Window Attention (SWA) offers computational advantages, its local nature resulted in significantly worse generative quality (higher FID and MMD scores) compared to full attention when trained on a landscape dataset requiring global context in our specific experiment. This underscores the importance of attention scope for certain generation tasks.
- **Evaluation Metrics:** CMMD, calculated using embeddings from CLIP, SigLIP, and ALIGN, provides a valuable alternative or complement to FID. It avoids the Gaussian assumption of FID and leverages semantically rich feature spaces. In our experiments, CMMD correlated well with FID and visual quality assessment, consistently ranking the models similarly across different embedding spaces (CLIP, SigLIP, ALIGN).

Overall, DiT proves to be a powerful and scalable architecture. Optimizing its components (like attention) and carefully choosing evaluation metrics appropriate for the task are crucial for leveraging its full potential in generative modeling.