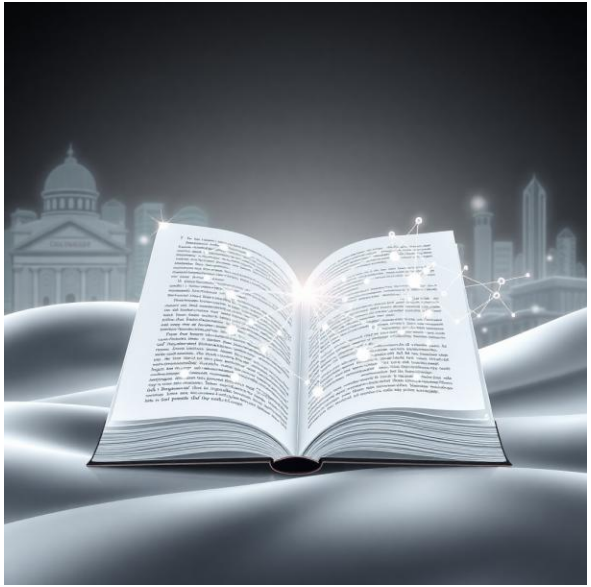
EEE 598: Generative AI: Theory and Practice

Team members

Aakash Kumar Tomar (1229632003)
Abhijeet Ghildiyal (1229612347)
Prateek Parashar(1229631743)

Project Title : Faster Diffusion – Rethinking the Role of the Encoder for Diffusion Model Inference



Paper Reference:

(https://arxiv.org/pdf/2312.09608)

Project Goals

DM	Sampling Method	T	FID ↓	Clip-score ↑	GFLOPs/image ↓	s/image ↓ Unet of DM	s/image ↓ DM
Stable Diffusion	DDIM [44]	50	21.75	0.773	37050	2.23	2.42
	DDIM [44] w/ Ours	50	21.08	0.783	**27350**$_{27\%\downarrow}$	**1.21**$_{45\%\downarrow}$	**1.42**$_{41\%\downarrow}$
	DPM-Solver [30]	20	21.36	0.780	14821	0.90	1.14
	DPM-Solver [30] w/ Ours	20	21.25	0.779	**11743**$_{21\%\downarrow}$	**0.46**$_{48\%\downarrow}$	**0.64**$_{43\%\downarrow}$
	DPM-Solver++ [31]	20	20.51	0.782	14821	0.90	1.13
	DPM-Solver++ [31] w/ Ours	20	20.76	0.781	**11743**$_{21\%\downarrow}$	**0.46**$_{48\%\downarrow}$	**0.64**$_{43\%\downarrow}$

Table 1. Quantitative evaluation in both SD model

Standard Baselines

- Stable Diffusion v1.5
- Samplers: DDIM, DPM-Solver, DPM-Solver++

Dataset

- MSCOCO-2017

Metrics

- FID
- Clip score
- Sampling time (s/image for Unet)

This covers the baseline evaluation of the models and highlights the faster diffusion method proposed by the paper

Project Goals

Main Goal

Our project aims to analyze encode and decoder features in stable diffusion models, verifying minimal change in encoder outputs and significant change in decoder outputs during generation

Feature Measurement

Encoder and decoder features are quantitatively measured using Mean Squared Error and Frobenius norm for accuracy and consistency.

Objective

Accelerating stable diffusion sampling time without retraining using encoder propagation,
While maintaining high generation image quality (evaluated using FID & Clip score)

MS-COCO 2017 Dataset

About the Dataset

- ✓ MS-COCO is a standard benchmark for evaluating text-to-image generation models, it contains diverse, natural image captions covering everyday scenes, objects & interactions.
- ✓ The training/validation for the dataset is 118K/5K, each image has 5 different captions/prompts (all of them are human-written descriptions of the image, collected via Amazon Mechanical Turk)
- ✓ For our project we are using the validation set (5000 images) where each image is mapped to one caption each.

E.g.1 A woman stands in the dining area at the table

E.g.2 A big burly grizzly bear is show with grass the background

Sampling Pipeline

Dataset Preparation

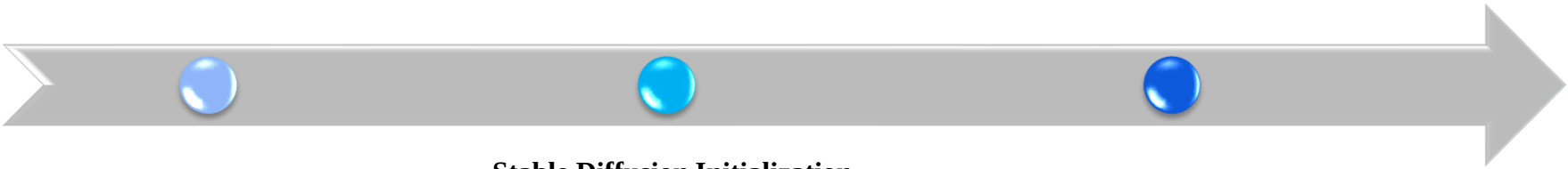
Validation set contains 5,000 images- each paired with one prompt

Stable Diffusion Initialization

Stable Diffusion v1.5 is initialized alongside samplers like DDIM, DDPM, DPM Solver, and DPM Solver++

Sampling & Metrics

Baseline image generation and metric calculations are performed, followed by implementation of faster diffusion for comparative analysis



FASTER DIFFUSION IMPLEMENTATION

Key Steps of Implementation

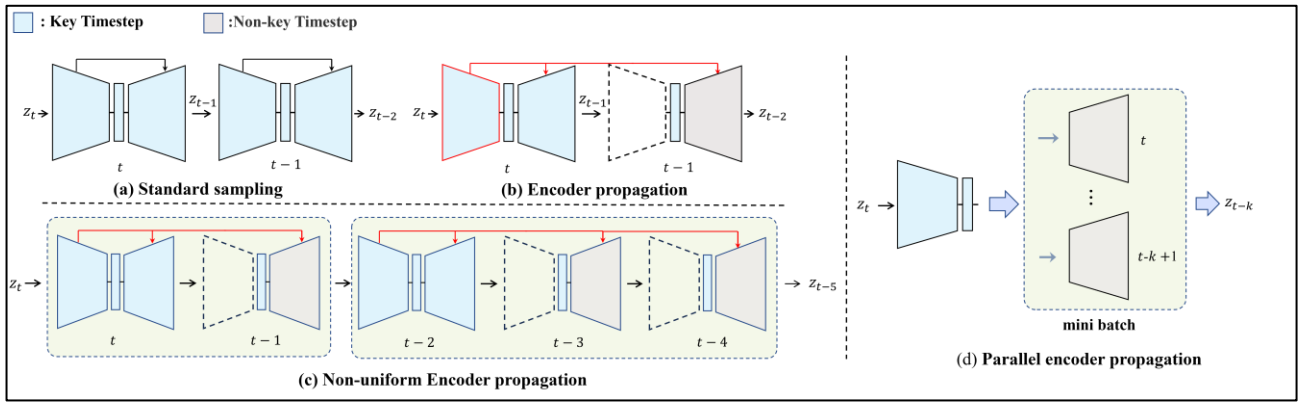
Encoder features are cached at selected timesteps, reused at non-key steps, and parallel encoding is enabled for efficiency.

Sampling Loop Alterations

The sampling loop is modified to accommodate encoder propagation and optimize decoding.

Evaluation

Metrics tracked include FID, CLIP score, and overall sampling time.



MODEL ARCHITECTURE OVERVIEW

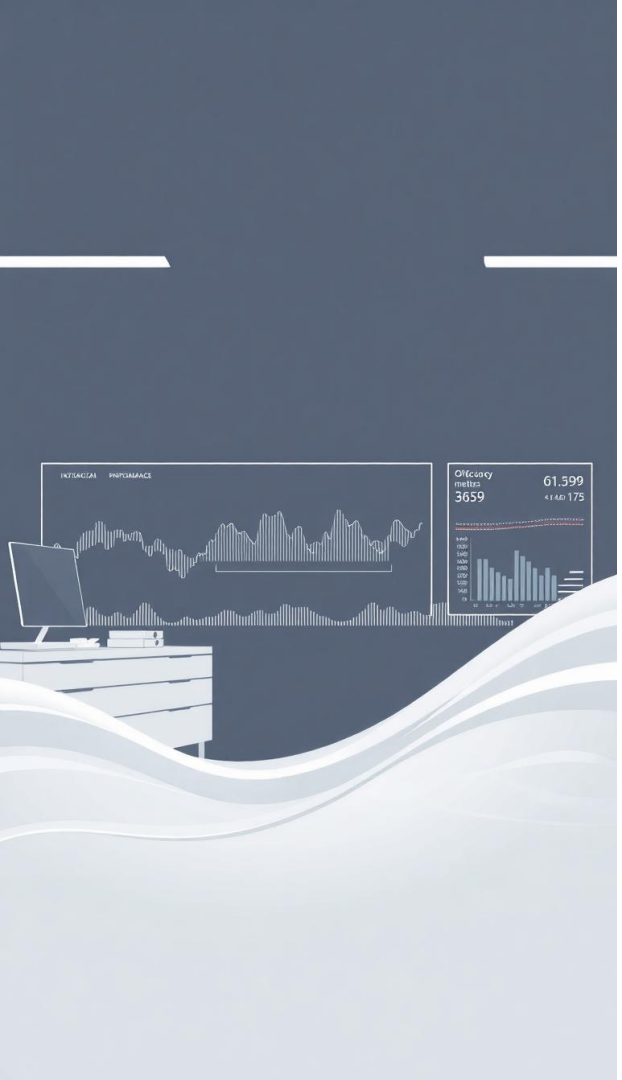
- Stable Diffusion v1.5 – Employs a UNet backbone with encoder, bottleneck, decoder & skip connections, Text is processed via a CLIP(ViT-L/14) encoder.
- Sampling Algorithms- Multiple schedulers such as DDIM, DDPM, DPM Solver & DPM Solver++.

LIBRARIES AND TOOLS

- Hugging Face Diffusers: for loading the Stable Diffusion model and running sampling
- PyTorch: for underlying tensor operations scikit-learn: (for optional PCA if re-analysing features, similar to paper)
- NumPy, Matplotlib: for numerical analysis and plotting
- torch metrics: (optional) for easier FID computation

OTHER DETAILS & METRICS

- Sampling only- The focus is strictly on inference & sampling optimization, not on model training
- Inference Optimization Methods- These include encoder propagation, parallel decoding, and capturing evaluation metrics such as FID, Clip score & Sampling time



Main Results

Hyperparameters and Ranges

Parameter	Range	Description	Why Tuned?
Key Time-Steps Selection (t_key)	Manual list [0, 1, 2, 3, 5, 10, 15, 25, 35]	Chosen time-steps where encoder features are recomputed	To balance speed-up and minimize quality loss
Prior Noise Injection Strength (α)	0.003	Amount of initial noise (zT) added during sampling	To recover texture details lost in encoder propagation
Scheduler Type	{DDIM, DPM-Solver, DPM-Solver++}	Sampling scheduler used for denoising	Different schedulers have different tradeoffs between speed and quality
Sampling Steps (T)	20–50	Number of sampling steps during generation	Higher steps = better quality but slower inference

RESULTS

1. What Worked:

•Encoder propagation reduced sampling time significantly (~24–41% faster than standard DDIM sampling).
•Prior noise injection helped recover texture and fine details without large computational cost.
•FID and Clip score metrics stayed very close to baseline Stable Diffusion results.
•Parallel decoding worked well, allowing multi-step decoding simultaneously.
•Sampling acceleration was achieved without retraining the model (huge benefit over distillation-based methods).

2. What Didn’t Work (Challenges):

•Without prior noise injection, some generated images were slightly smoother and lost fine textures (especially at early time steps).
•Aggressively skipping too many encoder time-steps (too few key steps) led to semantic drift: images not matching prompts as closely.
•Some minor memory increase when enabling caching (not huge, but noticeable for GPU).

Method	Pros	Cons
Faster Diffusion	No retraining needed, parallelism, good quality	Minor texture loss without noise injection
Distillation methods	One-step or few-step generation	Requires huge retraining time, new models

Prompt: A big burly grizzly bear is show with grass in the background

Stable Diffusion

Faster Diffusion

Result Visualization

Method	Steps	FID	Clip score	Sampling time (s)
DDIM	50	25.2524	0.7798	1.518927889
DDIM w/ Ours	50	25.0416	0.7808	1.10814
DPM-Solver	20	26.9162	0.7781	2.156606401
DPM-Solver w/ Ours	20	27.2168	0.779	0.685638421
DPM-Solver-PP	20	27.0645	0.7786	0.633730816
DPM-Solver-PP w/ Ours	20	26.8469	0.7806	0.49428855

Pros and Cons of Approach

➢Why should this approach be adopted?

- Significantly accelerates diffusion sampling (up to 41% faster) without any model retraining, making it accessible even with limited compute resources.
- Maintains high image quality (FID and Clipscore close to baseline) while enabling partial parallelization during inference.

➢Limitations/Problems with this approach

•Slight loss of fine texture details without applying prior noise injection during encoder propagation.
•Manual tuning of key time-steps is required to balance speedup vs. quality — not fully automatic yet.

Team Members

Aakash Kumar Tomar ASU ID – 1229632003	• Feature analysis, code implementation, sampling pipelines & validation
Abhijeet Ghildiyal ASU ID – 1229612347	• Data setup, sampling pipelines & documentation
Prateek Parashar ASU ID – 1229631743	• Result analysis, Project management, codebase maintenance & final reporting

References

1. Jennewein, Douglas M. et al. "The Sol Supercomputer at Arizona State University." In Practice and Experience in Advanced Research Computing (pp. 296–301). Association for Computing Machinery, 2023
2. Tsung-Yi Lin, Michael Maire, Serge Belongie, Lubomir Bourdev, Ross Girshick, James Hays, Pietro Perona, Deva Ramanan, C. Lawrence Zitnick, Piotr Dollár “Microsoft COCO: Common Objects in Context” , 2014
3. Robin Rombach, Andreas Blattmann, Dominik Lorenz, Patrick Esser, Björn Ommer, “High-Resolution Image Synthesis with Latent Diffusion Models(Stable Diffusion)” , 2021
4. Jack Hessel, Ari Holtzman, Maxwell Forbes, Ronan Le Bras, Yejin Choi, “CLIP Score: A Reference-free Evaluation Metric for Image Captioning” ,2021
5. Yu Yu, Weibin Zhang, Yun Deng, “Frechet Inception Distance (FID) for Evaluating GANs”,2021
6. Cheng Lu, Yuhao Zhou, Fan Bao, Jianfei Chen, Chongxuan Li, Jun Zhu “DPM-Solver++: Fast Solver for Guided Sampling of Diffusion Probabilistic Models”, 2022
7. Jiaming Song, Chenlin Meng, Stefano Ermon, “Denoising Diffusion Implicit Models”, 2020
8. Jonathan Ho, Ajay Jain, Pieter Abbeel “Denoising Diffusion Probabilistic Models”. 2020
9. Cheng Lu, Yuhao Zhou, Fan Bao, Jianfei Chen, Chongxuan Li, Jun Zhu “DPM-Solver: A Fast ODE Solver for Diffusion Probabilistic Model Sampling in Around 10 Steps”, 2022