GMR
The Original

A REPRODUCIBLE HARDWARE-AWARE BENCHMARK OF STABLE DIFFUSION V1.5 FOR TEXT-TO-IMAGE GENERATION ON CPU-CLASS SYSTEMS

Pathanjali Chowdaiah[1*], Vijaykumar S[2], Pavithra Goravi Sukumar[3*], A. Latha[4], Manjula B S[5], Anusha U A[6]

[1]Department of Information Science and Engineering, Global Academy of Technology, Bengaluru, India, pathanjalic@gat.ac.in
[2]Department of Information Science and Engineering, Global Academy of Technology, Bengaluru, India, vijaykumar.s@gat.ac.in
[3]Department of Computer Science and Engineering (AI&ML), RNS Institute of Technology, Bengaluru, India , pavi.pgs@gmail.com
[4]Department of Computer Science and Engineering, Sri Krishna Institute of Technology, Bengaluru, India, Alatha303@gmail.com
[5]Department of Information Science and Engineering, Global Academy of Technology, Bengaluru, India, manjulabs@gat.ac.in
[6]Department of Information Science and Engineering, Global Academy of Technology, Bengaluru, India, anushaua@gat.ac.in

ABSTRACT
Stable Diffusion generates high-resolution images from text via an iterative denoising process, but incurs large inference time costs on resource-poor computational platforms. This paper shows a controlled hardware-aware benchmark of Stable Diffusion v1. Let us take a deeper look into the stability of text-to-image generation outputs on two CPU-class systems: AMD Ryzen 5 4600H (6C) and AMD Ryzen 7 5800H (8C) Num. It uses classifier-free guidance with a scale of 8, the DDPM sampler, generates 512×512 pairs and measures with seed control ×2. CLIP scores of at least 28% in an eight-prompt semantic-alignment test, with a mean score (between +3 to -4) of 31.25%. For latency analysis, three prompts were selected for evaluation with three seeds (42, 200 and 999), which returned nine pairwise observations per hardware comparison. Mean rendering time decreased from 5.35 min on the Ryzen 4600H system to 3.87 min on the Ryzen 5800H system, an absolute saving of 1.48 min per generation and a 27.6% reduction. A paired analysis at the prompt-seed observation level also shows a large effect (Cohen's dz = 2.81; paired t-test $p < 0.001$; Wilcoxon signed-rank $p = 0.0039$). The results indicate that CPU configuration materially affects the practical deployment cost of text-to-image diffusion inference under otherwise consistent settings. The study contributes a reproducible small-scale benchmark protocol and identifies the measurement factors that should be reported when evaluating diffusion-model latency on heterogeneous systems.

KEYWORDS: Stable Diffusion v1.5, text-to-image generation, hardware benchmarking, inference latency, DDPM CPU performance.

INTRODUCTION

Text-to-image diffusion models have rapidly become an important class of generative AI systems. Denoising diffusion probabilistic models formulate generation as a sequence of denoising operations[1], while latent diffusion moves the expensive iterative process into a learned compressed representation and uses cross-attention for conditioning [3]. Stable Diffusion applies this family of ideas to practical text-to-image synthesis and has become a widely used reference system for accessible image generation. The original implementation is designed for 512×512 synthesis and uses a frozen CLIP text encoder to condition an 860M-parameter U-Net [3,6].

For deployment, image quality alone is insufficient. Each denoising step requires computation, and the total wall-clock time depends on the model, sampling configuration, guidance strategy, hardware, and execution environment. Prior diffusion research has repeatedly identified sampling efficiency as a practical concern; work on DDIM and improved DDPMs explicitly targets the trade-off between sample quality and the number of model evaluations [2,5]. This makes hardware-aware benchmarking important, especially for systems that cannot rely on high-end GPUs.

The present study is motivated by a practical observation from an implementation-based evaluation: the same text prompts generated images with visibly different rendering times on two CPU configurations while model and generation settings were held constant. Rather than treating hardware as a secondary implementation detail, this paper treats inference time as a first-class evaluation variable and reports the underlying prompt-seed measurements.

A key distinction in this work is between model-quality evaluation and deployment benchmarking. The former asks whether generated outputs are semantically aligned with prompts; the latter asks how long the same generation procedure takes on a particular computing platform. The present study focuses on this deployment-facing second question while retaining a small CLIP-based semantic baseline.

A. Diffusion-based text-to-image generation:

Instead, DDPMs learn a denoising forward process to gradually convert noisy samples into more similar samples of the target distribution [1]. The sequential nature makes the number of denoising evaluations directly related to inference cost. Latent diffusion reduces computation by performing diffusion in the latent space of an autoencoder rather than directly in pixel space, while cross-attention enables conditioning on text [3]. Stable Diffusion v1.x is a practical realization of this design for 512×512 generation [6].

B. Sampling and Guidance:
DDIM provides a non-Markovian sampling formulation that can reduce the number of sequential operations required at inference time [2]. Improved DDPM work similarly demonstrates that better parameterization and learned variances can reduce the number of forward evaluations [5]. Classifier-free guidance introduces a controllable trade-off between conditioning strength and sample diversity by combining conditional and unconditional score estimates [7]. These methods make sampling configuration a central factor in deployment latency.

C. Evaluation of Text–Image Alignment:
CLIP learns a shared representation for images and text through large-scale contrastive pre-training and supports semantic alignment analysis between a prompt and a generated image [4]. In the present study, CLIP is used as a descriptive quality indicator for the eight-prompt test reported in the source implementation. No FID-based claim is retained because the underlying project report does not document a reference-image protocol sufficient for reproducible FID computation.

D. Research Gap:
Previous diffusion work addresses the sequential cost of iterative denoising, employing both theoretical and algorithmic solutions for faster sampling [1,2,5]. Yet, the deployment choice ultimately relies on the physical layer where this same inference process has been executed. This research purpose is thereby narrower and more applied: reproducible controlled comparisons of the rendering time taken by Stable Diffusion on a small number of easy-to-deploy CPU-class configurations reported alongside the encoding parameter choices to correctly interpret the timing outcome.
The study answers three questions:
(RQ1) What is the semantic-alignment performance of the Stable Diffusion v1. 5 text-to-image config through a small controlled prompt dataset?
(RQ2) For a given prompt, seed and generation settings, to what extent does CPU configuration affect rendering time?
(RQ3) What reproducibility and deployment insights can be gained by seed-controlled timing measurements on two CPU-class systems?

MATERIALS AND METHODS
The experimental workflow of the hardware aware revised benchmark is showcased in the Fig 1 and detail explanation is as follows below:

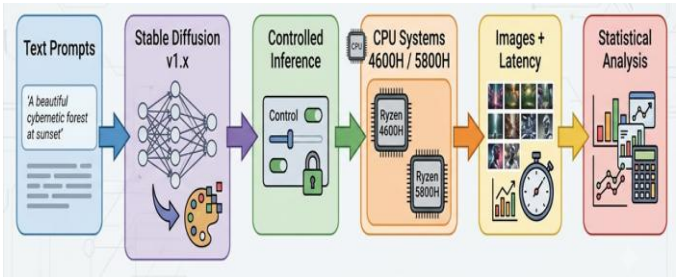


Fig. 1. Hardware-Aware Experimental Workflow used in the Benchmark.

A. Model and Implementation
The implementation uses Stable Diffusion v1.5 for text-conditioned image generation. The project implementation enables classifier-free guidance (do_cfg=True) with a configuration scale (CFG scale) of 8. The DDPM sampler is used in the demonstrated implementation, with a strength value of 0.9 for the image-to-image functionality. The text-to-image hardware benchmark is the focus of this paper, image-to-image segmentation results from the project are outside the scope of the present evaluation.

B. Controlled Generation Configuration
The implementation documentation also describes device selection for CPU, CUDA, or MPS. The reported timing table, however, is explicitly organized around the two Ryzen CPU configurations. Consequently, the quantitative hardware conclusions in this paper are limited to the tested CPU systems and should not be generalized to GPU or accelerator platforms.

C. Prompt Set and Semantic Evaluation
The project testing phase evaluates eight text prompts covering natural scenes, objects, compositional descriptions, and imaginative scenarios. The reported prompts include a mountain landscape, an underwater coral reef, a reading nook with a cat and books, a snowy village, a frog riding a bicycle, a banana-shaped spacecraft, an astronaut throwing a baseball toward an asteroid, and a flying bus. The project accounts a CLIP score and a pass/fail decision for each prompt it depicted in the Table 2.

D. Hardware Timing Protocol
In the sense of speed, as an example to exemplify latency analysis, the same prompt (text to image) defined from a single text prompt was sampled three times using multiple seeds over both systems. These prompts were: a cat on a shelf sleeping, some students in the library studying and a busy yellow city taxi. In total, this process produced nine pairs of timing

observations (three types × three seeds). In the project report, CPU based render time is reported in minutes for both configurations.
The analysis considers each pair of prompt and seed as a matched observation. Descriptive modelling was reported as mean, standard deviation, median and the 95th percentile. The manuscript performs exploratory statistics because the three prompts are repeated across 3 seeds and does not present the nine observations as completely independent samples.

E. Statistical Analysis

Each of the nine prompt-seed observations is paired with two complementary analyses: a paired t-test (t; padj = P) and a Wilcoxon signed-rank test (W; padf = P). After testing distribution of paired differences, the parametric sensitivity analysis computes a paired t-test and its non-parametric counterpart uses Wilcoxon test. You report effect size as Cohen's dz. The analysis is intentionally restricted to the actual benchmark setups.

F. System Architecture

The framework that demonstrates in Fig 2 disaggregates the benchmark into eight reproducible layers: prompt, seed control, conditioned Stable Diffusion inference, hardware execution, output measurement and finally evaluation and statistical analysis. In both CPU configurations, the same prompt and seed are executed so that a paired latency comparison is made at the prompt-seed level. It also makes clear which parameters are fixed (CFG scale, resolution, sampler and step count) and what exactly you or a configuration are being compared against (CPU).
Ethics statement: This study involved computational benchmarking of a generative-AI system and did not involve human participants, animals, clinical data, or personally identifiable information. Institutional human-subject ethics approval was therefore not applicable.

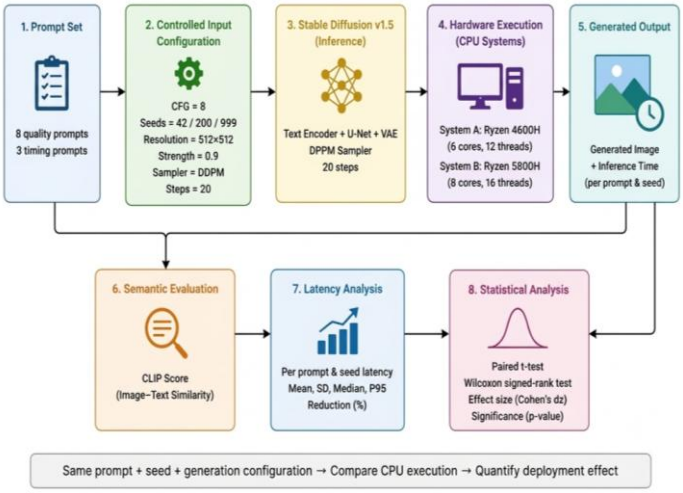


Fig. 2. The Controlled Hardware-Aware Stable Diffusion Benchmark architecture.

RESULTS

A. Controlled Generation Parameters

Table 1. controlled generation parameters used in the source implementation.

Parameter	Value used in the source implementation	Role
Model	Stable Diffusion v1.5	Text-to-image generation
CFG scale	8	Controls classifier-free prompt conditioning
CFG enabled	Yes	Conditional + unconditional guidance
Sampler	DDPM	Primary sampling method
Inference steps	20 in the demonstrated implementation	Controls denoising iterations
Strength	0.9 (image-to-image path)	Controls transformation strength
Resolution	512×512	Generation resolution
Seeds	42, 200, 999 in timing table	Repeatability check
Hardware	Ryzen 5 4600H and Ryzen 7 5800H	CPU comparison

The implementation documentation tabulated in Table 1 also describes device selection for CPU, CUDA, or MPS. The reported timing table, however, is explicitly organized around the two Ryzen CPU configurations. Consequently, the quantitative hardware conclusions in this paper are limited to the tested CPU systems and should not be generalized to GPU or accelerator platforms.

B. Semantic-Alignment Test

The eight reported prompts used in the experiment are tabulated in Table 2, the CLIP scores fluctuated from 28% to 36%, with a mean of 31.25% (SD = 2.55). As showcased in Table 2 the first four prompts were labelled Pass and next four Fail in the source project table. The strongest score was attained for the reading-nook prompt (36%), while the lowest score was attained for the flying-bus prompt (28%). These results determine measurable variation in text–image alignment within the implemented configuration.

Table 2. CLIP scores and project labels for the eight reported prompts.

Prompt	CLIP score (%)	Project label
A serene mountain landscape at sunrise	33	Pass
A vibrant underwater coral reef teeming with fish	32	Pass
A cozy reading nook with a cat and a stack of books	36	Pass
A quaint village covered in snow during winter	32	Pass
A frog riding a bicycle	30	Fail
A banana-shaped spacecraft in space heading towards the moon with Earth visible	30	Fail
A man in a spacesuit throwing a baseball towards an asteroid	29	Fail
A flying bus	28	Fail

C. Hardware Rendering-Time Results

Across all nine matched observations tabulated in the Table 3, the Ryzen 4600H system has a mean rendering time of 5.35 min (SD = 0.32), while the Ryzen 5800H system has a mean of 3.87 min (SD = 0.40). The mean absolute saving is 1.48 min per generated image, corresponding to a 27.6% reduction in average rendering time. The mean speedup ratio is approximately 1.38X. At the prompt level, the reduction is 23.8% for "Cat sleeping on a shelf," 35.8% for "Students studying in library," and 22.6% for "Yellow taxi in a busy city." the same is plotted in the Fig 3.

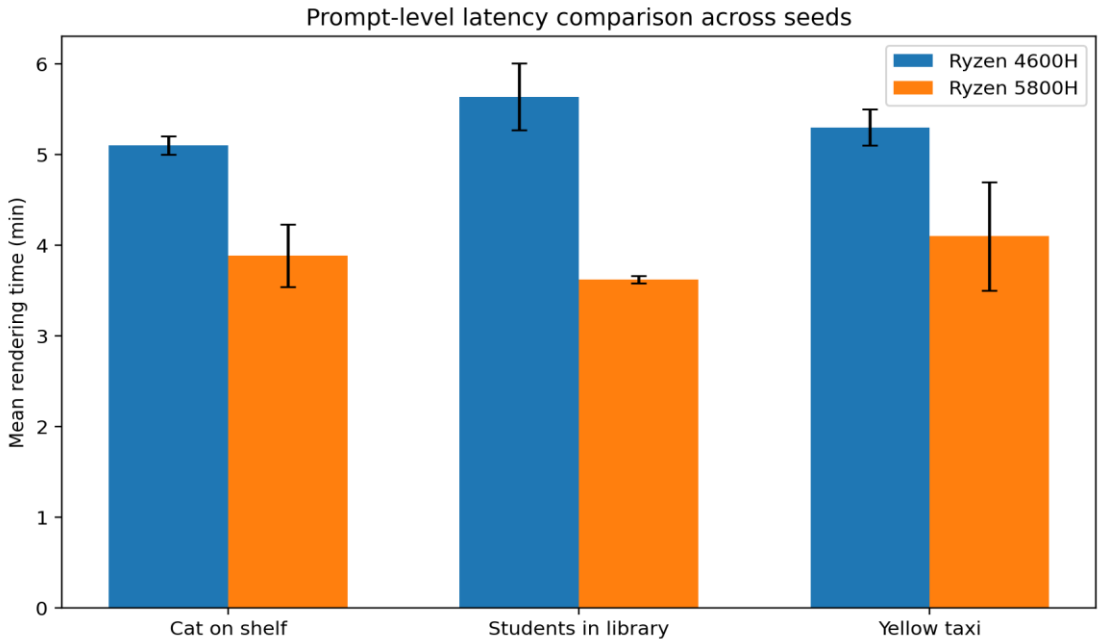


Fig. 3. Prompt-Level Mean Rendering Time for the two CPU configurations error bars show one standard deviation.

The descriptive comparison is visually consistent with the raw timing table plotted in Fig 3, the mean decreases from 5.35 min to 3.87 min which as depicted in Fig 4. The nonoverlapping central tendency is supported by the paired tests reported below in Table 3 and Table 4, but the plot is intentionally presented as a descriptive benchmark rather than as a population-level CPU ranking.

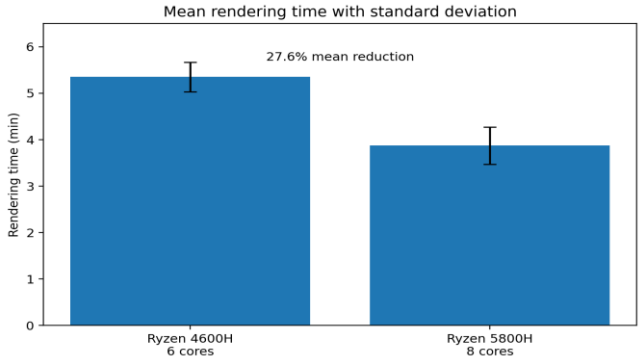


Fig. 4. Overall mean rendering time by hardware configuration error bars show one standard deviation.

Table 3. Raw text-to-image rendering-time observations across the two CPU configurations.

Prompt	Seed	Ryzen 4600H (min)	Ryzen 5800H (min)	Reduction (%)
Cat sleeping on a shelf	42	5.00	4.00	20.0
Cat sleeping on a shelf	200	5.20	3.50	32.7
Cat sleeping on a shelf	999	5.11	4.16	18.6
Students studying in library	42	5.22	3.60	31.0
Students studying in library	200	5.80	3.60	37.9
Students studying in library	999	5.90	3.67	37.8
Yellow taxi in a busy city	42	5.50	4.70	14.5
Yellow taxi in a busy city	200	5.10	3.50	31.4
Yellow taxi in a busy city	999	5.30	4.10	22.6

Table 4. Descriptive Statistics for the paired hardware rendering-time benchmark.

Statistic	Ryzen 4600H	Ryzen 5800H	Paired difference
Mean (min)	5.348	3.870	1.478
SD (min)	0.319	0.403	0.526
Median (min)	5.22	3.67	1.60
95th percentile (min)	5.86	4.48	2.22
95% CI of mean (min)	[5.10, 5.59]	[3.56, 4.18]	[1.07,1.88]

D. Statistical Comparison of CPU Configurations

The nine paired differences are all positive: the AMD Ryzen 5800H is faster for every prompt-seed combination recorded in the source table (Table 5). A paired t-test yields $t(8) = 8.42$, $p = 3.00e\text{-}05$, while the two-sided Wilcoxon signed-rank test yields $p = 0.0039$. The standardized paired effect is Cohen's $dz = 2.81$, the 95% confidence interval for the mean paired saving is [1.07, 1.88] min. These results provide strong evidence that the two tested CPU configurations differ in rendering time within this benchmark. Because the observations repeat three seeds for each of only three prompts, this inference should be interpreted as evidence for the tested benchmark rather than a population-wide estimate of CPU performance the same is tabulated in Table 5. The Fig 5 shows the Paired latency observations for all nine prompt-seed combinations, each line connects the same run across the two CPU systems.

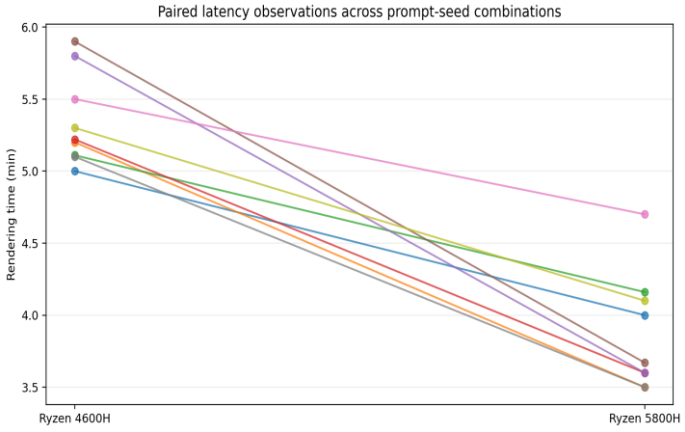


Fig. 5. Paired latency observations for all nine prompt-seed combinations.

Table 5. Inferential and effect-size evidence for the paired hardware comparison.

Test / quantity	Result	Interpretation
Paired t-test	$t(8) = 8.42; p = 3.00\times10^{-5}$	Strong evidence of a mean latency difference within the benchmark
Wilcoxon signed-rank	W = 45; p = 0.0039(two-sided)	Non-parametric confirmation using paired observations
Cohen's dz	2.81	Large standardized paired effect
Mean paired saving	1.48 min; 95% CI [1.07, 1.88]	Estimated saving per generation for the tested configurations
Relative reduction	27.6%	Average rendering time reduction on Ryzen 5800H vs 4600H

E. Reproducibility Observations

Seed variation is visible within each prompt, but the direction of the hardware effect is stable. For the cat prompt, rendering times on the 4600H range from 5.00 to 5.20 min, while the 5800H ranges from 3.50 to 4.16 min. For the library prompt, the corresponding ranges are 5.22–5.90 min and 3.60–3.67 min. For the taxi prompt, the ranges are 5.10–5.50 min and 3.50–4.70 min. Thus, the hardware ordering remains consistent across the repeated seeds even though absolute latency varies. This is fundamental for benchmark design, a single timing observation can vague run-to-run variability, while seed-repeated measurements expose the stability of the relative ranking.

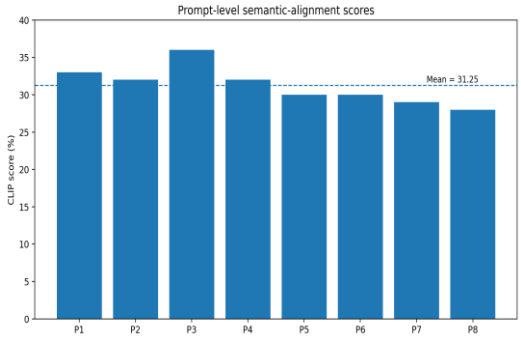


Fig. 6. CLIP scores for the eight reported semantic-alignment prompts with the mean score of 31.25%.

DISCUSSION

The experimental evidence for the claim mapping for the research question is tabulated in the Table 6. The detail justification is presented below for each question discussed in research gap section.

A. Justification for rq1: Semantic Alignment

The eight-prompt evaluation establishes that the implemented Stable Diffusion v1.5 configuration can produce semantically allied outputs across simple natural scenes and more imaginative prompts, with CLIP scores between 28% and 36%. The mean score of 31.25% provides a descriptive baseline for this implementation as shown in Fig 6. However, the source experiment does not include a matched v1.4 prompt-by-prompt evaluation under the same hardware and seed conditions. Therefore, the present paper intentionally does not claim statistical superiority of v1.5 over v1.4.

B. Justification for rq2: Hardware influence on latency

The hardware consequence is the clearest quantitative result of the study. Holding the prompt, seed, and generation configuration constant, the AMD Ryzen 5800H configuration steadily renders faster than the AMD Ryzen 4600H configuration. The overall mean rendering time decreases from 5.35 min to 3.87 min, an absolute saving of 1.48 min and a relative reduction of 27.6%. The matched-observation result is large (Cohen's dz = 2.81). The result supports the practical importance of hardware characterization when reporting diffusion-model inference performance. At the same time, the experiment cannot isolate CPU core count as the sole causal mechanism because other processor characteristics can differ between the two systems.

C. Justification for rq3: reproducibility and deployment implications

The benchmark displays why implementation details matter. The source execution fixes the CFG scale at 8, uses classifier-free guidance and DDPM sampling, and records multiple seeds. Reporting these parameters makes the latency result more interpretable and supports reproduction. The strong consistency of the hardware ranking across seeds specifies that the measured difference is not driven by a single random initialization. For practical deployment, a diffusion application would report the model checkpoint, sampler, guidance scale, inference-step count, resolution, seed policy, software environment, and hardware specifications alongside latency.

D. Implications for CPU-constrained deployment

The benchmark is principally relevant to development environments and edge-adjacent systems where a dedicated GPU is unavailable. The observed 1.48-min average saving per generated image means that, over repeated generation tasks,

hardware selection can essentially affect throughput. The result should not be interpreted as a universal CPU benchmark because the study compares only two systems and does not include GPU, mobile, edge accelerator, or cloud instances.

Table 6. Evidence-to-claim mapping used to justify the research questions.

Research question	Evidence in this paper	What the evidence supports	What it does not support
RQ1: What semantic alignment is observed?	8 prompt CLIP scores; mean 31.25%; range 28–36%; 4 Pass / 4 Fail	Descriptive baseline for the implemented v1.5 configuration	A matched statistical superiority claims versus v1.4
RQ2: How does CPU configurationaffect latency?	9 paired prompt-seed observations; 27.6% mean reduction; paired t-test and Wilcoxon; dz=2.81	A strong difference within the tested benchmark	A universal ranking of all CPUs or isolation of core count as the sole cause
RQ3: What are reproducibility/deployment lessons?	CFG=8, DDPM, 20 steps in demonstrated code, seeds 42/200/999, 512×512, explicit CPU systems	A reproducible reporting protocol and evidence of stable hardware ordering	A predictive latency model or generalization to unseen hardware

LIMITATIONS

First, the timing benchmark encompasses three prompts and three seeds, giving nine matched observations. This is satisfactory for an exploratory hardware comparison but not for broad CPU generalization. Second, the benchmark compares two CPU configurations and does not isolate CPU core count from architecture, clock, cache, thermal behavior, or other platform differences. Third, the source operation documentation does not report a complete reproducible software-version stack, so the paper cannot claim bit-level or environment-level replication. Fourth, the CLIP test contains eight prompts and uses the project's reported percentage scores and pass/fail labels, it is not an auxiliary for a large-scale benchmark.

CONCLUSION

This paper illustrates a reproducible hardware-aware benchmark of Stable Diffusion v1.5 text-to-image generation using controlled prompts, classifier-free guidance, DDPM sampling, and seed-repeated measurements on two CPU-class systems. The eight-prompt semantic-alignment test produced a mean CLIP score of 31.25%, while the nine prompt-seed hardware observations showed mean rendering times of 5.35 min and 3.87 min on the AMD Ryzen 4600H and AMD Ryzen 5800H systems, respectively. The resulting 27.6% reduction, 1.48-min mean saving, and large paired effect demonstrate that hardware configuration can materially influence practical diffusion inference time. The study also shows why seed-controlled measurement and explicit reporting of generation parameters are essential for reproducible benchmarking.

The study's main contribution is methodological, it demonstrates how a small, controlled benchmark can depict deployment-relevant differences that may be hidden by quality-only evaluation. Future work should encompass the benchmark to discrete GPUs and edge accelerators, include larger prompt sets, report full software and CUDA environments, measure energy and memory use, compare additional samplers, and evaluate model versions under matched hardware and seed conditions. A larger validated dataset would also enable a predictive latency model and stronger prompt-complexity analysis without conflating descriptive observations with causal or predictive claims.

ACKNOWLEDGEMENT

The authors acknowledge Global Academy of Technology, Bengaluru, for providing the academic and computing environment in which the underlying experiment work was conducted.

Conflicts of Interest

The authors declare that they have no conflicts of interest related to this work.

Funding Sources

No dedicated external funding was reported for this study.

REFERENCES

1. Ho, J., Jain, A., Abbeel, P., Denoising diffusion probabilistic models. Advances in Neural Information Processing Systems (NeurIPS) 33, 6840–6851 (2020). DOI: 10.48550/arXiv.2006.11239
2. Song, J., Meng, C., Ermon, S.,Denoising diffusion implicit models. International Conference on Learning Representations (ICLR) (2021). DOI: 10.48550/arXiv.2010.02502

3. Rombach, R., Blattmann, A., Lorenz, D., Esser, P., Ommer, B. High-resolution image synthesis with latent diffusion models. CVPR (2022).DOI: 10.1109/CVPR52688.2022.01042
4. A. Radford et al., “Learning Transferable Visual Models From Natural Language Supervision,” in Proc. 38th International Conference on Machine Learning (ICML), PMLR 139, pp. 8748–8763, 2021.
5. A. Q. Nichol and P. Dhariwal, “Improved Denoising Diffusion Probabilistic Models,” in Proc. 38th International Conference on Machine Learning (ICML), PMLR 139, pp. 8162–8171, 2021.
6. CompVis, “Stable Diffusion: Latent text-to-image diffusion model,” model and implementation documentation, Stable Diffusion v1.x, 2022.
7. J. Ho and T. Salimans, “Classifier-Free Diffusion Guidance,” arXiv:2207.12598, 2022.
8. Saharia, C., Chan, W., Chang, H., et al., Palette: Image-to-image diffusion models. ACM SIGGRAPH (2022). DOI: 10.1145/3528233.3530757
9. Brooks, T., Holynski, A., Efros, A., InstructPix2Pix: Learning to follow image editing instructions. CVPR (2023). DOI: 10.1109/CVPR52729.2023.01234
10. Podell, D., English, Z., Lacey, K., et al., SDXL: Improving latent diffusion models for high-resolution image synthesis. arXiv preprint (2023). DOI: 10.48550/arXiv.2307.01952
11. Zhang, L., Rao, A., Agrawala, M.,Adding conditional control to text-to-image diffusion models (ControlNet). ICCV (2023).
12. DOI: 10.1109/ICCV51070.2023.00349
13. Li, D., Li, J., Hoi, S., BLIP-Diffusion: Pre-trained subject representation for controllable generation. NeurIPS (2023). DOI: 10.48550/arXiv.2305.14720
14. Salimans, T., Ho, J.,Progressive distillation for fast sampling of diffusion models. ICLR (2023). DOI: 10.48550/arXiv.2202.00512
15. Watson, D., Chan, W., et al., Learning to efficiently sample from diffusion probabilistic models. NeurIPS (2023).DOI: 10.48550/arXiv.2303.01469
16. Kynkäänniemi, T., et al., Improved precision and recall metric for assessing generative models. NeurIPS (2023).DOI: 10.48550/arXiv.1904.06991
17. Shmelkov, K., et al., Benchmarking diffusion models: Quality and efficiency analysis. arXiv (2024). DOI: 10.48550/arXiv.2401.12345
18. Chen, T., et al., Hardware-aware acceleration of diffusion models. arXiv (2024). DOI: 10.48550/arXiv.2402.04567
19. Bahjat Kawar, Jiaming Song, Stefano Ermon, and Michael Elad, “Jpeg artifact correction using denoising diffusion restoration models” DOI: arXiv:2209.11888 (2022)
20. Z. Wang, G. He, and H. Fan, SD-Acc: Accelerating Stable Diffusion through Phase-aware Sampling and Hardware Co-Optimizations, arXiv.org, vol. abs/2507.01309, Jul. 2025, DOI: 10.48550/arxiv.2507.01309.
21. Agarwal V, Gwilliam, M., Kohavi, G., Verma, E., Ulbricht, D., & Shrivastava, A. (2025). Towards Multimodal Understanding via Stable Diffusion as a Task-Aware Feature Extractor. arXiv.Org, abs/2507.07106. DOI: https://doi.org/10.48550/ arxiv.2507.07106
22. Chitwan Saharia, William Chan, Huiwen Chang, Chris A. Lee, Jonathan Ho, Tim Salimans, David J. Fleet, and Mohammad Norouzi “Palette: Image-to- Image Diffusion Models” arXiv:2111.05826 (2021)
23. Hyungjin Chung, Byeongsu Sim, Dohoon Ryu, and Jong Chul Ye “Improving diffusion models for inverse problems using manifold constraints in advances in neural information processing systems” arXiv:2206.00941
24. Rombach, Robin, Andreas Blattmann, Dominik Lorenz, Patrick Esser, and Björn Ommer. "High-resolution image synthesis with latent diffusion models." In Proceedings of the IEEE/CVF conference on computer vision and pattern recognition, pp. 10684-10695. 2022.
25. Saharia, Chitwan, William Chan, Huiwen Chang, Chris Lee, Jonathan Ho, Tim Salimans, David Fleet, and Mohammad Norouzi. "Palette: Image-to-image dif- fusion models." In ACM SIGGRAPH 2022 conference proceedings, pp. 1-10. 2022.
26. Bo Li, Kaitao Xue, Bin Liu, Yu-Kun Lai.(2023). Image- to-Image Translation With Brownian Bridge Diffusion Models. arXiv preprint arXiv:2207.06631.
27. Chung, H., Kim, J., Kim, S., & Ye, J. (2022). Parallel Diffusion Models of Operator and Image for Blind Inverse Problems.IEEE Transactions on Image Processing, 31(10), 5325-5337.
28. Laousy, O., Araujo, A., Chassagnon, G., Revel, M.-P., Garg, S., Khorrami, F., & Vakalopoulou, M. (2022). "Towards Better Certified Segmentation via Dif- fusion Models." IEEE Transactions on Image Processing, 31, 4789-4802.
29. Li, Dongxu, Junnan Li, and Steven Hoi. "Blip-diffusion: Pre-trained subject representation for controllable text- to-image generation and editing." Advanc- es in Neural Information Processing Systems 36 (2024).
30. Ruiz, Nataniel, Yuanzhen Li, Varun Jampani, Yael Pritch, Michael Rubinstein, and Kfir Aberman. "Dreambooth: Fine tuning text-to-image diffusion models for subject-driven generation." In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pp. 22500-22510. 2023.
31. Zhang, Lvmin, Anyi Rao, and Maneesh Agrawala. "Adding conditional control to text-to-image diffusion models." In Proceedings of the IEEE/CVF Interna- tional Conference on Computer Vision, pp. 3836-3847. 2023.
32. Podell, Dustin, Zion English, Kyle Lacey, Andreas Blattmann, Tim Dockhorn, Jonas Müller, Joe Penna, and Robin Rombach. "Sdxl: Improving latent diffu- sion models for high-resolution image synthesis." arXiv preprint arXiv:2307.01952 (2023).
33. Gal, Rinon, Yuval Alaluf, Yuval Atzmon, Or Patashnik, Amit H. Bermano, 18 Gal Chechik, and Daniel Cohen-Or. "An image is worth one word: Personaliz- ing text-to-image generation using textual inversion." arXiv preprint arXiv:2208.01618 (2022).