MIXED PRECISION INFERENCE FOR DIFFUSION LANGUAGE MODELS

Anonymous authors
Paper under double-blind review

ABSTRACT

Diffusion Language Models (DLMs) have drawn increasing attention as an alternative to classical Large Language Models architecture due to their potential to significantly increase inference speed. However, they lack in practice some key optimizations. One such feature is mixed-precision inference, i.e., using a quantized or a full precision model at different steps of the generation process. This hybrid design enables one to benefit from the accelerated inference of quantized models while maintaining an output quality similar to that of original floating-point precision. As existing approaches are not applicable to this setup, we propose here the first mixed precision framework specifically targeted for DLMs. By modeling the quantization error, we are able to design a training-free, adaptive mechanism to successfully exploit the speed of a quantized model and the precision of a regular one. Our results show we can achieve a 66% speed increase without significant degradation of the model's performance. We also demonstrate this can be combined with other acceleration systems to further speed up the sampling process.

Our code is made available at `https://anonymous.4open.science/r/MPQDLM-ICLR27/`.

1 INTRODUCTION

The commonly used autoregressive (AR) language models generate text one token at a time. Each new token depends on the previously generated ones, which makes generation inherently sequential and slow. Diffusion Language Models (DLMs) (Austin et al., 2021a; Lou et al., 2024; Li et al., 2022; Hoogeboom et al., 2021) aim to reduce this bottleneck and promise higher token throughput. Instead of producing only one new token per model inference, they predict a group of tokens at once and refine them iteratively. This provides more parallel work for the GPU by allowing model weights to be reused across multiple token positions. However, DLMs still require several such refinement passes until all tokens of a group are committed, so their inference cost remains substantial.

Quantization is a natural way to reduce the cost of these repeated model evaluations. Reducing the weight precision only reduces model size and memory traffic, which is particularly effective for memory-bound AR decoding. DLMs, however, process many token positions together and therefore expose more computation to the accelerators. To also reduce this computational cost, both weights and activations need to be quantized to formats natively supported by the hardware for efficient matrix multiplication. Such quantization can provide substantial acceleration, but it also introduces numerical errors affecting generation quality.

Using the same precision for every DLM refinement step assumes that every step can tolerate the same amount of quantization error, which is generally not the case. During inference, the set of refined and committed tokens changes after every step. Quantization errors may therefore have almost no effect at one step, but change which tokens are committed at another and thereby also affect subsequent steps. Similar observations have motivated dynamic precision for AR language models (Chen et al., 2024; Kwon et al., 2026) and temporal precision scheduling for diffusion models (Lewandowski et al., 2026), but neither directly solves the DLM case. AR methods are designed around token-by-token decoding, while continuous diffusion models propagate numerical errors through a continuous latent space. DLMs instead repeatedly update a partly generated sequence and

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

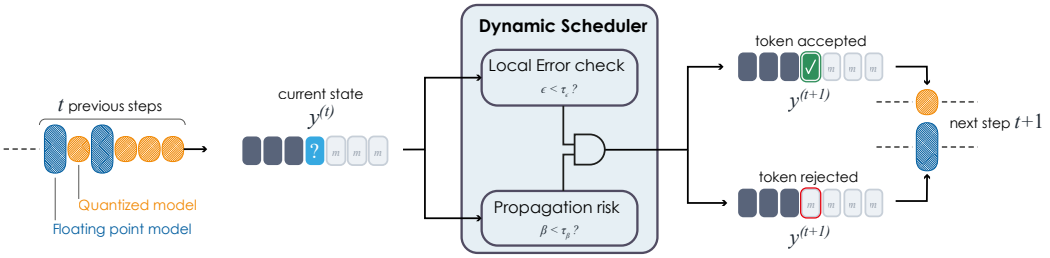


Figure 1: Overview of the mixed precision inference framework

make discrete token decisions. Their precision therefore has to be adapted using signals that reflect these specific error dynamics.

In this work, we propose the first mixed precision framework to overcome these limitations and create a solution specifically dedicated to DLMs. We first establish a linear bound on the expected quantization error and show it correlates with experimental observations. We also study the dynamics of quantization error propagation to propose a robust model of the error evolution. These findings enable us to build a mixed precision system that efficiently balances speed and quality to improve DLM inference, as illustrated in Figure 1.

Our contributions are as follows:

- We derive a model for expected quantization error using uncertainty. In addition, we propose a linear relationship between entropy velocity and quantization error, based on experimental data.
- Leveraging these findings, we create an algorithm for mixed precision inference. Relying on inference statistics, this algorithm is agnostic of the model and quantization method used, and does not require any training.
- We evaluate this algorithm on various tasks and inference setups, showing strong performance. We further consider a wider context of application, demonstrating it pairs successfully with exiting acceleration methods and can support more than two inference formats.

2 Related Works

2.1 Accelerated Inference

Mixed precision inference has been previously considered for AR models. Earlier approaches focused on finding an optimal mix of inference formats to apply to all samples (Rakka et al., 2025). More modern approaches acknowledge that this problem is better addressed at a per-sample level and have been developed accordingly. DP-LLM (Kwon et al., 2026) uses a hybrid approach to dynamically assign a precision to each layer of the model. DMR (Li et al., 2026a) trains a router network from a mix of supervised and reinforcement learning to send different requests to different models during the interaction with a user. PMPD (Chen et al., 2024) also trains a classifier network to pick the optimal model based on the KV-cache values. EvoPress (Sieberling et al., 2025) relies on evolutionary search to determine the optimal sparsity of each layer of a model. FlexQuant (Liu et al., 2025) integrates token-wise inference model switching based on measured "perplexity entropy."

DLM inference acceleration often involves designing new dynamic token unmasking policies. Methods often use the uncertainty of a step's output to decide which tokens should be unmasked (Wu et al., 2026; Kasa et al., 2026; He et al., 2025). Li et al. (2026b) uses a trained model to predict the optimal tokens. Prophet (Li et al., 2025b) uses top-2 uncertainty evolution to identify the tokens more likely to have converged towards their final values. Other approaches focus on speculative decoding with different drafting strategies (Liu et al., 2024; Li et al., 2025a; Sandler et al., 2025; Christopher et al., 2025).

Quantization is a widespread method to improve the efficiency of trained models. While many techniques have been designed towards diffusion models or AR LLMs, there remain for now few methods specifically targeting DLMs. Quant-dLLM (Zhang et al., 2025) targets weight-only quantization for DLMs. DLLMQuant (Xu et al., 2026) also includes activation quantization but is not

associated with a codebase for now. As demonstrated by Lin et al. (2025), applying AR quantization techniques to DLMs can produce decent results.

2.2 Masked Diffusion Language Models

Masked diffusion language models (Austin et al., 2021a; Sahoo et al., 2024; He et al., 2023) were introduced as a way to generate text following a diffusion pattern. We summarize their behavior here.

Forward process (masking). The forward process gradually masks tokens. At timestep t, the conditional distribution of the corrupted sequence $x^{(t)}$ given the clean data x is

$$q_t(x^{(t)} \mid x) = \prod_{i=1}^{L} \Big[(1-\alpha_t)\mathbb{1}[x_i^{(t)} = x_i] + \alpha_t \mathbb{1}[x_i^{(t)} = \mathbf{m}]\Big], \tag{1}$$

where $\alpha_t \in [0,1]$ is a predefined noise schedule (increasing with t) and $\mathbf{m}$ denotes the mask token. Each position is corrupted independently: a token remains unchanged with probability $1-\alpha_t$ and is replaced by $\mathbf{m}$ with probability α_t.

Reverse process (inference). Inference is a T-step iterative denoising procedure that starts from a fully masked sequence and progressively reveals tokens. Let $y^{(t)}$ denote the sequence after t denoising steps, with $y^{(0)}$ consisting entirely of mask tokens. We use a learned mask predictor p_θ and a predefined schedule of mask counts $m_0, \ldots, m_T$ where $m_0 = L$, $m_T = 0$, and m_t is strictly decreasing.

At step t (where $y^{(t)}$ contains m_t mask tokens), we predict a provisional fully unmasked sequence via greedy decoding:

$$\hat{y}_i^{(t)} = \arg\max_{v \in [V]} p_\theta\big(v,\ y^{(t)}\big), \qquad i = 1, \ldots, L, \tag{2}$$

with $[V] = \{1, \ldots, V\}$ the vocabulary. For each position we define the confidence score p as the predicted probability of the chosen token:

$$p := p_\theta(\cdot \mid y^{(t)}), \tag{3}$$

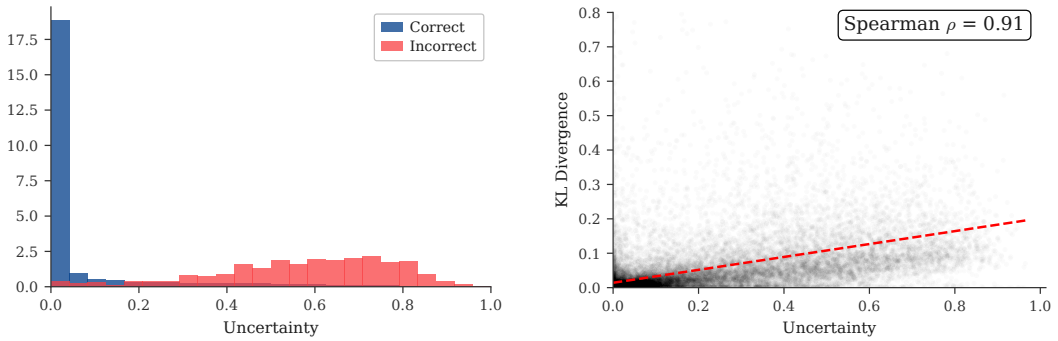
We then form the next sequence $y^{(t+1)}$ by keeping the predictions at the $L - m_{t+1}$ positions with the *highest* confidence and setting the remaining m_{t+1} positions to the mask token. Equivalently, we remask the m_{t+1} positions with the smallest p_i values. The schedule m_t (or, equivalently, the masking ratio m_t/L) decreases over steps, controlling the coarse-to-fine generation. After T iterations, $y^{(T)}$ is the final generated sequence, which is fully unmasked. In practice, modern DLMs use a semi-autoregressive setup where unmasking is not possible on the entire sequence $y^{(t)}$ but on predetermined *blocks* of indices, progressively increasing over time.

3 The dynamics of error propagation

In the following section, we will highlight the different dynamics of quantization error in the denoising process and propose a model to predict them. We first establish a link between uncertainty and quantization error at the step level (assuming the input of both models is the same), then analyze the propagation of the error.

3.1 Per-step quantization error

In classic continuous-space diffusion models, the quantization error accumulates monotonically with the denoising schedule (Lee et al., 2025). This occurs because the latent accumulates a deviation at each step, causing an increasing divergence from the undisturbed path. Such behavior cannot be directly transposed to the case of masked DLMs. Since they run in a discrete space, only part of the quantization error propagates to the subsequent steps. Indeed, the only discrepancy between the correct trajectory and the erroneous one is the one caused by committing different tokens. Most of the quantization error is therefore "masked" as it does not change the output tokens.



(a) Uncertainty distribution (density) for correct vs. incorrect quantized token generation.

(b) Uncertainty correlation to quantization error (KL Divergence w.r.t. fp inference) on a quantized output.

Figure 2: We confirm the strong link between uncertainty ϵ and quantization error by showing how uncertainty is a strong discriminator for correct/incorrect tokens (a) and how it correlates with the error e (b).

The immediate way of defining quantization error is the discrepancy between a model output and its quantized counterpart.

$$e_i := D_{\mathrm{KL}}\left(p \parallel \hat{p}\right) = \sum_{v=1}^{V} p_v \log \frac{p_v}{\hat{p}_v}. \tag{4}$$

We refer to this value as the local error e, as it accounts for the error at a given step t of the denoising process. Notably, the input y is here the same for both the quantized $\hat{p}$ and unquantized p model. This paradigm is the default expression of the quantization error, as it is common to all models where it is used.

The upside of this error is that we can find direct estimators for its value, for instance taking uncertainty. We define the *uncertainty*[1] ϵ_i of a model output as the gap between its top probability and the maximum value. Let $k = \arg\max_{v \in [V]} p_v$, so that

$$\epsilon_i := 1 - p_k \tag{5}$$

In practice, the quantization error e_i is expensive to measure since it requires a ground-truth reference. On the other hand, uncertainty ϵ does not incur any additional computations and remains consistent between quantized and floating point (fp) models. Intuitively, we can envision a link between the two values based on the idea that a more uncertain model is more likely to generate errors. We show the expected quantization error $\mathbb{E}[e_i]$ is approximately linear in uncertainty ϵ_i.

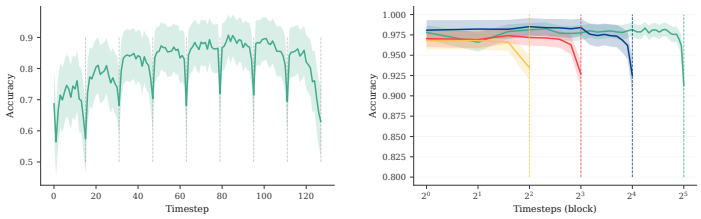
Theorem 3.1. *Assuming the perturbation noise is bounded, homogeneous and i.i.d. (assumptions 1 and 2), we have:*

$$\mathbb{E}[e_i] = \Theta(\epsilon_i)$$

Refer to Section B.2 for the full proof.

This opens the way to the approximation of the quantization error by the uncertainty. This relation is supported empirically, as we observe in our experiments a strong correlation between the error and the uncertainty of the model output. The values reported in Figure 2b indicate this behavior is consistent throughout the generation process. We also measure the uncertainty values while binning the tokens that are correctly output by the quantized model (i.e. identical to the floating point version) or not. Values are collected throughout the inference (128 steps) with prompts taken from the general purpose GPT4All dataset (Anand et al., 2023). The results reported in Figure 2a show there is a clear distinction in the distributions of these tokens, with correct tokens being predominantly low-uncertainty. This confirms we can use uncertainty as a cheap metric to predict per-step quantization errors for DLMs.

[1]The literature sometimes directly exploits the confidence p instead of the uncertainty, we prefer the later as it correlates positively with the error.



(a) Evolution of the accuracy through the 128 inference steps. Dashed lines indicate the boundaries of each block.

(b) Accuracy evolution averaged over all the blocks. Each color denotes a different block size: 4, 8, 16, 32.

Figure 3: Analysis of the error evolution over the entire inference.

One consequence of this relationship is that we can predict the average evolution of the error throughout the denoising process. As unmasking progresses, uncertainty decreases and so does the quantization error. As shown in Figure 3, there is a noticeable trend of decreasing error as the generation progresses. Reporting accuracy (the proportion of identical unmasked tokens between the quantized and floating-point models) shows an increase throughout the entire inference course. However, this trend is not monotonic, as we can observe error spikes on the last denoised token of each block. This happens because as the model gets closer to the last token of the block, the set of tokens available for unmasking shrinks. Since the leftover tokens are the most uncertain ones, reaching the end of the block means the sampler has to eventually commit these erratic tokens. This phenomenon is consistent across block sizes, as shown in Figure 3.

3.2 Error propagation

The pattern exposed above enables us to better predict quantization error, but it only works at a per-step level. In a real inference process, the errors are propagated and lead the decoding trajectory to progressively diverge. Because the output of one step is the input of the next one, end-to-end quantization error requires deeper analysis.

Indeed, it is understood that not all quantization error is equally bad. Both AR and diffusion literature for instance notice that error in the earlier phase tends to propagate to subsequent steps and is therefore more impactful (Huang et al., 2026; Lewandowski et al., 2026). Operating in a discrete domain such as DLMs only makes matters harder as only an error modifying the ranking of the top logits will lead to a modification of the output. In a well quantized setup, most of the errors are thus invisible, either because they are not large enough to modify the output or because they are not impactful enough to cause a meaningful shift in the generated text.

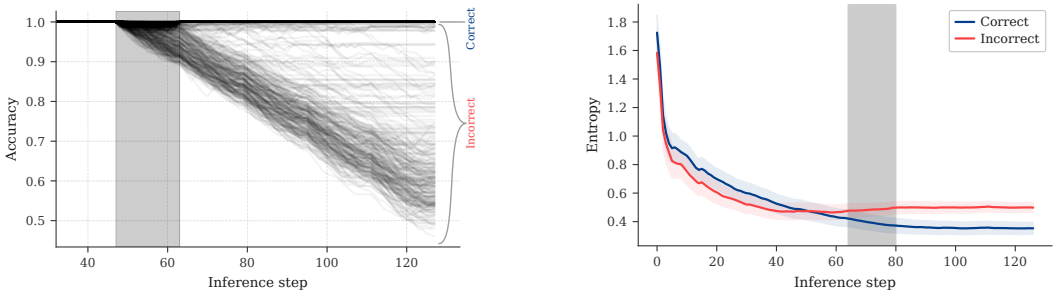
Hence, we distinguish between the *minor* errors and the *disruptive* ones. A disruptive error causes a trajectory shift, causing a significant part of the generated text to diverge. In contrast, a minor one remains invisible or limited to a small scale (the unmasked token remains the same or the change is not significant). It is therefore key to identify the tokens more likely to produce disruptive errors; this risk not only depends on the distribution of token probability across the vocabulary (more skewed distributions are less prone to errors) but also on the position of the token in the generated sequence.

To pinpoint which tokens are most likely to propagate an error, we rely on the local evolution of the denoising entropy. We compute the entropy velocity as the slope of the entropy of the unmasked tokens over the last decoding steps. Let $\alpha, \beta \in \mathbb{R}$ such that:

$$\alpha, \beta = \arg\min_{a,b} \sum_{k=t-n}^{t} \|H(p_k) - (a + bk)\|$$

We define β as the entropy velocity over the last n steps. We argue a growing entropy in the unmasked tokens (i.e. $\beta > 0$) is the sign that the model is likely to engage in a faulty sequence. This indicates the tokens picked for unmasking are ambiguous and that there is no clear path for the remainder of the sequence. Crucially, this does not rely solely on the uncertainty of a model's output (as in the previous subsection), but it considers the dynamics over a whole window of preceding steps. The two criteria are therefore complementary and target different aspects of the quantization error.

We report in Figure 4 the results of our experiments on error propagation. On a standard generation, we switch the model used during one block (16 steps/32 tokens) and measure the error throughout



(a) Token accuracy (quantized vs. unquantized) over the inference. Each line is a different sample.

(b) Logit entropy over the course of the unmasking process.

Figure 4: To highlight the impact of entropy velocity on error propagation, we use a quantized model during a single block of denoising (grayed out area) and report the evolution of accuracy (a) and entropy (b). We observe in (a) that only part of the error caused by quantization goes on to be propagated until the end of the process. Looking at the difference between the samples where error is propagated to the end result ("Incorrect") or not ("Correct"), we notice in (b) they show a discrepancy in the trend followed by their output's entropy.

the inference sequence. Figure 4a shows the accuracy evolution of each sample generated, making the ones where error propagates clearly distinguishable from the ones that do not. For a part of the sequences generated (40%), the errors generated by the quantized model do not propagate and the final sequence is identical to what it would have been with only fp steps (those are the constant lines with 100% accuracy in the figure). This is typically the case where minor errors dominate, i.e. they are not impactful enough to cause a token flip. In the remaining samples however, the error does get propagated into a significant change that is observable in the end.

Comparing the entropy of these two sets of samples, we can observe in Figure 4b that the distinction is here also very noticeable. Looking at the quantized section (grayed out), the trend appears to be opposite between erroneous and correct sequences. Correct sequences show a downward trend ($\beta < 0$), while incorrect ones are increasing ($\beta > 0$), thus corroborating our hypothesis. The evolution of the entropy indicates which tokens are more likely to generate an error on average. Critically, this trend predates the quantized block, indicating that the output quality degradation it induces is not the only cause, but rather, that it occurs at a point where the generation trajectory is unsettled. Thus, the entropy velocity highlights the regions where the quantization error causes the most damage.

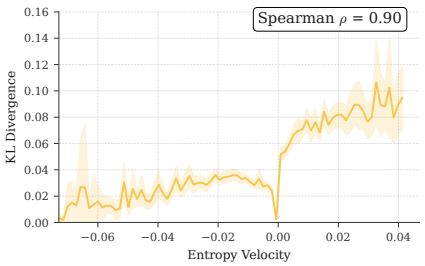


Figure 5: Quantization error depending on entropy velocity β. Aside from values at the $\beta = 0$ neighborhood that do not exhibit a clear pattern, we observe overall a linear trend with positive velocity resulting in larger error values. The shaded region shows 99% confidence interval.

To broaden this observation, we compute the error with regard to the entropy velocity in a general case, not anymore considering a single block. The results we report in Figure 5 confirm there is overall a clear linear correlation (Pearson $r = 0.89$) between the error e and the velocity β. The neighborhood of 0 ($\beta \approx 0$) appears as an outlier on this global trend, as values where the entropy is constant are on average the ones where there is the least error ($e \approx 0$). A constant entropy ($\beta = 0$) suggests the context clearly indicates what should be the next token, with little risk of error.

4 Methodology

Exploiting the error model analyzed in the previous section, we design a framework to distribute inference between quantized and default floating point (FP) models. We construct two versions of the method, depending on whether we run a calibration step or not. As our experimental results will show, the calibration step is lightweight and enables slight improvements in the results. More

importantly, calibration provides fine-grained control over the inference budget, enabling precise management of the balance between accuracy and speed.

4.1 A ROBUST RULE FOR MIXED INFERENCE

We aim to discard the most probable error sources in the generated sequence by leveraging our prior findings on its evolution. To do so, we design two complementary rules to decide whether or not a given step is reliable. The first one is aimed at the immediate, per-step quantization error, as defined in Equation (4). Since we have established that this error is approximately linear with the uncertainty (Theorem 3.1), we can rely on the measured uncertainty to identify tokens that are more prone to error. Therefore, we establish a threshold $\tau_\epsilon(t)$ to determine whether the risk of error is acceptable or not. This threshold depends on the inference step t, as we have previously reported that the error rate was dependent on the progress of the denoising.

Additionally, we introduce a second criterion to account for the error propagation. Following the findings of Section 3.2, we use the entropy velocity β as a proxy of the propagation risk at a given step. We thus establish a second threshold τ_β which is aimed at mitigating the risk of the token propagating an error. We keep the threshold τ_β constant throughout the inference as our data shows it does not provide significant improvement. Moreover, it would also increase the risk of targeting the same error as the first threshold τ_ϵ, thus providing no improved insight.

When any of these thresholds is reached, we do not proceed to unmask the token and change the model to use at next step to the fp one. This way, we aim to ensure the unmasking of the token does not lead to a quantization error. In the case where only part of the candidate tokens reaches a threshold, the remaining ones get committed normally. The rejection of a token causes a "delay" in the inference as the number of unmasked tokens is lower than what is originally planned by the scheduler. To fix this, we add some additional inference steps at the end of the block to recover the additional tokens missed because of the error. Naturally, these extra inference steps incur additional computational costs, which is why setting the parameters allows our method to strike a balance between accuracy (more correction) and speed (fewer fp steps).

Finally, we fix the first few timesteps to use a floating-point model in order to prevent unreliable entropy evolution tracking due to a too short window. We find an 8-step warm start is sufficient to prevent any erratic behavior.

4.2 CALIBRATED REFINEMENT

The mixed precision scheduler presented shows the benefit of being immediately applicable without any prior data. However, it has two drawbacks: First, it relies on setting an adequate value for the thresholds τ_ϵ and τ_β, which is hard to do without any prior knowledge. Second, it is not possible to tune it to follow a specified budget. The ratio of quantized steps used and the acceleration provided are not predictable.

To address these limitations, we introduce a calibration step that eliminates the need for a precise setting of the parameters. To do so, we run a reduced set of samples and measure the uncertainty and entropy velocity throughout the inference. It is then possible to determine the optimal thresholds for a given budget by using the value that corresponds to the respective proportion of timesteps (e.g. setting τ_ϵ as the third quartile of measured uncertainty to achieve 75% of quantized steps below the threshold).

5 EXPERIMENTAL RESULTS

5.1 EXPERIMENTAL SETUP

Models. We primarily use the LLaDA-8B model (Nie et al., 2025) in our experiments. This is a standard model for masked diffusion and it has been proven reliable in various domains. It was trained on 2.3 trillion tokens spanning various topics. We also use Dream (Ye et al., 2025) to verify that our method results remain consistent across models. These models were originally proposed with a "top-k" unmasking logic which consists of unmasking the tokens with the lowest uncertainty.

Table 1: Scores for the LLaDA-8B Instruct model on several datasets with different mixed precision scheduling strategies. Number of few-shot samples is indicated in parentheses.

	GSM8K (4)	MATH (0)	MBPP (3)	Human (0)	BBH (0)	IFEval (0)
FP16	74.8	31.6	27.8	31.7	50.8	45.5
W4A4	69.8	28.2	25.2	24.3	48.0	42.5
Static	72.4	29.4	26.6	30.1	48.4	44.4
PMPD	72.8	30.5	27.0	31.1	49.4	44.7
Ours	74.7	31.9	28.6	29.9	49.1	45.8

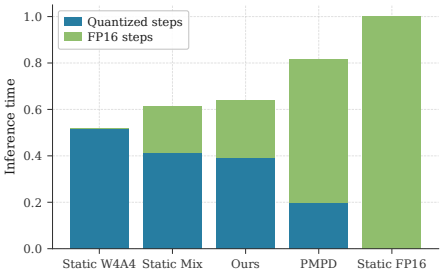


Figure 6: Inference mix of the different methods tested. We set the reference time to 1.0 when using exclusively the fp model for all steps.

Baselines. In addition to the purely quantized or floating point setups, we consider two baselines as comparison to our proposed approach. **Static** denotes always selecting the same steps to be quantized during the inference. Our experiments show that, for a fixed budget, the most efficient naïve strategy is to group all fp steps at the beginning (cf. Figure 12). We also consider an adaptation of the scheduling policy **PMPD** proposed by Chen et al. (2024). It relies on training a classifier on the KV cache values to decide which model to use. The original approach also proposed finding the optimal pair of models through a calibration step, but we only consider here the scheduling method for easier comparison.

More details on the experimental setup are given in Section A.1, including information on the **Inference parameters**, **Quantization** methods and **Datasets** used.

5.2 Main Results

Running mixed-precision inference on the group of datasets we consider shows that our method successfully manages to achieve a good balance between accuracy and speed. The targeted ratio was 80% of quantized steps, thus achieving around 60% of the original FP16 sampling time ($1.66\times$ speedup), as reported in Figure 6. The results on benchmark datasets reported in Table 1 show we achieve a strong performance on all of them, even surpassing the 16-bit baseline in some occurrences. This happens because the additional steps to mitigate the quantization error (i.e. extra remasking) also help adapt to difficult samples regardless of the model precision. The results show a slight under-performance on the HumanEval dataset, but the sample size ($n = 164$) is too small to make this gap conclusive.

Setting a similar ratio on a static mixed-precision baseline gives strong results, however it fails to outperform our method. This method suffers from two pitfalls: first, it is constrained to allocate the same ratio of quantized/fp steps to all samples, disregarding that some require more precision as they represent a harder task. Second, it is designed to allocate more compute to the earlier steps, assuming these are the ones most prone to errors. While this reflects the global trend of error evolution, we have demonstrated above that it is not true everywhere.

Finally, the PMPD approach fails to learn a significant sample distribution from the KV values signal. While it has shown promising results on AR inference, it remains to be tested in a DLM context. It is designed on the assumption that tokens always become progressively easier to decode, which is not the case in DLMs as we have seen earlier. Moreover, we can speculate that the KV values from a DLM are not as stable over the steps as in an autoregressive model, making the task more challenging than in the initial setting. The result is that the method overly relies on the fp model, thus yielding little acceleration.

5.3 Pairing acceleration methods

We continue by studying the integration of our method into already existing frameworks for DLM inference acceleration. While mixed precision inference remains unexplored, several methods have proposed speeding up the process by tweaking the unmasking policy. We consider here the scheduling approach of Fast-dLLM (Wu et al., 2026) which proposes varying the number of tokens un-

Table 2: Results for various combinations of Fast-dLLM and Mixed Precision quantization on the GSM8K dataset.

MP quant	Fast-dLLM	Score	INT steps %	Step count difference
✗	✗	74.8	0.0	0
✓	✗	74.8	77.6	+5.3 %
✗	✓	79.5	0.0	-30 %
✓	✓	78.8	69.8	-12 %

masked at each step depending on their uncertainty. Although other methods exist, we focus on this one given that they all share a similar principle. We disregard approaches that rely on block KV caches and speculative decoding since we consider them completely orthogonal to ours.

Classic DLM inference as presented above considers for unmasking at each step the tokens with the largest confidence p_k. Fast-dLLM proposes a dynamic approach by unmasking instead all tokens whose confidence exceeds a predefined threshold ($p_k > \tau$). The number of tokens unmasked at each step is then variable, with at least one token unmasked if none of them meet the threshold.

We can thus easily adapt our approach as it also relies on an uncertainty threshold to select the model used. By merging the two decision criteria, we pick based on the uncertainty both the model to use and the number of tokens to unmask. We also maintain our second threshold on the entropy velocity τ_β to directly fall back to the fp model in the regions where risk of error propagation is high.

We report the results of this hybrid method in Table 2 as well as the corresponding baselines. It shows we can successfully pair the two methods to benefit from both acceleration systems. The ratio of quantized steps is lower when using both of them as the steps removed are mainly the ones with low uncertainty, hence leaving a higher proportion of uncertain steps. Conversely, the number of steps removed is smaller as quantization degrades confidence.

Finally, we show in Section A.2 that it is also possible to use multiple quantization bitwidths to enable further acceleration. Our experiments indicate adding intermediate quantization levels can improve results without further increasing FP model calls.

5.4 Ablation studies

We conduct various experiments to assess the reliability of our method. We report in Table 4 the results on the GSM8K dataset when using only one of the two correction thresholds. It appears none of them single-handedly surpasses the proposed method. The fixed τ_ϵ gives a decent result but it is somewhat biased as we set it to the optimal value obtained from calibration data.

Furthermore, we show in Table 3 that our results are consistent across quantization methods and models. We study in Figure 10 the results for different values of the threshold τ_β and τ_ϵ. It shows both values have an effect on the acceleration/quality balance, with higher acceleration leading to worse quality, and conversely.

6 Conclusion

In this article, we demonstrated that it is possible to apply mixed precision to DLMs. To achieve this, we derived a novel model for error prediction based on the uncertainty and proposed using entropy velocity as a signal of the risk of error propagation. This allowed us to create a robust method for mixed precision inference, which we have shown can successfully improve the performance of a quantized model with minimal impact on the acceleration it provides. Across our experiments, we have verified it enables a significant improvement in inference speed without any notable decrease of the generation quality.

Limitations: As with all mixed precision inference work, this solution relies on a quantization accelerated model, which is not always easy to find, especially in DLMs. We hope future work will increase the number of available models and further facilitate their simultaneous use.

AI USAGE DISCLOSURE

Herein, we confirm none of the content of this article, including text, figures and bibliography, is the product of a generative AI tool. Note that we acknowledge some tools could have been used during the research process, such as to help create the associated codebase. The authors have duly ensured that the result meets their standards and take full responsibility for it.

REFERENCES

Yuvanesh Anand, Zach Nussbaum, Brandon Duderstadt, Benjamin Schmidt, and Andriy Mulyar. GPT4All: Training an assistant-style chatbot with large scale data distillation from GPT-3.5-turbo, 2023. URL https://github.com/nomic-ai/gpt4all.

Saleh Ashkboos, Amirkeivan Mohtashami, Maximilian L. Croci, Bo Li, Pashmina Cameron, Martin Jaggi, Dan Alistarh, Torsten Hoefler, and James Hensman. QuaRot: Outlier-Free 4-Bit Inference in Rotated LLMs. In *Advances in Neural Information Processing Systems*, volume 37, pp. 100213–100240. Curran Associates, Inc., 2024. doi: 10.52202/079017-3180. URL https://proceedings.neurips.cc/paper_files/paper/2024/file/b5b939436789f76f08b9d0da5e81af7c-Paper-Conference.pdf.

Jacob Austin, Daniel D. Johnson, Jonathan Ho, Daniel Tarlow, and Rianne van den Berg. Structured Denoising Diffusion Models in Discrete State-Spaces. In *Advances in Neural Information Processing Systems*, November 2021a. URL https://openreview.net/forum?id=h7-XixPCAL.

Jacob Austin, Augustus Odena, Maxwell Nye, Maarten Bosma, Henryk Michalewski, David Dohan, Ellen Jiang, Carrie Cai, Michael Terry, Quoc Le, and others. Program synthesis with large language models. *arXiv preprint arXiv:2108.07732*, 2021b.

Hao Mark Chen, Fuwen Tan, Alexandros Kouris, Royson Lee, Hongxiang Fan, and Stylianos Venieris. Progressive Mixed-Precision Decoding for Efficient LLM Inference. In *The Thirteenth International Conference on Learning Representations*, October 2024. URL https://openreview.net/forum?id=OVxmpus9NA.

Mark Chen, Jerry Tworek, Heewoo Jun, Qiming Yuan, Henrique Ponde de Oliveira Pinto, Jared Kaplan, Harri Edwards, Yuri Burda, Nicholas Joseph, Greg Brockman, Alex Ray, Raul Puri, Gretchen Krueger, Michael Petrov, Heidy Khlaaf, Girish Sastry, Pamela Mishkin, Brooke Chan, Scott Gray, Nick Ryder, Mikhail Pavlov, Alethea Power, Lukasz Kaiser, Mohammad Bavarian, Clemens Winter, Philippe Tillet, Felipe Petroski Such, Dave Cummings, Matthias Plappert, Fotios Chantzis, Elizabeth Barnes, Ariel Herbert-Voss, William Hebgen Guss, Alex Nichol, Alex Paino, Nikolas Tezak, Jie Tang, Igor Babuschkin, Suchir Balaji, Shantanu Jain, William Saunders, Christopher Hesse, Andrew N. Carr, Jan Leike, Josh Achiam, Vedant Misra, Evan Morikawa, Alec Radford, Matthew Knight, Miles Brundage, Mira Murati, Katie Mayer, Peter Welinder, Bob McGrew, Dario Amodei, Sam McCandlish, Ilya Sutskever, and Wojciech Zaremba. Evaluating large language models trained on code, 2021. arXiv: 2107.03374 [cs.LG].

Jacob K. Christopher, Brian R. Bartoldson, Tal Ben-Nun, Michael Cardei, Bhavya Kailkhura, and Ferdinando Fioretto. Speculative Diffusion Decoding: Accelerating Language Generation through Diffusion. In Luis Chiruzzo, Alan Ritter, and Lu Wang (eds.), *Proceedings of the 2025 Conference of the Nations of the Americas Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*, pp. 12042–12059, Albuquerque, New Mexico, April 2025. Association for Computational Linguistics. ISBN 979-8-89176-189-6. doi: 10.18653/v1/2025.naacl-long.601. URL https://aclanthology.org/2025.naacl-long.601/.

Karl Cobbe, Vineet Kosaraju, Mohammad Bavarian, Mark Chen, Heewoo Jun, Lukasz Kaiser, Matthias Plappert, Jerry Tworek, Jacob Hilton, Reiichiro Nakano, Christopher Hesse, and John Schulman. Training verifiers to solve math word problems. *arXiv preprint arXiv:2110.14168*, 2021.

Leo Gao, Jonathan Tow, Baber Abbasi, Stella Biderman, Sid Black, Anthony DiPofi, Charles Foster, Laurence Golding, Jeffrey Hsu, Alain Le Noac'h, Haonan Li, Kyle McDonell, Niklas Muennighoff, Chris Ociepa, Jason Phang, Laria Reynolds, Hailey Schoelkopf, Aviya Skowron, Lintang Sutawika, Eric Tang, Anish Thite, Ben Wang, Kevin Wang, and Andy Zou. A framework for few-shot language model evaluation, December 2023. URL https://zenodo.org/records/10256836. Version number: v0.4.0.

Haoyu He, Katrin Renz, Yong Cao, and Andreas Geiger. MDPO: Overcoming the Training-Inference Divide of Masked Diffusion Language Models, 2025. URL https://arxiv.org/abs/2508.13148. _eprint: 2508.13148.

Zhengfu He, Tianxiang Sun, Qiong Tang, Kuanning Wang, Xuanjing Huang, and Xipeng Qiu. DiffusionBERT: Improving generative masked language models with diffusion models. In Anna Rogers, Jordan Boyd-Graber, and Naoaki Okazaki (eds.), *Proceedings of the 61st annual meeting of the association for computational linguistics (volume 1: Long papers)*, pp. 4521–4534, Toronto, Canada, July 2023. Association for Computational Linguistics. doi: 10.18653/v1/2023.acl-long.248. URL https://aclanthology.org/2023.acl-long.248/.

Dan Hendrycks, Collin Burns, Saurav Kadavath, Akul Arora, Steven Basart, Eric Tang, Dawn Song, and Jacob Steinhardt. Measuring Mathematical Problem Solving With the MATH Dataset. In *Thirty-fifth Conference on Neural Information Processing Systems Datasets and Benchmarks Track (Round 2)*, August 2021. URL https://openreview.net/forum?id=7Bywt2mQsCe.

Emiel Hoogeboom, Didrik Nielsen, Priyank Jaini, Patrick Forré, and Max Welling. Argmax Flows and Multinomial Diffusion: Learning Categorical Distributions. In *Advances in Neural Information Processing Systems*, November 2021. URL https://openreview.net/forum?id=6nbpPqUCIi7.

Yafan Huang, Sheng Di, and Guanpeng Li. Not All Errors Are Equal: A Systematic Study of Error Propagation in Large Language Model Inference. In *Proceedings of the 40th ACM International Conference on Supercomputing*, ICS '26, pp. 972–985, New York, NY, USA, 2026. Association for Computing Machinery. ISBN 979-8-4007-2522-7. doi: 10.1145/3797905.3800513. URL https://dl.acm.org/doi/10.1145/3797905.3800513.

Siva Rajesh Kasa, Yasong Dai, Sumit Negi, and Hongdong Li. Fast-dLLM++: Fréchet Profile Decoding for Faster Diffusion LLM Inference, 2026. URL https://arxiv.org/abs/2606.02955. _eprint: 2606.02955.

Sangwoo Kwon, Seong Hoon Seo, Jae W. Lee, and Yeonhong Park. DP-LLM: Runtime Model Adaptation with Dynamic Layer-wise Precision Assignment. In *Advances in Neural Information Processing Systems*, volume 38, pp. 144921–144945, April 2026. URL https://proceedings.neurips.cc/paper_files/paper/2025/hash/d56b84c063265da949fe0feb815dcce8-Abstract-Conference.html.

Seunghoon Lee, Jeongwoo Choi, Byunggwan Son, JaeHyeon Moon, Jeimin Jeon, and Bumsub Ham. AccuQuant: Simulating multiple denoising steps for quantizing diffusion models. In D. Belgrave, C. Zhang, H. Lin, R. Pascanu, P. Koniusz, M. Ghassemi, and N. Chen (eds.), *Advances in neural information processing systems*, volume 38, Main Conference, pp. 110108–110143. Curran Associates, Inc., 2025. doi: 10.52202/085713-3679. URL https://proceedings.neurips.cc/paper_files/paper/2025/file/9f8265edac41c4f8b4445b0d0cb83b80-Paper-Conference.pdf.

Basile Lewandowski, Simon Kurz, Aditya Shankar, Robert Birke, Jian-Jia Chen, and Lydia Y. Chen. TMPDiff: Temporal mixed-precision for diffusion models, 2026. URL https://arxiv.org/abs/2603.14062. arXiv: 2603.14062 [cs.CV].

Guanghao Li, Zhihui Fu, Min Fang, Qibin Zhao, Ming Tang, Chun Yuan, and Jun Wang. DiffuSpec: Unlocking Diffusion Language Models for Speculative Decoding, 2025a. URL https://arxiv.org/abs/2510.02358. _eprint: 2510.02358.

Pengxiang Li, Yefan Zhou, Dilxat Muhtar, Lu Yin, Shilin Yan, Li Shen, Yi Liang, Soroush Vosoughi, and Shiwei Liu. Diffusion Language Model Knows the Answer Before It Decodes. In *The Fourteenth International Conference on Learning Representations*, October 2025b. URL https://openreview.net/forum?id=g88nt4ieTG.

Xiang Lisa Li, John Thickstun, Ishaan Gulrajani, Percy Liang, and Tatsunori Hashimoto. Diffusion-LM Improves Controllable Text Generation. In *Advances in Neural Information Processing Systems*, October 2022. URL https://openreview.net/forum?id=3s9IrEsjLyk.

Yuanzhe Li, Jianing Deng, Jingtong Hu, Tianlong Chen, Song Wang, and Huanrui Yang. Dynamic Mixed-Precision Routing for Efficient Multi-step LLM Interaction, May 2026a. URL http://arxiv.org/abs/2602.02711. arXiv:2602.02711 [cs.AI].

Zekai Li, Ji Liu, Yiqing Huang, Ziqiong Liu, Dong Li, and Emad Barsoum. Efficient Diffusion LLMs via Temporal-Spatial Parallel Decoding and Confidence Extrapolation, 2026b. URL https://arxiv.org/abs/2605.30753. Version Number: 1.

Chin-Yew Lin. Rouge: A package for automatic evaluation of summaries. In *Text summarization branches out*, pp. 74–81, 2004.

Haokun Lin, Haobo Xu, Yichen Wu, Jingzhi Cui, Yingtao Zhang, Linzhan Mou, Linqi Song, Zhenan Sun, and Ying Wei. DuQuant: Distributing Outliers via Dual Transformation Makes Stronger Quantized LLMs. In *Advances in Neural Information Processing Systems*, volume 37, pp. 87766–87800. Curran Associates, Inc., 2024. doi: 10.52202/079017-2786. URL https://proceedings.neurips.cc/paper_files/paper/2024/hash/9febda1c8344cc5f2d51713964864e93-Abstract-Conference.html.

Haokun Lin, Haobo Xu, Yichen Wu, Ziyu Guo, Renrui Zhang, Zhichao Lu, Ying Wei, Qingfu Zhang, and Zhenan Sun. Quantization Meets dLLMs: A Systematic Study of Post-training Quantization for Diffusion LLMs, 2025. URL https://arxiv.org/abs/2508.14896v3.

Fangxin Liu, Zongwu Wang, Jinhong Xia, Junping Zhao, Shouren Zhao, Jinjin Li, Jian Liu, Li Jiang, and Haibing Guan. FlexQuant: A Flexible and Efficient Dynamic Precision Switching Framework for LLM Quantization. In Christos Christodoulopoulos, Tanmoy Chakraborty, Carolyn Rose, and Violet Peng (eds.), *Findings of the Association for Computational Linguistics: EMNLP 2025*, pp. 4152–4161, Suzhou, China, November 2025. Association for Computational Linguistics. ISBN 979-8-89176-335-7. doi: 10.18653/v1/2025.findings-emnlp.221. URL https://aclanthology.org/2025.findings-emnlp.221/.

Tianyu Liu, Yun Li, Qitan Lv, Kai Liu, Jianchen Zhu, Winston Hu, and Xiao Sun. PEARL: Parallel Speculative Decoding with Adaptive Draft Length. In *The Thirteenth International Conference on Learning Representations*, October 2024. URL https://openreview.net/forum?id=QOXrVMiHGK.

Aaron Lou, Chenlin Meng, and Stefano Ermon. Discrete diffusion modeling by estimating the ratios of the data distribution. In *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *ICML'24*, pp. 32819–32848, Vienna, Austria, 2024. JMLR.org.

Shen Nie, Fengqi Zhu, Zebin You, Xiaolu Zhang, Jingyang Ou, Jun Hu, Jun Zhou, Yankai Lin, Ji-Rong Wen, and Chongxuan Li. Large Language Diffusion Models. In *The Thirty-ninth Annual Conference on Neural Information Processing Systems*, October 2025. URL https://openreview.net/forum?id=KnqiC0znVF.

Mariam Rakka, Marios Fournarakis, Olga Krestinskaya, Jinane Bazzi, Khaled N. Salama, Fadi Kurdahi, Ahmed M. Eltawil, and Mohammed E. Fouda. Mixed-Precision Quantization for Language Models: Techniques and Prospects, 2025. URL https://arxiv.org/abs/2510.16805v1.

Subham Sekhar Sahoo, Marianne Arriola, Yair Schiff, Aaron Gokaslan, Edgar Mariano Marroquin, Justin T. Chiu, Alexander M. Rush, and Volodymyr Kuleshov. Simple and Effective Masked Diffusion Language Models. In *The Thirty-eighth Annual Conference on Neural Information Processing Systems*, November 2024. URL https://openreview.net/forum?id=L4uaAR4ArM&noteId=fwSYxq1dTI.

Jameson Sandler, Jacob K. Christopher, Thomas Hartvigsen, and Ferdinando Fioretto. SpecDiff-2: Scaling Diffusion Drafter Alignment For Faster Speculative Decoding, 2025. URL https://arxiv.org/abs/2511.00606. _eprint: 2511.00606.

Oliver Sieberling, Denis Kuznedelev, Eldar Kurtic, and Dan Alistarh. EvoPress: Accurate Dynamic Model Compression via Evolutionary Search. In *Forty-second International Conference on Machine Learning*, June 2025. URL https://openreview.net/forum?id=l7QzcZpjc5&noteId=zzq2Pj7rlO.

Mirac Suzgun, Nathan Scales, Nathanael Schärli, Sebastian Gehrmann, Yi Tay, Hyung Won Chung, Aakanksha Chowdhery, Quoc V Le, Ed H Chi, Denny Zhou, , and Jason Wei. Challenging BIG-bench tasks and whether chain-of-thought can solve them. *arXiv preprint arXiv:2210.09261*, 2022.

Chengyue Wu, Hao Zhang, Shuchen Xue, Zhijian Liu, Shizhe Diao, Ligeng Zhu, Ping Luo, Song Han, and Enze Xie. Fast-dLLM: Training-free Acceleration of Diffusion LLM by Enabling KV Cache and Parallel Decoding. In *The Fourteenth International Conference on Learning Representations*, 2026. URL https://openreview.net/forum?id=3Z3Is6hnOT.

Guangxuan Xiao, Ji Lin, Mickael Seznec, Hao Wu, Julien Demouth, and Song Han. Smoothquant: Accurate and efficient post-training quantization for large language models. In *International conference on machine learning*, pp. 38087–38099. PMLR, 2023.

Chen Xu, Zhixuan Chen, and Dawei Yang. DLLMQuant: A Post-Training Quantization Framework Tailored for Diffusion-Based Large Language Models. In *Forty-third International Conference on Machine Learning*, June 2026. URL https://openreview.net/forum?id=fl93PQfTT6.

Jiacheng Ye, Zhihui Xie, Lin Zheng, Jiahui Gao, Zirui Wu, Xin Jiang, Zhenguo Li, and Lingpeng Kong. Dream 7B: Diffusion large language models, 2025. URL https://arxiv.org/abs/2508.15487. arXiv: 2508.15487 [cs.CL].

Tianao Zhang, Zhiteng Li, Xianglong Yan, Haotong Qin, Yong Guo, and Yulun Zhang. Quant-dLLM: Post-Training Extreme Low-Bit Quantization for Diffusion Large Language Models, 2025. URL https://arxiv.org/abs/2510.03274. Version Number: 1.

Jeffrey Zhou, Tianjian Lu, Swaroop Mishra, Siddhartha Brahma, Sujoy Basu, Yi Luan, Denny Zhou, and Le Hou. Instruction-following evaluation for large language models. *arXiv preprint arXiv:2311.07911*, 2023.

Appendix

A Extended experimental results

A.1 Additional setup details

Datasets. Consistent with other existing works, we evaluate on the datasets GSM8K (Cobbe et al., 2021), MATH (Hendrycks et al., 2021), MBPP (Austin et al., 2021b), HumanEval (Chen et al., 2021), Big Bench Hard (Suzgun et al., 2022), Instruction-Following Evaluation (Zhou et al., 2023). We use the `lm-eval` harness (Gao et al., 2023) to run the evaluations.

Quantization. We use DuQuant as our main quantization method throughout our experiments. DuQuant (Lin et al., 2024) is a lightweight quantization method widely adopted for LLM inference. We use W4A4 quantization by default (4bit weights and activations). We also consider QuaRot (Ashkboos et al., 2024) and SmoothQuant (Xiao et al., 2023) to ensure results remain consistent across quantization methods. We infer the speedup by computing the BOPs and accounting for the unquantized bottleneck (classic roofline model). This gives us a $2.05\times$ speedup for DuQuant, consistent with the range of $1.95\times$ to $2.34\times$ originally reported. Details are reported below.

Inference Parameters. Unless specified otherwise, we use 128 steps, 256 tokens distributed in blocks of 16 steps (32 tokens), similar to previous works. It means there are two tokens unmasked at each step, which is more than in the original work but enables a notable acceleration with regard to AR models without significant quality degradation. We disregard KV-caching as it is totally orthogonal to our approach.

A.2 Multi-level mixed precision

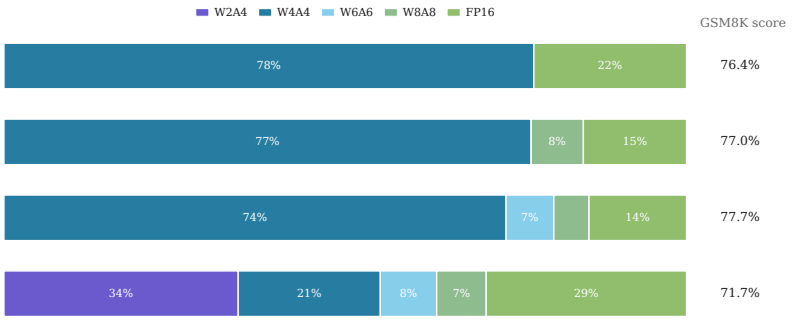


Figure 7: Distribution of steps across format mixes.

We also consider the result of using several model precisions (instead of the basic INT4/FP16 couple considered so far) and report the results in Figure 7. We implement the different formats as a "cascade of fallbacks," meaning that reaching a threshold on a format triggers remasking in the subsequent higher precision, and so on. The results of our experiment on a subset of GSM8K indicate adding intermediate quantization types successfully manages to improve the results without requiring more fp model calls. However, using W2A4 quantization produces significantly more quantization error, leading to a decrease in the score. Therefore, we can establish that the use of multi-level quantization is an interesting perspective, but it currently lacks efficient hardware support for non-standard quantization formats and dedicated low-bit methods.

A.3 Ablation results

We report in Table 3 the result of our method with various quantization methods. We use W4A4 for LLaDA and W4A8 for Dream, as the latter is more prone to errors. The *Default* baseline denotes using only DuQuant quantization for every steps. The scores appear to be consistent across the

Table 3: Results for different models and quantization methods on GSM8K.

	LLaDA Instruct	Dream
Default	69.8	44.6
Duquant	74.8	63.0
Quarot	74.5	70.9
Smoothquant	60.9	14.7

Table 4: Ablation results for GSM8K

	LLaDA Instruct
Default	69.8
τ_ϵ only	73.0
τ_β only	69.5
fixed τ_ϵ and τ_β	73.4
τ_ϵ and τ_β	74.8

different methods, with the exception of Smoothquant yielding worse results in both cases. This aligns with what was originally reported on AR models.

Additionally, we study in Table 4 the outcome when using only one of the two thresholds. Both fail to achieve a score as strong as the one from their combination. Using only the β threshold even gives worse results than the default baseline, we can speculate tuning it better (not only targeting propagation) would give better results.

B PROOFS

B.1 ASSUMPTIONS

Fix a position i and a denoising step t. Let $z \in \mathbb{R}^V$ denote the logits of the full-precision model at this position, $p = \mathrm{softmax}(z)$, and let $\hat{z} = z + \delta$ and $\hat{p} = \mathrm{softmax}(\hat{z})$ be the logits and distribution of the quantized model, where $\delta \in \mathbb{R}^V$ is the logit perturbation induced by quantization. Subscripts $v \in [V] := \{1, \dots, V\}$ index vocabulary entries; to lighten notation, we do not write explicitly the dependence of z, p, $\hat{p}$ and δ on i and t.

Assumption 1 (Bounded logit perturbation)**.** $\|\delta\|_\infty \le \Delta$*, where* Δ *is controlled by the quantization step size.*

For uniform b-bit affine quantization, each weight or activation is rounded with an error of at most half a quantization step, i.e. $O(2^{-b})$ relative to its clipping range. Since the network is locally Lipschitz, and hence Lipschitz on the bounded set of inputs under consideration, this error propagates to a bounded logit perturbation with $\Delta = O(2^{-b})$, where the hidden constant depends on the network.

Assumption 2 (Homogeneous, decorrelated noise)**.** *Conditionally on* z*, the perturbation decomposes as* $\delta_v = \mu + \eta_v$ *for all* $v \in [V]$*, where* $\mu \in \mathbb{R}$ *is common to all entries and* $(\eta_v)_{v=1}^V$ *are i.i.d. with mean* 0 *and variance* σ^2*. For a fixed quantization scheme,* σ^2 *does not depend on* i *or* t*.*

The shift μ plays no role, since the softmax is invariant to adding the same constant to all logits. We only use the first two moments of η below, so uncorrelated entries with a common variance are sufficient. Assumption 2 carries the classical additive model of quantization noise over to the logits, treating the residual quantization error as unstructured noise. This is a simplification: errors propagated through the output projection are correlated across similar tokens, and the approximation is least accurate for heavily structured quantization.

B.2 PROOF

For a distribution q on $[V]$, $\mathrm{Var}_q(\delta) := \sum_v q_v \delta_v^2 - \big(\sum_v q_v \delta_v\big)^2$ is the variance of δ_v when $v \sim q$. Additionally, let

$$\phi(x) := \frac{e^x - 1 - x}{x^2} = \int_0^1 (1-s)\, e^{sx}\, \mathrm{d}s \quad (x \neq 0), \qquad \phi(0) := \tfrac{1}{2},$$

a positive, increasing function with $\phi(x) = \frac{1}{2} + \frac{x}{6} + O(x^2)$ as $x \to 0$.

Lemma B.1 (KL divergence as a quadratic form in δ)**.** *Let* $F(p) = \mathrm{diag}(p) - pp^\top$ *be the Fisher information matrix of the categorical distribution* $\mathrm{softmax}(z)$ *with respect to the logits* z*, so that* $\mathrm{Var}_p(\delta) = \delta^\top F(p)\, \delta$*. Under assumption 1, for every realization of* δ*,*

$$\phi(-2\Delta)\, \mathrm{Var}_p(\delta) \;\le\; D_{\mathrm{KL}}(p \,\|\, \hat{p}) \;\le\; \phi(2\Delta)\, \mathrm{Var}_p(\delta).$$

In particular, $D_{\mathrm{KL}}(p \,\|\, \hat{p}) = \frac{1}{2}\,\delta^\top F(p)\,\delta\,\big(1 + O(\Delta)\big)$*, where the* $O(\Delta)$ *term is uniform in* p *and* V*.*

Proof. Let $A(\delta) := \log \sum_u e^{z_u+\delta_u}$ be the log-partition function, so that $\log \hat{p}_v = z_v + \delta_v - A(\delta)$ and $\log p_v = z_v - A(0)$. Since $\sum_v p_v = 1$,

$$D_{\mathrm{KL}}(p \,\|\, \hat{p}) = \sum_v p_v\big(A(\delta) - A(0) - \delta_v\big) = A(\delta) - A(0) - p^\top\delta.$$

For $s \in [0,1]$, let $q_s := \mathrm{softmax}(z + s\delta)$ and $g(s) := A(s\delta) - A(0) - s\,p^\top\delta$. The gradient and Hessian of A are the standard softmax moments, $\nabla A(s\delta) = q_s$ and $\nabla^2 A(s\delta) = F(q_s)$. Thus, $g(0) = g'(0) = 0$ and $g''(s) = \delta^\top F(q_s)\,\delta = \mathrm{Var}_{q_s}(\delta)$, the last equality following by direct expansion.

Taylor's formula with integral remainder therefore yields the exact identity

$$D_{\mathrm{KL}}(p \,\|\, \hat{p}) = g(1) = \int_0^1 (1-s)\,\mathrm{Var}_{q_s}(\delta)\,\mathrm{d}s.$$

It remains to compare $\mathrm{Var}_{q_s}(\delta)$ with $\mathrm{Var}_p(\delta)$. By assumption 1, all entries of δ lie in an interval of length 2Δ, so the likelihood ratio $q_s(v)/p_v = e^{s\delta_v}/\sum_u p_u e^{s\delta_u}$ lies in $[e^{-2s\Delta}, e^{2s\Delta}]$ for every v. Since a variance is the smallest mean squared deviation from a constant, i.e. $\mathrm{Var}_q(\delta) = \min_{c\in\mathbb{R}} \sum_v q_v(\delta_v - c)^2$, it follows that

$$\mathrm{Var}_{q_s}(\delta) \;\le\; \sum_v q_s(v)\,\big(\delta_v - p^\top\delta\big)^2 \;\le\; e^{2s\Delta}\,\mathrm{Var}_p(\delta),$$

and, exchanging the roles of p and q_s, that $\mathrm{Var}_p(\delta) \le e^{2s\Delta}\,\mathrm{Var}_{q_s}(\delta)$. Substituting these two bounds into the identity above and using $\int_0^1 (1-s)\,e^{\pm 2s\Delta}\,\mathrm{d}s = \phi(\pm 2\Delta)$ proves the two inequalities. The final statement follows from $\phi(\pm 2\Delta) = \frac{1}{2} \pm \frac{\Delta}{3} + O(\Delta^2)$. □

Lemma B.2 (Expected variance is proportional to the Gini–Simpson index)**.** *Under assumption 2,*

$$\mathbb{E}_\eta\big[\mathrm{Var}_p(\delta)\big] = \sigma^2\big(1 - \|p\|_2^2\big).$$

Proof. Since Var_p is invariant to adding a constant, $\mathrm{Var}_p(\delta) = \mathrm{Var}_p(\eta)$. As p is fixed given z and $\mathbb{E}_\eta[\eta_v^2] = \sigma^2$ for every v,

$$\mathbb{E}_\eta\Big[\sum_v p_v\eta_v^2\Big] = \sum_v p_v\,\mathbb{E}_\eta[\eta_v^2] = \sigma^2\sum_v p_v = \sigma^2.$$

Moreover, since the η_v are centered and uncorrelated, the cross terms vanish and

$$\mathbb{E}_\eta\Big[\big(\textstyle\sum_v p_v\eta_v\big)^2\Big] = \textstyle\sum_v p_v^2\,\mathbb{E}_\eta[\eta_v^2] = \sigma^2\|p\|_2^2.$$

Subtracting the second identity from the first gives the claim. □

Lemma B.3 (Sandwich bound: Gini–Simpson index vs. top-1 uncertainty)**.** *Let* $k \in \arg\max_{v\in[V]} p_v$ *and* $\epsilon_i := 1 - p_k$*. Then* $p_k^2 \le \|p\|_2^2 \le p_k$*, and consequently*

$$\epsilon_i \;\le\; 1 - \|p\|_2^2 \;\le\; \epsilon_i(1 + p_k) \;\le\; 2\epsilon_i.$$

Proof. The lower bound is immediate, since $\|p\|_2^2 = p_k^2 + \sum_{v\neq k} p_v^2 \ge p_k^2$. For the upper bound, every $v \neq k$ satisfies $p_v \le p_k$, hence $p_v^2 \le p_k\,p_v$. Summing over $v \neq k$ gives

$$\sum_{v\neq k} p_v^2 \;\le\; p_k \sum_{v\neq k} p_v \;=\; p_k(1 - p_k),$$

so that $\|p\|_2^2 \le p_k^2 + p_k(1-p_k) = p_k$. Subtracting both bounds from 1 gives:

$1 - p_k \le 1 - \|p\|_2^2 \le 1 - p_k^2 = (1-p_k)(1+p_k)$, which is the second chain since $p_k \le 1$. □

We can now prove theorem 3.1: under assumptions 1 and 2, $\mathbb{E}_\eta[e_i] = \Theta(\epsilon_i)$, where the constants hidden in Θ depend only on σ and Δ.

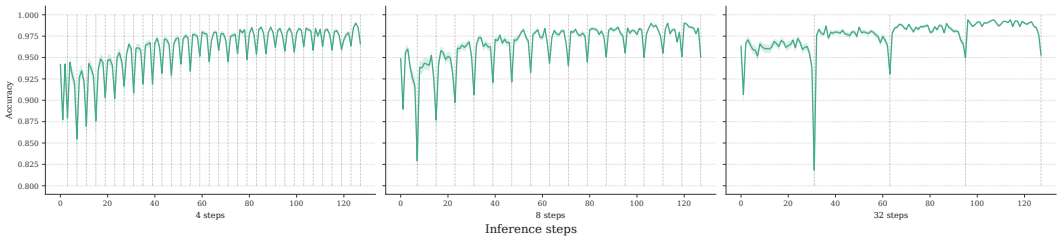


Figure 8: Accuracy w.r.t. the FP16 baseline over the inference process for different sizes of blocks.

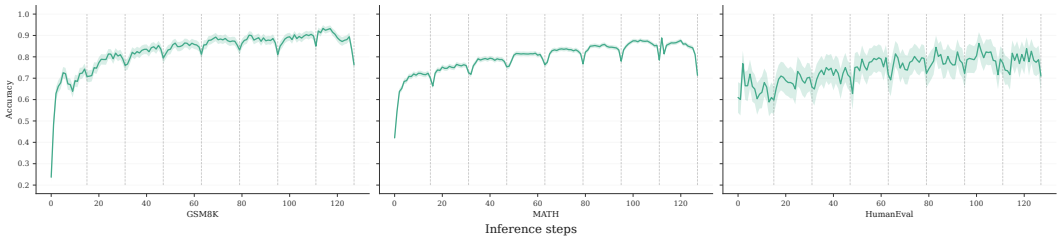


Figure 9: Accuracy w.r.t. the FP16 baseline over the inference process for different datasets.

Proof of theorem 3.1. By lemma B.1, for every realization of δ,

$$\phi(-2\Delta)\,\mathrm{Var}_p(\delta) \;\le\; e_i \;\le\; \phi(2\Delta)\,\mathrm{Var}_p(\delta).$$

Both bounds are deterministic multiples of $\mathrm{Var}_p(\delta)$, so no remainder term needs to be controlled when taking expectations; by lemma B.2,

$$\phi(-2\Delta)\,\sigma^2\big(1-\|p\|_2^2\big) \;\le\; \mathbb{E}_\eta[e_i] \;\le\; \phi(2\Delta)\,\sigma^2\big(1-\|p\|_2^2\big).$$

Finally, $\epsilon_i \le 1-\|p\|_2^2 \le 2\epsilon_i$ by lemma B.3, hence

$$\phi(-2\Delta)\,\sigma^2\,\epsilon_i \;\le\; \mathbb{E}_\eta[e_i] \;\le\; 2\,\phi(2\Delta)\,\sigma^2\,\epsilon_i.$$

Both constants are positive and depend only on σ and Δ, not on p, i, t or V; therefore $\mathbb{E}_\eta[e_i] = \Theta(\epsilon_i)$. For small Δ, the constants are $\frac{\sigma^2}{2}\big(1+O(\Delta)\big)$ and $\sigma^2\big(1+O(\Delta)\big)$. □

C Additional experiments

We reproduce our analysis on quantization error evolution in different contexts. As reported in Figure 8, changing the block size of the unmasking sampler reproduces the last-step error spike we identified previously. The loss of accuracy is proportional to the size of the block. This pattern is also consistent across datasets, as demonstrated in Figure 9. The trend of increasing accuracy through the unmasking is observable across all experiments.

In addition to the scores coming from the different datasets, we use the Rouge measure (Lin, 2004) to assess how similar the output is to the fp baseline. When comparing the generated sequence with the default one, we observe all methods get a strong similarity, as reported in Table 5.

We also report in Figure 10 the different rouge values with regard to the speedup provided for different values of the threshold τ_ϵ and τ_β. As expected, the results indicate that stricter thresholds lead to stronger similarity and reduced acceleration. We use $\tau_\beta = 0.1$ in the rest of our experiments as it offers a good error correction without sacrificing too much of the acceleration.

We analyze in Figure 11 the result of the entropy evolution when downgrading different blocks to a quantized model. Similar to what we report in Figure 4b, it appears that the errors are more likely in contexts where the entropy decrease is less marked. This pattern is absent in the last experiment (quantizing the penultimate block) but the errors in this case are very limited, with the majority of generated samples (70%) not presenting any error.

We consider several strategies for the "Static mix" baseline and report the results in Figure 12. The different options are:

Table 5: Rouge score across methods and datasets. Reference is the FP16 output.

	GSM8K	MATH	MBPP	Human
W4A4	58.2	53.6	69.7	66.6
PMPD	94.4	90.3	98.2	97.9
static	84.7	74.5	94.7	92.0
our	83.7	73.9	93.1	89.7

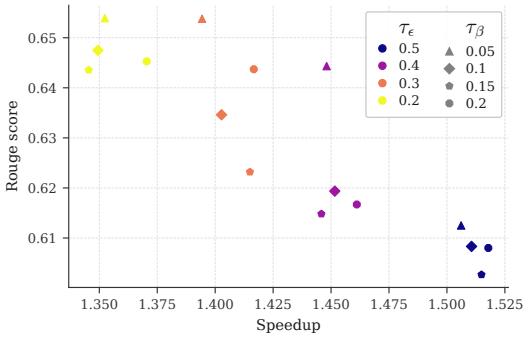


Figure 10: Rouge vs. speedup comparison for different fixed threshold values.

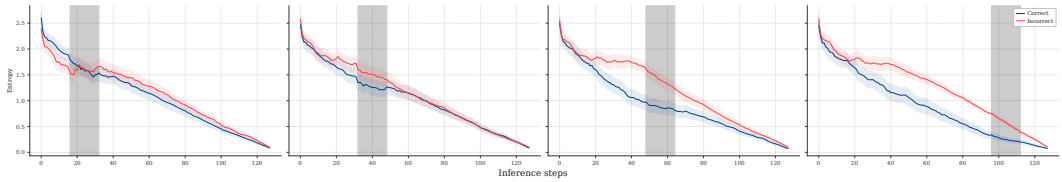


Figure 11: Entropy evolution for correct/incorrect samples for different quantization regions.

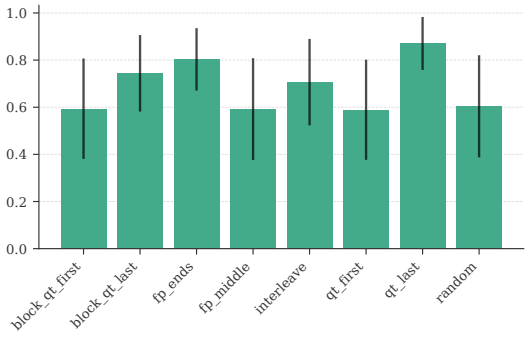


Figure 12: Rouge score result for the GSM8K dataset with 75% of quantized steps. Reference is the FP16 output.

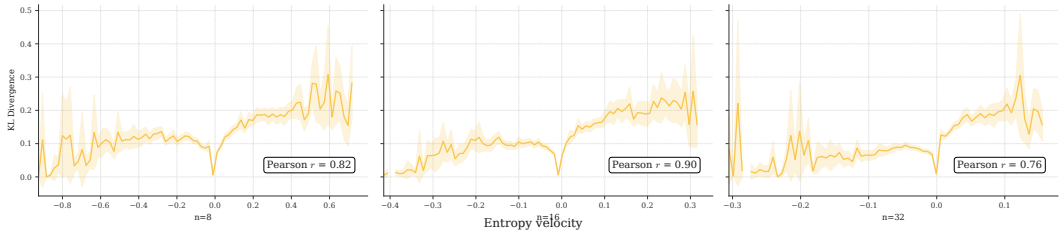


Figure 13: KL divergence vs. entropy velocity for various window sizes for the determination of β

- `interleave`: alternating FP and INT on an evenly spread pattern
- `qt_last`: FP warm-up, then INT until the end
- `qt_first`: INT start, then FP for the last steps
- `fp_middle`: FP only in the middle of the trajectory
- `fp_ends`: FP at the very start and the very end
- `block_qt_last`: same as `qt_last`, within every block
- `block_qt_first`: same as `qt_first`, within every block
- `random`: N steps drawn at random

The results show the `qt_last` strategy gives the best result on average, and we implement it as our static baseline. However, variance is high and the performance gap between strategies is narrow, suggesting that the strategy choice does not have a dramatic impact in practice.

C.1 SPEEDUP ESTIMATION

We estimate the end-to-end speedup from two complementary perspectives: (1) the reduction in bit operations (BOPs) achieved by quantization, and (2) a roofline/Amdahl analysis of the denoising step.

Step 1: BOPs per denoising step. For each matrix multiplication, we estimate the computational cost as

$$\text{BOPs} = \text{MACs} \times (\text{weight bits}) \times (\text{activation bits})$$

The resulting number of BOPs per denoising step is:

Precision	BOPs/step	vs. BF16
BF16	1743 TBOPs	1.00×
W4A4 (DuQuant)	223 TBOPs	7.80×

Table 6: Estimated bit operations per denoising step.

The LM head remains at 16-bit precision and accounts for approximately 55% of the BOPs in the quantized configuration. Hence, although W4A4 substantially reduces cost of the quantizable matrix multiplications, the full denoising step is constrained by the remaining higher-precision computation.

However, BOPs are not a proxy for latency: they scale with the product of weight and activation, whereas tensor-core throughput scales roughly linearly with bitwidth. Moving from 16bit to 4bit operands therefore divides BOPs by 16 but raises peak throughput by only about 4 times. We use BOPs as a measure of arithmetic cost and estimate latency separately below.

Step 2: Roofline analysis and Amdahl's law. The BF16 denoising step takes 15.79 ms on a modern GPU (GH200). Of this time, approximately 85.4% is spent in quantizable matrix multiplications, 6.0% in the LM head, and 8.6% in operations other than matrix multiplication (normalization, residual connections, rotary embeddings).

The arithmetic intensity of the matrix multiplications is approximately 616 to 721 FLOP/byte, which is well above the estimated machine balance of 247 FLOP/byte. These operations are therefore compute-bound rather than memory-bound, so the primary source of acceleration is the higher throughput of low-precision Tensor Cores.

Assuming an ideal $4\times$ throughput improvement for 4bit Tensor Cores (as targeted by Nvidia FP4), the quantizable portion would be reduced from approximately 13.4 ms to 3.4 ms. This gives an ideal step time of approximately 5.76 ms, corresponding to a $2.74\times$ speedup.

If all quantizable matrix multiplications were accelerated by exactly $4\times$, Amdahl's law would bound the speedup by:

$$S_{\text{ideal}} = \frac{1}{\frac{0.854}{4} + 0.146} = 2.78\times,$$

where the 0.146 term represents the LM head and other non-quantized/non-matmul operations. The slightly lower $2.74\times$ estimate accounts for the fact that the two attention matrix multiplications achieve only approximately 2.4-$3.0\times$ acceleration rather than a full $4\times$.

Accounting for practical overheads. In practice, the effective Tensor Core speedup is closer to $2.5\times$ than the ideal $4\times$, due to activation quantization and other implementation inefficiencies. Applying this more conservative arithmetic speedup to the quantizable fraction gives

$$S_{\text{real}} = \frac{1}{\frac{0.854}{2.5} + 0.146} = 2.05\times.$$

We therefore use $2.05\times$ as the expected end-to-end speedup for W4A4 inference.