Supplementary Materials for Paper #2212

MixDQ: Memory-Efficient Few-Step Text-to-Image Diffusion Models with Metric-Decoupled Mixed Precision Quantization

Tianchen Zhao[12*], Xuefei Ning[1*†], Tongcheng Fang[12*], Enshu Liu[1], Guyue Huang[3], Zinan Lin[4], Shengen Yan[2], Guohao Dai[25], and Yu Wang[1†]

[1] Tsinghua University
[2] Infinigence AI
[3] University of California Santa Barbara
[4] Microsoft Research
[5] Shanghai Jiaotong University

{suozhang1998, foxdoraame, sgcbflrftc, hguyue1, linzinan1995}@gmail.com,
les19@mails.tsinghua.edu.cn, yanshengen@infini-ai.com,
daiguohao@sjtu.edu.cn, yu-wang@mail.tsinghua.edu.cn

1 Further Experiments

We further compare MixDQ's performance with other quantization techniques for W8A8 in Tab. 1. The experiments are conducted on a subset of COCO annotations, with all other experimental settings maintained as described in Section 4.1 of the main paper.

For PTQD [1], we conduct linear regression to determine K. However, we observe a relatively lower linear correlation (0.59) compared to the original paper. In the case of 1-step generation, the sampling process is deterministic, the "uncorrelated noise correction" inapplicable, and only the "correlated noise correction" is adopted. Despite being effective for multi-step quantization, we observe that PTQD causes notable quality degradation for the challenging 1-step quantization. EDA-DM [4] improves upon the quantization parameter tuning in Q-Diffusion [3] with layer-wise reconstruction. However, only a marginal improvement is witnessed. The term "MP-only" refers to "mixed precision only" with the SQNR-based method [11], where due to suboptimal quantization sensitivity analysis, a significant performance degradation is still evident. The "Non-uniform" refers to the use of the FP8 (Floating-point 8-bit) format [7], which introduces exponential bits to handle a larger dynamic range of data distribution. While this notably improves performance, it still falls short of the FP16 baseline. In contrast, MixDQ achieves W8A8 quantization without any performance loss.

2 Application example of MixDQ

As discussed in the Section 3.3 and the Section 4 of the main paper, we introduce mixed-precision bit-width allocation methods based on integer programming.

* Equal contribution
† Corresponding Authors: Yu Wang and Xuefei Ning

Table 1: Comparison with other quantization techniques for W8A8 on COCO annotations subset. The "CLIP" and "IR" denotes CLIP Score and ImageReward metric. The metrics are evaluated with the first 1024 prompts in COCO annotations. The "MP-only" represents adopting the mixed precision only.

Model	Method	FID(↓)	CLIP(↑)	IR(↑)
SDXL-turbo (1 step)	FP16	84.51	0.26	0.84
	Naive PTQ	165.92 (+81.4)	0.15 (-0.11)	-1.72 (-2.56)
	PTQD [1]	340.74 (+256.2)	0.12 (-0.14)	-2.28 (-3.12)
	Q-Diffusion [3]	149.15 (+64.6)	0.16 (-0.10)	-1.69 (-2.53)
	EDA-DM [4]	137.98 (+53.5)	0.16 (-0.10)	-1.71 (-2.55)
	MP-only [11]	114.70 (+30.2)	0.15 (-0.11)	-0.61 (-1.45)
	Non-Uniform (FP8) [7]	101.73 (+17.2)	0.24 (-0.01)	0.16 (-1.45)
	MixDQ	83.39 (-1.12)	0.27 (+0.01)	0.84 (+0.00)

This approach proves effective in obtaining points on the Pareto frontier. In this section, we further provide a detailed description of an application case: **"How to determine the optimal bit-width configuration given a certain memory budget"**?

We provide a detailed description for MixDQ in this application in Algorithm 1. For simplicity, the algorithm only describes the bit-width allocation for weights, a similar process could be applied for activation bit-width allocation. Firstly, we perform sensitivity analysis after BOS-aware quantization (discussed in the main paper Sec. 3.1). Following the idea of "metric-decouple", we categorize all layers of the model into content-related layers and quality-related layers. Specifically, the content-related layers include cross-attention layers and feed-forward networks (FFNs), while the quality-related layers encompass self-attention layers, convolutions, and other remaining layers (further discussed in Sec. 5). Then we calculate the sensitivity separately for content/quality-related layers weight/activation, generating 4 groups of layer-wise relative sensitivities ($\mathcal{S}_{\text{content}}$ and $\mathcal{S}_{\text{quality}}$ in Algorithm 1).

Subsequently, we allocate the bit-width under budget constraints based on the acquired sensitivity. Specifically, we describe the budget with averaged bit-width (e.g., W5A8) multiplied by the number of parameters for weight and activation, respectively (the $\mathcal{B}_{\text{all}}$). We linearly scan through M nearby budgets $\mathcal{B}_j$ within range $[\mathcal{B}_{\text{all}} - \Delta\mathcal{B}, \mathcal{B}_{\text{all}}]$ to ensure we get the optimal result. Then, we introduce a hyperparameter K (we choose linear space from 0.45 to 1,36 for weights, and 0.94 to 1.09 for activation) to describe the ratio between budgets assigned for the content-related and quality-related groups. It satisfies: $\mathcal{B}_{j,\text{all}} = \mathcal{B}^{j,K}_{\text{quality}} + \mathcal{B}^{j,K}_{\text{content}}$. After acquiring the budget for content or quality-related layers, integer programming could be used to get the mixed precision configuration: $\{b^{j,K}_{l,\text{content}}\}$, which is a layer-wise choice of candidate bit-width 2,4,8, the l indexes the layers. The aforementioned process is iterated for each

$B_{j,\text{model}}$ and K, and the optimal configurations are determined through a rapid evaluation of the generated images.

Algorithm 1: MixDQ Bit-width Allocation Algorithm

Input: Sensitivity of content-related layers: $\mathcal{S}_{\text{content}}$, Sensitivity of quality-related layers: $\mathcal{S}_{\text{quality}}$, Parameter size of content-related layers : $\mathcal{P}_{\text{content}}$, Parameter size of quality-related layers : $\mathcal{P}_{\text{quality}}$, The target budget of the whole model, content and quality-related layers: $\mathcal{B}_{\text{all}}$, $\mathcal{B}_{\text{quality}}$,$\mathcal{B}_{\text{content}}$, The candidate budgets: $\{\mathcal{B}_j\} \in [\mathcal{B}_{\text{all}} - \Delta\mathcal{B}, \mathcal{B}_{\text{all}}]$ of length M, The candidate ratios: $\{k_i\}$ of length N, The FP model $\mathcal{M}$, the quantized model with bit-width configuration $\{b_l\}$: $\mathcal{M}_q^{\{b_l\}}$ **Output:** The mixed precision configuration $\{b_l\}$.

for $j = 1$ *to* M **do**
  for $k = k_0$ *to* k_{N-1} **do**
    `// Split the budgets for content and quality-related layers.`
    $\mathcal{B}^{j,k}_{\text{content}}, \mathcal{B}^{j,k}_{\text{quality}} = SplitBudget(\mathcal{B}_{j,\text{all}}, \mathcal{P}_{content}, \mathcal{P}_{quality}, k)$;
    `// perform integer programming respectively`
    $\{b^{u,k}_{l,\text{content}}\} = IntegerProgramming(\mathcal{B}^{u,k}_{\text{content}}, S_{\text{content}}, \mathcal{P}_{\text{content}})$;
    $\{b^{u,k}_{l,\text{quality}}\} = IntegerProgramming(\mathcal{B}^{u,k}_{\text{quality}}, S_{\text{quality}}, \mathcal{P}_{\text{quality}})$;
    `// get the mixed precision configuration for the whole model`
    $\{b^{u,k}_{l,\text{all}}\} = \{b^{u,k}_{l,quality}, b^{u,k}_{l,content}\}$;
    `// infer with the quantized model`
    $\text{imgs} = \mathcal{M}_{\mathcal{Q}}{}^{\{b^{u,k}_{l,\text{all}}\}}(\text{prompts})$;
    `// evaluate the generated images`
    $\text{score}_{u,k} = eval(\text{imgs})$;
`// choose the optimal configuration with best score`
$\{b_l\} = \underset{\{b^{u,k}_{l,\text{all}}\}}{\text{argmax}}(\text{score}_{u,k})$;
return $\{b_l\}$

3 Description of Challenges for Few-step Diffusion Quantization

As mentioned in the main paper Sec. 1 and Fig. 1, Fig. 2, the few-step diffusion model quantization faces the challenges of (1) The few-step diffusion models are more sensitive to quantization than multi-step ones. (2) The quantization's effect on image content causes degradation in image-text alignment. In this section, we provide a detailed discussion of these two challenges.

3.1 Few-step Diffusion Models are More Sensitive to Quantization

As depicted in Figure 1 of the main paper and discussed in Section 1, "the 2-step model exhibits significantly less degradation compared to the SDXL-turbo

1-step model". We delve into the reasons behind this phenomenon. Examining the challenge in Fig. 1, we contrast the generation processes of the 2-step and 1-step models.

In the 2-step generation, the second step incorporates an image with both Gaussian noise and quantization noise as input. Recent literature on diffusion-based image editing, as highlighted in [6, 9], indicates that diffusion models can effectively recover content from partially disrupted image input (image inpainting). Consequently, **the additional denoising steps (2nd or more) facilitate the reconstruction of quantization errors**, resulting in improved image quality.

In contrast, the 1-step diffusion model lacks this recovery phase, making it more "vulnerable" to quantization noise.

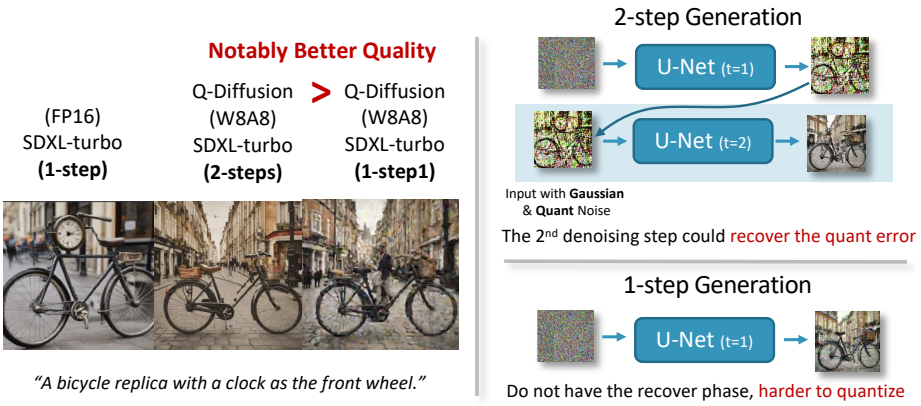

Fig. 1: The illustration of the few-step diffusion model is comparatively harder to quantize. Left: the 2-step SDXL-turbo quantized model exhibits notably better image quality than the 1-step one. Right: the explanation of why the 1-step model is harder to quantize.

3.2 Quantized Diffusion Models Lose Image-text Alignment

As outlined in Section 1 of the main paper, prior research predominantly concentrates on preserving image quality. Nevertheless, it's crucial to recognize that quantization affects not just the image quality but also the content itself. The alteration in content can result in the degradation of image-text alignment. In Fig. 2, we provide additional examples of alignment degradation, illustrating that such deterioration occurs not only in few-step models but also in less challenging multi-step diffusion models.

In the case of the prompt "A black Honda motorcycle parked in front of a garage," the W8A8 quantized SDXL-turbo 1-step model fails to include the "in front of garage" in the text instruction and instead generates an image of a man

riding a bicycle. With the multi-step SDXL model, the output deviates even further, presenting a yellow truck instead of the specified black motorcycle.

Similarly, for the prompt "A room with blue walls and a white sink and door," both the W8A8 quantized models generate an image of a white room while omitting the "blue walls" component. In contrast, our MixDQ W8A8 produces images with **significantly improved image-text alignment**. They fulfill all components of the prompt, closely resembling the images generated by the FP16 model.

Fig. 2: The illustration of the "Image-text Alignment loss" challenge for diffusion quantization. The Q-diffusion witnesses alignment loss (lacking of components described in the text instruction) for both the multi-step and few-step models.

4 Detailed Analysis of Hardware Experiments

In this section, we provide a detailed analysis of the hardware resource savings for different bit-width configurations and conclude our findings as follows:

(1) Illustrated in Figure 7b of the main paper Sec. 4.3, **the W8A8 bar displays an almost negligible red segment**, corresponding to the "1% most sensitive activation layers retained from FP16" as discussed in Sec. 4.2 of the main paper. This component introduces minimal overhead while significantly contributing to the preservation of image quality.

(2) Observing the table in Figure 7 of the main paper Sec. 4.3, we note that **W4A16 quantization solely achieves memory savings, while the latency remains consistent with the FP16 baseline**. The reduction in memory primarily stems from the decreased size of the model parameters, which are transferred to the GPU and serve as the "static memory cost." However, the

W4A16 model continues to utilize FP16 computation, thus maintaining similar latency.

(3) Observing the table in Figure 7 of the main paper Sec. 4.3, **W4A16 and W4A8 exhibit similar memory optimization**. The key distinction lies in the fact that W4A8 quantizes the activations, thereby reducing the memory cost through saved activations, primarily affecting the shortcut feature for U-Net. However, we discover that the occupied size by these activations is relatively small, amounting to less than 100MB compared to the model size of 6GB. Consequently, the additional savings attributed to activation quantization are not evident.

(4) Observing the table in Figure 7 of the main paper Sec. 4.3, **both W4A8 and W8A8 demonstrate a noteworthy 1.52× latency speedup compared to the FP16 baseline**. This acceleration is attributed to the utilization of INT8 GPU kernels. It's worth noting that we haven't implemented the INT4 operators, making W4A8 and W8A8 quite similar in terms of latency improvement. However, W4A8 has the potential to achieve even higher latency speedup.

5 Justification of Layer Grouping in Metric Decouple

As mentioned in the main paper Sec. 3.2, we split the layers into two groups based on their effect on image content and quality. Specifically, the cross-attention and feed-forward layers are regarded as the "content-related layers", and the convolution and self-attention layers as the "quality-related layers". In this section, we present experimental results to **demonstrate the rationale of the layer grouping**.

Firstly, we summarize 5 layer types for the text-to-image diffusion model: cross-attention layers, feed-forward neural networks (FFNs), convolutions, self-attention layers, and other miscellaneous layers. We independently quantize the weight and activation into 8-bit for each layer type, and compute their effects on the SSIM score. As seen in Fig. 3, the cross-attention and feed-forward layers have a significantly larger influence (0.8 and 0.05) on the image content compared with other layer types (less than 0.01). Therefore, we choose them as the "content-related layers", and the rest as "quality-related" layers.

6 Qualitative Results

In this section, we provide more qualitative results to demonstrate the effectiveness of our MixDQ, especially in terms of **preserving the "image-text alignment"**, As can be seen from Fig. 4, even under W4A8, our method generates images with visual quality and image-text alignment similar to the FP16 model generated ones.

Fig. 5 highlights the superior text-image alignment achieved by MixDQ in comparison to baseline quantization methods. With W8A8 quantization, the Q-diffusion-generated images not only exhibit an "oil-painting" like blurring and tiny artifacts but also deviate from following the text instructions. For instance,

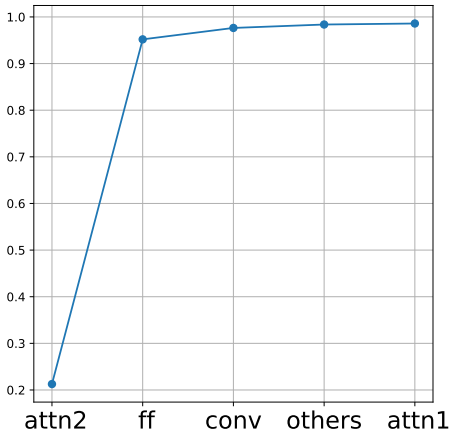

Fig. 3: SSIM Score when quantizing a certain group. Cross-attention (CrossAttn) and Feed-Forward Networks (FFNs) are the two types of layers that have the greatest impact on content.

it omits "in front of a garage" from the prompt and, instead, generates an image of a man riding a motorcycle. It also struggles to interpret “an older man skiing”; instead, it generates a cowboy riding a strange two-headed horse. Moreover, the quantized model loses the capability to adhere to explicit color instructions like "blue walls" and "gray and white kitten." In contrast, with the more challenging W4A8 setting, our MixDQ quantized model produces images with nearly identical content compared to the FP16 baseline.

7 Analysis of Quantization Method

Takeaway Knowledge of Layer Sensitivity. We conclude key takeaways of layer sensitivity for text-to-image diffusion models as follows: (1) The CrossAttn & FFN affects image content while the SelfAttn & Conv affect quality. This is illustrated in Figure 5 and Section 3.2 of the main paper. (2) The activation for “conv_in” and weight for “conv_out” are highly sensitive compared with other convolution layers. (3) The sensitivity of activation for “to_k/v” layers notably decreases after BOS-aware quantization, but their weight’s sensitivity remain high. (4) In contrast to multi-step models, time embedding is not sensitive for few-step models. This fits intuition since distillation enables the network to denoise from any timestep.

Properties of MixDQ Quantization. (1) **Efficient Quantization**: Previous quantization methods require costly optimization of extensive AdaRound and scaling factor parameters, which takes tens of GPU hours. This process needs to be conducted repeatedly for different bit-width configurations. MixDQ adopts

Fig. 4: The comparison between images generated by full precision model and quantized one. Text-guided image generation from our W4A8 quantized SDXL-turbo with a fixed random seed.

the naive minmax quantization without bells and whistles. After obtaining sensitivity, it only requires several minutes to acquire the entire Pareto frontier. (2) **Hardware Friendly.** When designing quantization schemes, we prioritize making them hardware-friendly to minimize the hardware overhead for supporting them. This includes employing tensor-wise scaling for activations and utilizing output-channel-wise scaling for weights. The granularity of mixed precision is set at the layer level, enabling the utilization of existing kernels for hardware resource savings and avoiding the cost of implementing customized kernels. (3) **Flexibility.** The framework of MixDQ is flexible and extensible, allowing it to be combined with other quantization techniques such as efficient QAT [10].

8 Limitations and Future Directions

We introduce MixDQ, a mixed-precision quantization method that handles both the imbalance sensitivity and alignment degradation problems for diffusion quantization. We summarize the current limitations and potential future improvements as follows:

- **Specialized Quantization Techniques for other Sensitive Layers.** In MixDQ, we pinpoint the bottleneck in text embedding quantization and craft BOS-aware quantization specifically tailored for it. Nevertheless, there are other highly sensitive layers, such as "conv-in" and "conv-out," that could also gain advantages from the design of specialized quantization techniques.

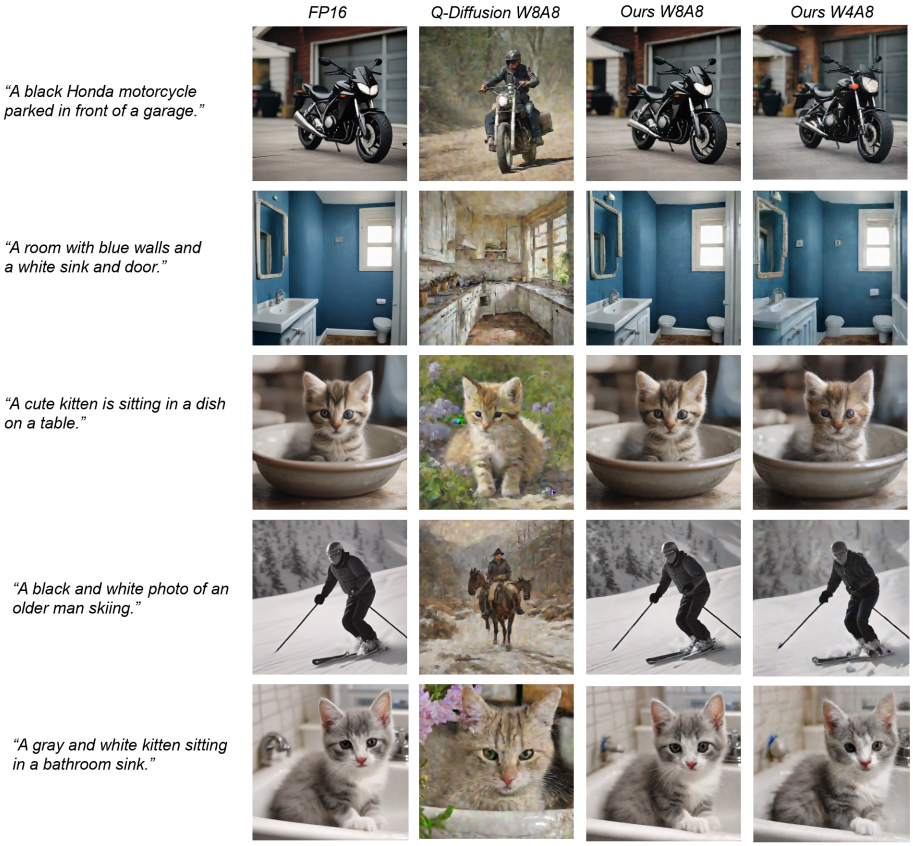

Fig. 5: The comparison between images generated by quatized model. Our quantization scheme better maintains the content and quality of the images.

- **Combine with Improved Quantization Techniques.** In MixDQ, we employ naive min-max quantization as the quantization method. Its performance can be enhanced by incorporating advanced quantization techniques, such as Adaround [8], or by introducing quantization-aware training [2].
- **Lower Bit-width.** In MixDQ, we opt for the 2, 4, and 8 as candidate bit-widths. When coupled with efficient lower-bit quantization methods, the mixed-precision bit-width allocation remains compatible with lower-bit-width combinations.
- **More Hardware Supports.** As discussed in Sec. 4, we only utilize the INT8 GPU kernels for now, which restricts the performance for W4 quantization. The newest Nvidia TensorCore [5] supports INT4 computing, which could be further utilized for better efficiency.

References

1. He, Y., Liu, L., Liu, J., Wu, W., Zhou, H., Zhuang, B.: Ptqd: Accurate post-training quantization for diffusion models. ArXiv **abs/2305.10657** (2023), https://api.semanticscholar.org/CorpusID:258762678
2. Jacob, B., Kligys, S., Chen, B., Zhu, M., Tang, M., Howard, A.G., Adam, H., Kalenichenko, D.: Quantization and training of neural networks for efficient integer-arithmetic-only inference. 2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition pp. 2704–2713 (2017), https://api.semanticscholar.org/CorpusID:39867659
3. Li, X., Lian, L., Liu, Y., Yang, H., Dong, Z., Kang, D., Zhang, S., Keutzer, K.: Q-diffusion: Quantizing diffusion models. ICCV (2023)
4. Liu, X., Li, Z., Xiao, J., Gu, Q.: Enhanced distribution alignment for post-training quantization of diffusion models. arXiv preprint arXiv:2401.04585 (2024)
5. Markidis, S., Chien, S.W.D., Laure, E., Peng, I.B., Vetter, J.S.: Nvidia tensor core programmability, performance & precision. 2018 IEEE International Parallel and Distributed Processing Symposium Workshops (IPDPSW) pp. 522–531 (2018), https://api.semanticscholar.org/CorpusID:3887305
6. Meng, C., He, Y., Song, Y., Song, J., Wu, J., Zhu, J.Y., Ermon, S.: Sdedit: Guided image synthesis and editing with stochastic differential equations. In: International Conference on Learning Representations (2021), https://api.semanticscholar.org/CorpusID:245704504
7. Micikevicius, P., Stosic, D., Burgess, N., Cornea, M., Dubey, P., Grisenthwaite, R., Ha, S., Heinecke, A., Judd, P., Kamalu, J., et al.: Fp8 formats for deep learning. arXiv preprint arXiv:2209.05433 (2022)
8. Nagel, M., Amjad, R.A., van Baalen, M., Louizos, C., Blankevoort, T.: Up or down? adaptive rounding for post-training quantization. ArXiv **abs/2004.10568** (2020), https://api.semanticscholar.org/CorpusID:216056295
9. Nichol, A., Dhariwal, P., Ramesh, A., Shyam, P., Mishkin, P., McGrew, B., Sutskever, I., Chen, M.: Glide: Towards photorealistic image generation and editing with text-guided diffusion models. arXiv preprint arXiv:2112.10741 (2021)
10. Wang, H., Shang, Y., Yuan, Z., Wu, J., Yan, Y.: Quest: Low-bit diffusion model quantization via efficient selective finetuning. ArXiv **abs/2402.03666** (2024), https://api.semanticscholar.org/CorpusID:267500241
11. Yang, Y., Dai, X., Wang, J., Zhang, P., Zhang, H.: Efficient quantization strategies for latent diffusion models. ArXiv **abs/2312.05431** (2023), https://api.semanticscholar.org/CorpusID:266163100