# CONTINUOUS SPACE DRIFTING FOR DISCRETE DIFFUSION LANGUAGE MODELS

**Anonymous authors**
Paper under double-blind review

## ABSTRACT

Masked diffusion language models have become capable generators, motivating objectives that complement token reconstruction with sequence-level feedback. We present LD2LM (Large Drifting Diffusion Language Model), which adapts continuous feature-space supervision to an 8B discrete block-diffusion model. Building on TokenDrift's differentiable soft-token interface, LD2LM retains denoising cross-entropy (CE) and adds multiscale feature views across encoder depths and sequence regions. We also study how this supervision interacts with token- and feature-level KL constraints. Although TokenDrift reports benefits from feature-space Drifting, directly transferring its Drift-only recipe to DreamReasoner8B performs substantially worse than CE. We hypothesize that Drift-only feedback may not transfer directly to scaled discrete block-diffusion models. Drawing on scaling experience with richer representations, LD2LM combines continuous multiscale feature feedback with token-level CE. In a matched training-stage comparison, the Full schedule raises HumanEval from 74.39% to 75.00%, MBPP from 60.40% to 61.20%, and LiveCodeBench pass@1 from 44.84% to 49.26%; tokens per forward pass increase from 1.5490 to 1.5679, with the inference sampler unchanged. In a separate token-KL comparison, adding Drifting improves HumanEval from 73.78% to 75.61% and LiveCodeBench from 49.56% to 51.33%, while tying MBPP at 61.40%. These results show the promise of continuous multiscale feature feedback for adapting Drifting to scaled discrete block-diffusion models.

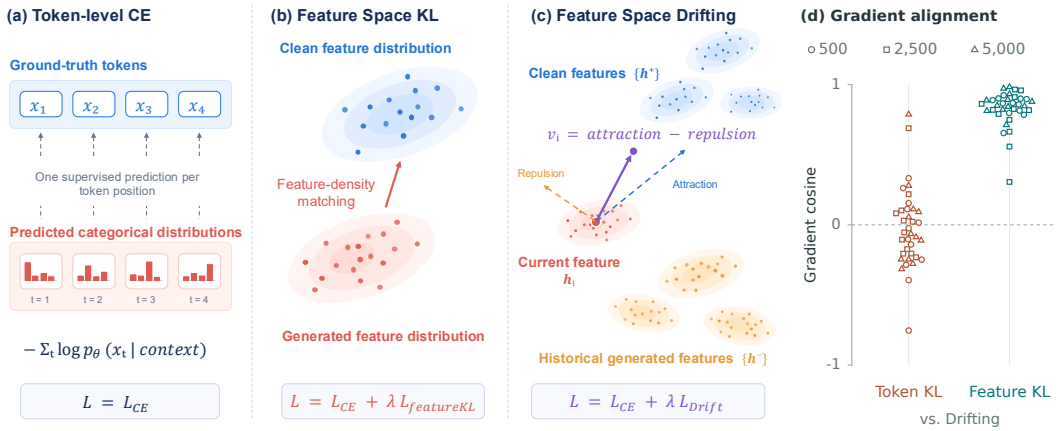


Figure 1: **Continuous feature-space feedback for a discrete denoiser.** (a) Token-level CE supervises discrete predictions. (b) Feature-space KL matches feature distributions. (c) Feature-space Drifting compares current predictions with clean and generated references, including historical features. (d) Gradient alignment supports this feature-space interpretation: Drifting has higher median cosine alignment with feature KL than with token KL at each measured training step. Circles, squares, and triangles denote 500, 2,500, and 5,000 training steps, respectively. LD2LM combines feature-space feedback with token-level CE.

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.

# 1 INTRODUCTION

Masked diffusion language models (MDLMs) have developed into capable large language models, combining contextual denoising with parallel token generation (Sahoo et al., 2024; Nie et al., 2026; Ye et al., 2025). Most denoising objectives supervise token predictions, even though the model conditions on sequence context, as in Figure 1(a). Recent methods improve token correction, training trajectories, and categorical distribution matching (Chen et al., 2026; Ren et al., 2026; Bie et al., 2026; Hoogeboom et al., 2026). Other approaches, such as CADD and CCDD, enrich token representations with continuous information during denoising (Zheng et al., 2026; Zhou et al., 2025).

These developments motivate us to examine not only the model's input representation, but also the space in which its predictions receive supervision. Can training benefit from explicit comparisons between sequence representations, beyond supervision of individual token predictions? Semantic coherence depends on relationships among tokens, motivating us to leverage their joint representation. We explore a complementary direction with the previous MDLMs: comparing predicted and clean sequences in a continuous feature space.

Generative Modeling via Drifting (Deng et al., 2026) offers a way to construct this feedback through attraction to data features and repulsion from generated features. TokenDrift (Oba et al., 2026) extends this idea to discrete diffusion language models through differentiable soft-token lifting, reporting improved generative perplexity in smaller-model unconditional text generation, as Figure 1(b)-(c) illustrates. Applying this feedback to a strong pretrained reasoning model is nontrivial. In our experiments, directly transferring its original Drift-only training recipe to DreamReasoner8B (Wu et al., 2026b) does not improve code generation (see Section 4.1 and Table 1). This motivates studying how feature-space feedback should complement the existing denoising objective of a capable 8B model.

We develop **LD2LM** (Large Drifting Diffusion Language Model) on DreamReasoner8B. LD2LM retains denoising cross entropy (CE) and uses Drifting to provide additional sequence-level supervision through TokenDrift's differentiable interface. We construct multiscale feature views across encoder depths with global and local sequence regions, allowing different aspects of a prediction to contribute to the feature loss (see Section 3). Our feature drifting is only deployed to training, leaving the inference sampler unchanged. We further investigate when to introduce Drifting and how it interacts with token- and feature-level KL constraints. Figure 1 contrasts these learning spaces; panel (d) shows that Drifting gradients align more closely with feature KL than with token KL, consistent with its feature-space objective, indicating the complementarity of our methods with existing token-level approaches.

Our experiments show how continuous feature feedback can be adapted to an 8B discrete block-diffusion model. TokenDrift's Drift-only recipe performs substantially worse than CE when transferred to our setting, motivating LD2LM: we retain token-level CE and add continuous multiscale sequence features, drawing on prior scaling experience with richer representations. In the matched training-stage study, Full improves HumanEval, MBPP, and LiveCodeBench by 0.61, 0.80, and 4.42 percentage points at 1.22% higher tokens per forward pass (TPF) (Figure 3). The objective-role comparison records 75.61%/61.20%/48.08% on HumanEval/MBPP/LiveCodeBench and 80.00%/60.00% on AIME24/AIME25 for CE plus Drifting (Table 1). Adding Drifting to CE with token KL improves HumanEval and LiveCodeBench by 1.83 and 1.77 points and ties MBPP (Table 2); feature-group and queue sweeps show gains on selected tasks (Table 3; Figure 5).

Our contributions are threefold:

- **Effective Drifting at 8B scale.** We develop LD2LM, which adapts feature-space Drifting to DreamReasoner8B and improves code generation through joint denoising and sequence-level supervision.
- **A study of the learning space.** We construct multiscale sequence features and investigate feature granularity, training stage, and KL constraints, characterizing feature-view choices and gains beyond token-level anchoring (Sections 4.2–4.3; Tables 2 and 3).
- **Improved accuracy and decoding efficiency.** Controlled experiments establish an accuracy–efficiency Pareto improvement (Figure 4) over CE continual training baseline on the tested backbone and protocol, without changing the inference sampler.

# 2 Related Work

## 2.1 Diffusion Language Models

Early continuous diffusion language models generate text through embedding or relaxed token spaces (Li et al., 2022; Gong et al., 2022; Han et al., 2023; Chen et al., 2023). Discrete diffusion instead defines corruption directly over finite states, as in D3PM (Austin et al., 2021a) and multinomial diffusion (Hoogeboom et al., 2021). Discrete diffusion is further developed through advances in training objectives and model adaptation (Lou et al., 2023; Sahoo et al., 2024; Shi et al., 2024; Gong et al., 2025). LLaDA (Nie et al., 2026) and Dream 7B (Ye et al., 2025) scale masked diffusion to large language models. Block diffusion (Arriola et al., 2025) combines sequential block generation with parallel denoising within each block. DreamReasoner8B (Wu et al., 2026b) extends this setting to long-chain-of-thought reasoning and achieves frontier performance.

For conditional masked diffusion, let $x_i = [c_i; y_i]$ contain a prompt and response. At corruption level $\rho$, a nonempty set $\mathcal{M}_i$ of response positions is replaced by mask symbols to form $\bar{x}_i$; all other tokens remain visible. The denoiser produces logits $\ell_{it} = f_\theta(\bar{x}_i)_t$ and probabilities $p_{it} = \mathrm{softmax}(\ell_{it})$. A normalized token reconstruction objective is

$$\mathcal{L}_{\mathrm{CE}}(\theta) = \mathbb{E}_{x_i,\rho,\mathcal{M}_i}\left[-\frac{1}{|\mathcal{M}_i|}\sum_{t\in\mathcal{M}_i}\log\left[\mathrm{softmax}(f_\theta(\bar{x}_i)_t)\right]_{x_{i,t}}\right]. \tag{1}$$

This token-indexed loss provides the denoising foundation for our study of complementary sequence-level supervision.

## 2.2 Learning Spaces for Masked Diffusion

MDLM training can be distinguished by the space in which supervision acts. Recent methods improve token refinement and denoising trajectories (Chen et al., 2026; Ren et al., 2026; Bie et al., 2026). D-MMD matches categorical distributions (Hoogeboom et al., 2026) through maximum mean discrepancy (Gretton et al., 2012). CADD (Zheng et al., 2026) and CCDD (Zhou et al., 2025) enrich discrete states with continuous representations, expanding the information available for reconstruction. Our focus is complementary: using relations between clean and predicted sequence features as a training signal for a discrete denoiser. The distinction concerns the supervision space, rather than whether the model has access to sequence context.

## 2.3 Drifting and Contrastive Representation Learning

DMD (Yin et al., 2024b) and DMD2 (Yin et al., 2024a) use distribution matching for image-generator distillation. Drifting (Deng et al., 2026) constructs an attraction–repulsion field from data and generated samples or features. TokenDrift (Oba et al., 2026) applies this principle to discrete language models through a soft-token interface that makes sequence features differentiable with respect to token predictions. We build on this foundation to study feature-space refinement of an 8B denoiser for conditional generation.

Representation learning objectives impose structure on feature spaces through predictive, geometric, and statistical constraints (van den Oord et al., 2018; Wang & Isola, 2020; Zbontar et al., 2021; Bardes et al., 2022). Instance-discrimination memory banks (Wu et al., 2018), Momentum Contrast (He et al., 2020), cross-batch memory (Wang et al., 2020), and large-batch SimCLR (Chen et al., 2020) provide related ways to compare examples beyond individual pairs. LD2LM instead uses a fixed representation space to supervise a generative model; representation learning is not its training objective.

# 3 LD2LM: Structured Feature Drifting for an 8B MDLM

LD2LM introduces continuous sequence-level feedback into the training of DreamReasoner8B. Building on TokenDrift (Oba et al., 2026), we retain the differentiable soft-token interface and combine feature-space Drifting with token reconstruction. We first review this interface, then explain the joint objective and its training-stage and KL controls. We next describe multiscale feature views

and historical reference queues, whose roles are tested separately in Section 4. A frozen copy of the starting checkpoint provides the feature encoder $\phi$; only the denoiser $f_\theta$ is updated. All additional components operate during training, leaving the inference sampler unchanged.

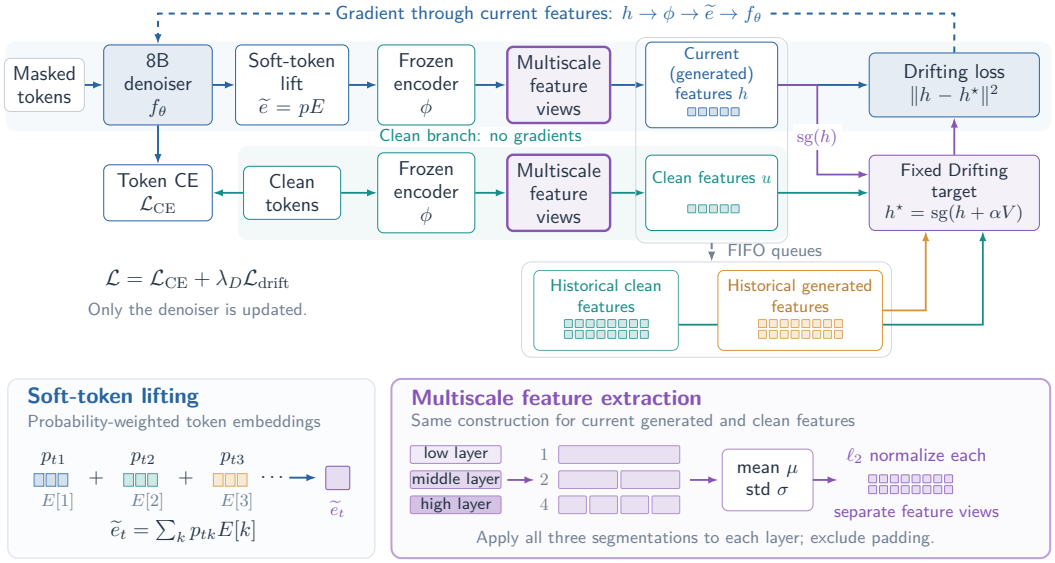


Figure 2: LD2LM training. Denoiser predictions are lifted to soft embeddings and encoded in a frozen feature space, following TokenDrift (Oba et al., 2026). LD2LM augments token-level CE with Drifting toward a stopped target constructed from current and historical clean and generated references. Our multiscale construction extracts separately normalized means and standard deviations across encoder depths and sequence segments. Gradients pass through the encoder inputs to update only the denoiser; detached current features enter historical queues after each update. Insets detail soft-token lifting and multiscale extraction; tile counts are schematic.

### 3.1 Preliminary: Differentiable Features

The logits $\ell_{it}$ from Section 2.1 lie in $\mathbb{R}^K$ for vocabulary size $K$. Let $E \in \mathbb{R}^{K \times d}$ be the frozen encoder's embedding table. Following the soft-token lift of TokenDrift, we map token probabilities $p_{it}$ to continuous inputs:

$$\widetilde{e}_{it} = \begin{cases} p_{it} E, & t \in \mathcal{M}_i, \\ E[\bar{x}_{it}], & t \notin \mathcal{M}_i. \end{cases} \tag{2}$$

Unlike argmax or token sampling, Equation 2 lets gradients pass from continuous features to the logits that predict discrete tokens. The encoder parameters and embeddings are frozen, but their inputs remain differentiable. The encoder is therefore a fixed feature space, not a learned critic.

For feature view $r$, the current differentiable feature and the fixed clean reference are

$$h_{ir} = \phi_r(\widetilde{e}_i), \qquad u_{ir} = \mathrm{sg}(\phi_r(E[x_i])) \,, \tag{3}$$

where sg stops gradients. Copies of $h_{ir}$ with gradients stopped form generated references; the current $h_{ir}$ remains differentiable.

For each valid view, clean features are positive references and generated features are negative references, drawn from the current batch and their respective history queues. Following the attraction–repulsion construction in Drifting and TokenDrift (Deng et al., 2026; Oba et al., 2026), $V_{ir}$ moves the live feature toward clean references and away from generated references. Appendix A, Equation 12, gives the exact field definition; Appendix A, Algorithm 2, specifies the weight, normalization, and masking steps. The field and its references are detached, while the current $h_{ir}$ remains differentiable.

The field defines a stopped target and a feature-matching loss:

$$h^{\star}_{ir} = \mathrm{sg}(h_{ir} + \alpha V_{ir}), \tag{4}$$

$$\mathcal{L}_{\mathrm{drift}} = \frac{1}{2B|\mathcal{R}|} \sum_{(i,r)\ \mathrm{valid}} \|h_{ir} - h^{\star}_{ir}\|_2^2. \tag{5}$$

Here $B$ is the global batch size, $\mathcal{R}$ is the set of feature views, and $\alpha$ is the field step size. Gradients through the live features reach the denoiser via Equation 2; the target and references are fixed during each update.

### 3.2 Feedback beyond token anchoring

Our initial application of TokenDrift's original Drift-only recipe to DreamReasoner8B did not improve code generation. Table 1 reports that comparison under the controlled-local Code protocol. This observation motivates retaining direct token supervision while adding feature feedback. A sequence representation need not preserve every token distinction required for exact code reconstruction; CE provides those token targets, while Drifting supplies a complementary comparison with clean and generated sequence features. We therefore use

$$\mathcal{L} = \mathcal{L}_{\mathrm{CE}} + \lambda_D \mathcal{L}_{\mathrm{drift}}. \tag{6}$$

Appendix A, Algorithm 1, specifies inverse-corruption weighting and response-length normalization for the implemented CE term; Equation 1 illustrates its reconstruction basis. TokenDrift already studies base-denoising loss combined with Drifting (Oba et al., 2026); our focus is its effective use and analysis in this 8B setting.

**Joint training and its timing.** We investigate when the additional feature signal is useful. For optimizer update $s$, let $\lambda_D(s) = \lambda_0 \mathbf{1}[s \in \mathcal{A}]$, where $\mathcal{A}$ is the set of Drifting-active updates. CE remains active throughout. Taking $\mathcal{A}$ to be all updates, the first half, the second half, or the empty set gives the four controls in Figure 3. This compares the timing and duration of feature feedback within the same architecture; the second-half setting first trains token reconstruction and then applies the joint objective.

**KL controls.** Drifting sets a target using clean and generated feature references. Token KL instead anchors masked-token distributions to the starting model. We test whether the former still contributes when the latter is present, using separate CE-plus-KL controls with and without Drifting (Table 2). Feature KL provides a second control in feature space. These controls probe the value of the training signal; neither KL term is required by the standard LD2LM objective. Unlike RL trust regions that constrain divergence from a rollout policy (Qi et al., 2026), our token KL anchors the starting model.

### 3.3 Multiscale feature views

TokenDrift forms a single normalized sequence representation by concatenating mean-pooled states from its encoder's last two layers (Oba et al., 2026). We instead keep multiple views separate, so that global and local regions each contribute their own feature-space loss. Local segments expose broad position information to the comparison; this is the design tested by the global-only ablation in Table 3.

The same frozen encoder processes soft-token and clean sequences, so matching views use the same layer and statistic. We summarize the valid prompt and response, excluding padding. Only masked response positions use probability-weighted embeddings; other positions use their known-token embeddings. For an encoder with $D$ zero-indexed layers, we select fixed low, middle, and high layers:

$$\mathcal{S}_{\mathrm{layer}} = \{\lfloor D/3 \rfloor - 1,\ \lfloor 2D/3 \rfloor - 1,\ D-1\}. \tag{7}$$

We do not learn which layers to use. Within each layer, we split valid tokens in sequence order into $n \in \{1, 2, 4\}$ adjacent, nonoverlapping segments. If $z_{it}$ marks a valid token, then for valid position $t$,

$$L_i = \sum_t z_{it}, \qquad k_i(t) = \sum_{s \le t} z_{is} - 1, \qquad b_i^{(n)}(t) = \left\lfloor \frac{n\, k_i(t)}{L_i} \right\rfloor. \tag{8}$$

Thus $n = 1$ gives one segment for the whole sequence; $n = 2$ and $n = 4$ keep broad position information. These are fixed segments, not moving windows or language units. Let $\mathcal{I}_{i,n,j}$ be the valid positions in segment $j$. For encoder states $H_{it}^{(\ell)}$, each selected layer and segment gives a mean and standard deviation for each feature:

$$\mu_{i\ell nj} = \frac{1}{|\mathcal{I}_{i,n,j}|} \sum_{t \in \mathcal{I}_{i,n,j}} H_{it}^{(\ell)}, \tag{9}$$

$$\sigma_{i\ell nj} = \left( \frac{1}{|\mathcal{I}_{i,n,j}|} \sum_{t \in \mathcal{I}_{i,n,j}} (H_{it}^{(\ell)} - \mu_{i\ell nj})^{\odot 2} + \epsilon \right)^{1/2}. \tag{10}$$

We L2-normalize each vector separately. Means and standard deviations remain separate views rather than being joined. Three layers, $1 + 2 + 4$ segments, and two statistics give $|\mathcal{R}| = 3(1 + 2 + 4)2 = 42$ possible views. Empty segments are invalid and are not stored as zero-valued references. All valid views have the same fixed loss weight. Later tests remove groups of views; they do not train a feature-selection module.

### 3.4 Reference banks and queue capacity

Following TokenDrift (Oba et al., 2026), we supplement the current batch with two first-in, first-out (FIFO) reference banks: one for clean features and one for generated features, both with gradients stopped. We maintain matching references for each feature view $r$:

$$\mathcal{P}_r = \mathcal{U}_{\mathrm{cur},r} \cup Q_r^+, \qquad \mathcal{N}_r = \mathrm{sg}(\mathcal{H}_{\mathrm{cur},r}) \cup Q_r^-, \tag{11}$$

where $\mathcal{U}_{\mathrm{cur},r}$ and $\mathcal{H}_{\mathrm{cur},r}$ contain the current global batch. Queue capacity $Q$ counts samples in each bank, not views or generation paths. The banks store features, validity masks, and sample IDs. They do not store full generation paths or gradients from earlier updates. We exclude current IDs from historical matches. We build the field from the existing banks and add the current features only after a successful optimizer update, so one batch is not counted twice.

The banks provide more reference examples, but older features may no longer match the current denoiser. A queue of size $Q$ holds roughly $Q/B$ updates of history at global batch size $B$. This is a history-length estimate, not a measured queue age. Queue capacity is a tested setting; the standard LD2LM rule gives every reference the same weight.

### 3.5 Conceptual connections to reinforcement learning

LD2LM constructs feedback jointly over a batch: row- and column-normalized affinities couple examples, attraction and repulsion have equal total weights at each anchor, and the field is rescaled by its batch root-mean-square length. This shares a design principle with critic-free RL with batch-normalized rewards, where relative positive and negative contributions shape the update (Wu et al., 2026a). Here, model-derived representations provide implicit comparative feedback against clean data and generated references, without an external scalar reward. The resulting $V_{ir}$ is a vector in feature space, rather than a scalar advantage multiplying a sampled policy score. LD2LM thus derives an implicit training signal from relations among clean and generated sequence features.

## 4 Experiments

We begin with a comparison in mathematics and code, then analyze the training objective, multiscale features, and queue capacity in the order introduced in Section 3.

### 4.1 Objective roles in mathematics and code

The Code benchmarks are HumanEval (HE) (Chen et al., 2021), Mostly Basic Programming Problems (MBPP) (Austin et al., 2021b), and LiveCodeBench (LCB) (Jain et al., 2025). The Math tables use GSM8K, MATH-500, AIME 2024 (AIME24), and AIME 2025 (AIME25). We use these short labels in the main table headers.

Table 1 compares CE and feature-space Drifting. On Code, LD2LM improves HumanEval over CE (75.61% versus 74.39%) and ties MBPP (61.20%), but lowers LiveCodeBench (48.08% versus 51.03%); Drifting alone scores 54.27%, 48.40%, and 10.91% on these tasks. On AIME 2024/2025 (Zhang & Math-AI Team, 2024; 2025), LD2LM reaches 80.00%/60.00% versus CE's 73.33%/56.67%. The historical Math runs differ in budget, updated parameters, and corruption schedule, so this comparison is descriptive.

Table 1: Code pass@1 and Math accuracy for objective-role controls. Code uses controlled-local scoring; Math uses strict Math-Verify 0.8.0 and historical, non-matched runs. Bold and underline mark the best and second-best task scores; TPF is tokens per forward.

| | Code pass@1 (%) | | | | Math accuracy (%) | | | | |
|---|---|---|---|---|---|---|---|---|---|
| Objective | HE | MBPP | LCB | TPF | GSM8K | MATH-500 | AIME24 | AIME25 | TPF |
| Starting checkpoint (E0) | 73.17 | <u>60.60</u> | **51.92** | 1.6227 | 92.19 | 92.20 | 56.67 | 53.33 | 2.1997 |
| CE continuation | <u>74.39</u> | **61.20** | <u>51.03</u> | 1.5832 | <u>92.57</u> | **93.80** | <u>73.33</u> | <u>56.67</u> | 2.0887 |
| Drifting-only | 54.27 | 48.40 | 10.91 | 4.5947 | 55.80 | 32.20 | 0.00 | 0.00 | 21.3746 |
| **LD2LM** | **75.61** | **61.20** | 48.08 | 1.5986 | **92.80** | <u>92.40</u> | **80.00** | **60.00** | 2.1366 |

**Training and evaluation protocol.** We continue training DreamReasoner8B on deduplicated PromptCoT (Zhao et al., 2025) code examples. A frozen copy of the starting checkpoint serves as the feature encoder. Unless stated otherwise, code runs use rank-8 LoRA (Hu et al., 2022), $\lambda_{\text{CE}} = 1$, $\lambda_D = 0.1$, a learning rate of $10^{-5}$, and eight H100 ranks with one sequence each. The full code set has 311,050 sequences; one drop-last pass takes 38,881 optimizer steps at global batch size eight. Full runs sample corruption from $[0.1, 1.0]$, while matched mechanism pilots use fixed corruption $0.5$. Rows described as matched share a seed, sample order, training budget, and checkpoint-selection rule. In the tables, E0 is the starting checkpoint, CE continuation uses CE alone, and LD2LM combines CE with Drifting. Where shown, bold and underlining mark the highest and second-highest task scores within a comparable panel; ties share a mark.

**Objective-role conclusion.** Adding Drifting to CE raises HumanEval from 74.39% to 75.61%; on Math, AIME24 rises from 73.33% to 80.00% and AIME25 from 56.67% to 60.00%. Drifting alone is much weaker than CE: HumanEval, MBPP, and LiveCodeBench fall from 74.39%, 61.20%, and 51.03% to 54.27%, 48.40%, and 10.91%; GSM8K, MATH-500, AIME24, and AIME25 fall from 92.57%, 93.80%, 73.33%, and 56.67% to 55.80%, 32.20%, 0%, and 0% (Table 1). TPF is similar for CE and CE+Drifting on Code (1.5832 vs. 1.5986) and Math (2.0887 vs. 2.1366); Drift-only TPF rises to 4.5947 and 21.3746 because degenerate repetitions inflate the measure. Although TokenDrift reports strong Drift-only results, this recipe may not transfer to scaled discrete block-diffusion models.

The stage/KL controls and feature-removal study are separate experimental panels; each comparison uses its own panel baseline. Their LD2LM rows should not be read as a single shared checkpoint.

For code evaluation, we use HumanEval (Chen et al., 2021) (164 tasks), MBPP (Austin et al., 2021b) (500), and LiveCodeBench (Jain et al., 2025) (339 tasks from October 2024 through April 2025). The code tables use the controlled-local generation and scoring protocol, with thresholds specified where they vary. We report pass@1 and tokens per forward pass (TPF), defined as completion tokens divided by all model forward calls, including prefill and cache updates.

### 4.2 Feedback beyond token anchoring

**Joint training and drifting stage.** Figure 3 varies when Drifting is active while retaining CE throughout training. Both schedules that end with Drifting (including Full and Late) improve HumanEval and LiveCodeBench over CE alone while maintaining or improving MBPP. These results support the effectiveness of Drifting through the final training stage, whether used throughout joint training or introduced after an initial CE phase.

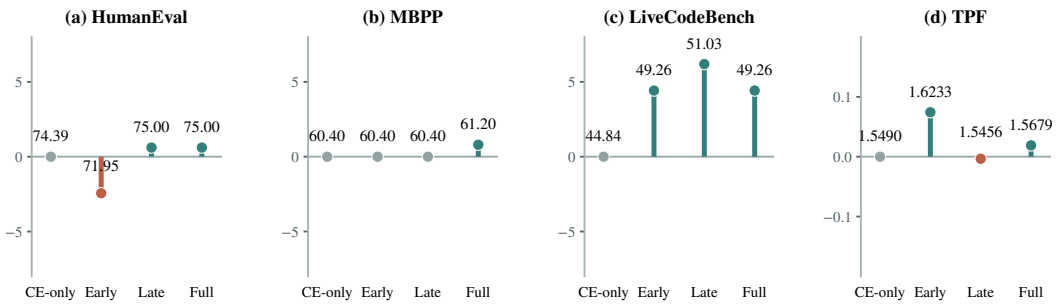


Figure 3: Code scores under four Drifting activation schedules. Annotations are raw scores; stems show changes from CE-only (percentage points for pass@1 and tokens per forward for TPF). Early and Late activate Drifting in the first and final 2,500 updates; Full keeps Drifting active throughout training.

**Additional benefit beyond token KL.** Table 2 compares Drifting with token-level and feature-level KL controls. Combining Drifting with token KL matches or improves upon either objective paired with CE across all three tasks. Adding feature KL to Drifting, however, provides no further gains. This pattern suggests that Drifting complements token-level anchoring, while its feature-space feedback partly overlaps with feature KL.

**Continuous-feature gains across decoding thresholds.** Figure 4 compares CE continuation with LD2LM plus token KL across decoding thresholds. The combined objective improves HumanEval and MBPP at every tested threshold and LiveCodeBench at $\tau \geq 0.80$, with slightly higher mean TPF wherever recorded. These results suggest that the accuracy gains persist across a range of decoding settings at comparable inference efficiency.

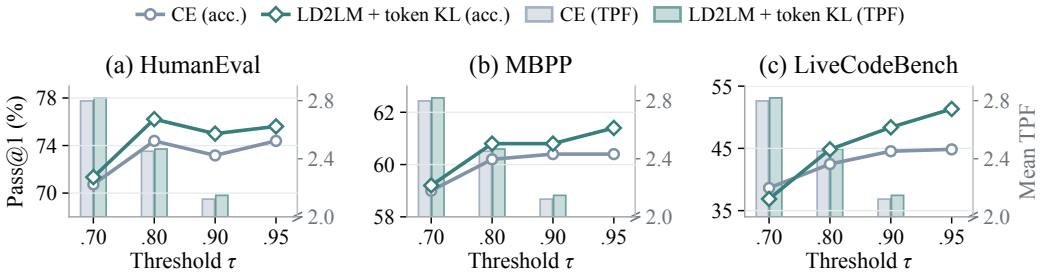


Figure 4: Accuracy and efficiency across decoding thresholds. Lines show benchmark pass@1 (left axes); paired bars show the same cross-benchmark mean TPF in every panel (right axes). Exact values are in Appendix Table 6.

**Gradient alignment across learning spaces.** Figure 1(d) compares the gradient directions of Drifting and the two KL objectives. Drifting has higher median cosine alignment with feature KL than with token KL at every measured training step. This directional similarity suggests that Drifting and feature KL provide partially overlapping optimization signals in feature space, consistent with the KL-control results above. Together, these observations support Drifting as a complement to token-level anchoring. Probe details are provided in Appendix C.

### 4.3 Multiscale feature views

Table 3 shows performance variation across encoder-depth, local-pooling, and feature-statistic configurations, with different tasks favoring different feature selections. These task-dependent effects suggest that the views capture distinct aspects of sequence structure, motivating a multiscale representation that exposes the Drifting objective to a richer set of feature comparisons.

Table 2: Separate token and feature KL controls (Code pass@1, %).

| Training setup | HE | MBPP | LCB |
|---|---|---|---|
| CE continuation | 74.39 | 60.40 | 44.84 |
| LD2LM | 75.00 | 61.20 | 49.26 |
| CE + token KL | 73.78 | **61.40** | 49.56 |
| LD2LM + token KL | **75.61** | **61.40** | 51.33 |
| CE + feature KL | 75.00 | 61.20 | **51.62** |
| LD2LM + feature KL | 75.00 | 60.40 | 47.79 |

Table 3: Code feature-group ablation after 5,000 updates ($Q = 1024$; pass@1, %).

| Removed group | HE | MBPP | LCB |
|---|---|---|---|
| None | 75.61 | 61.20 | 48.08 |
| Low depth | 74.39 | 61.20 | **49.85** |
| Middle depth | **75.00** | 59.80 | 45.72 |
| High depth | 71.34 | **62.00** | **49.85** |
| Local segments | 71.95 | 60.40 | 48.08 |
| Standard deviation | **75.00** | 61.40 | 48.67 |

### 4.4 Historical reference queues

We vary the historical queue capacity to examine how the number of reference features affects training. Figure 5 reports separate Code and Math sweeps. Increasing $Q$ from 1024 to 4096 improves LiveCodeBench under high-confidence decoding ($\tau = 0.90, 0.95$) and improves most Math benchmarks at both checkpoints, including overall accuracy. The clearest Math gains occur on AIME, suggesting that a larger reference set can benefit challenging reasoning tasks.

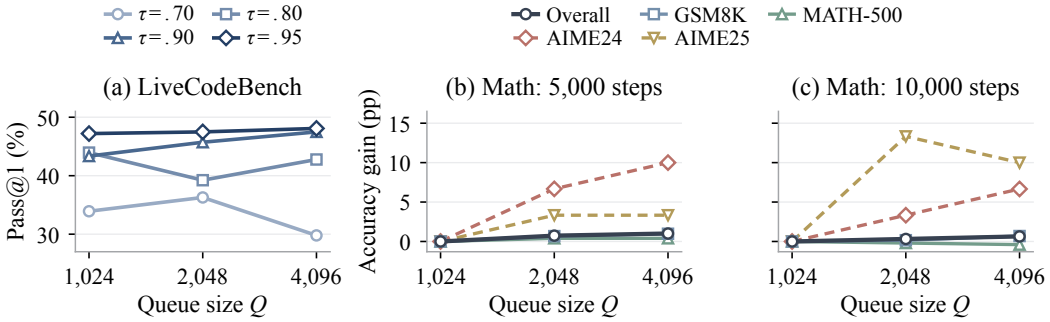


Figure 5: Queue-capacity comparisons. (a) LiveCodeBench pass@1 at four decoding thresholds. (b,c) Math accuracy changes relative to $Q = 1024$ at 5,000 and 10,000 training steps, in percentage points. Math runs use multiscale features and batch-relative feedback; Overall pools 1,879 problems. Full scores and efficiency metrics are in Appendix Tables 4 and 5.

These results support larger historical queues as a useful way to strengthen feature-space supervision, with benefits depending on the task and decoding threshold. A larger queue exposes each current prediction to a broader set of clean and generated reference features. We hypothesize that this broader coverage makes the attraction–repulsion field more representative of the underlying feature distributions and less dependent on the current minibatch, thereby helping shape sequence-level optimization throughout training.

## 5 Conclusion

TokenDrift reports benefits from feature-space Drifting, but its Drift-only recipe performs poorly on DreamReasoner8B, suggesting limited transfer to scaled block diffusion; drawing on scaling experience with richer representations, LD2LM retains CE and adds continuous multiscale feature views without changing inference. In the matched stage study, Full improves HumanEval, MBPP, and LiveCodeBench by 0.61, 0.80, and 4.42 points at 1.22% higher TPF; separately, CE plus Drifting scores 75.61%/61.20%/48.08% on these benchmarks and 80.00%/60.00% on AIME24/AIME25, while adding Drifting to CE with token KL improves HumanEval and LiveCodeBench by 1.83 and 1.77 points and ties MBPP.

### AI USE STATEMENT

We used generative AI tools to edit language, assist with code, and organize experiment records. The authors reviewed all AI-assisted text and checked reported methods, settings, and numbers against source code and saved evaluation results. We take responsibility for the final content.

### REPRODUCIBILITY STATEMENT

Section 3 defines the soft-token mapping, feature views, and reference-bank updates; Appendix A gives the complete update algorithm. Section 4 describes the model, data, training, evaluation, and TPF settings. The supplementary material will include saved configurations, checkpoint-selection rules, dataset hashes, evaluation versions, and commands for the selected runs.

## REFERENCES

Marianne Arriola, Aaron Gokaslan, Justin Chiu, Zhihan Yang, Zhixuan Qi, Jiaqi Han, Subham Sahoo, and Volodymyr Kuleshov. Block diffusion: Interpolating between autoregressive and diffusion language models. In *International Conference on Learning Representations*, volume 2025, pp. 50726–50753, 2025.

Jacob Austin, Daniel D Johnson, Jonathan Ho, Daniel Tarlow, and Rianne Van Den Berg. Structured denoising diffusion models in discrete state-spaces. *Advances in neural information processing systems*, 34:17981–17993, 2021a.

Jacob Austin, Augustus Odena, Maxwell Nye, Maarten Bosma, Henryk Michalewski, David Dohan, Ellen Jiang, Carrie Cai, Michael Terry, Quoc Le, and Charles Sutton. Program synthesis with large language models, 2021b. URL https://arxiv.org/abs/2108.07732.

Adrien Bardes, Jean Ponce, and Yann LeCun. VICReg: Variance-invariance-covariance regularization for self-supervised learning. In *International Conference on Learning Representations*, 2022. URL https://openreview.net/forum?id=xm6YD62D1Ub.

Tiwei Bie, Maosong Cao, Xiang Cao, Bingsen Chen, Fuyuan Chen, Kun Chen, Lun Du, Daozhuo Feng, Haibo Feng, Mingliang Gong, et al. Llada2. 1: Speeding up text diffusion via token editing. *arXiv preprint arXiv:2602.08676*, 2026.

Mark Chen, Jerry Tworek, Heewoo Jun, Qiming Yuan, et al. Evaluating large language models trained on code, 2021. URL https://arxiv.org/abs/2107.03374.

Ting Chen, Simon Kornblith, Mohammad Norouzi, and Geoffrey Hinton. A simple framework for contrastive learning of visual representations. In *International Conference on Machine Learning*, volume 119, pp. 1597–1607, 2020.

Ting Chen, Ruixiang Zhang, and Geoffrey Hinton. Analog bits: Generating discrete data using diffusion models with self-conditioning. In *International Conference on Learning Representations*, 2023. URL https://openreview.net/forum?id=3itjR9QxFw.

Zigeng Chen, Gongfan Fang, Xinyin Ma, Ruonan Yu, and Xinchao Wang. Dmax: Aggressive parallel decoding for dllms. *arXiv preprint arXiv:2604.08302*, 2026.

Ching-Yao Chuang, Joshua W. Robinson, Yen-Chen Lin, Antonio Torralba, and Stefanie Jegelka. Debiased contrastive learning. In *Advances in Neural Information Processing Systems*, volume 33, 2020. URL https://proceedings.neurips.cc/paper/2020/hash/63c3ddcc7b23daa1e42dc41f9a44a873-Abstract.html.

Mingyang Deng, He Li, Tianhong Li, Yilun Du, and Kaiming He. Generative modeling via drifting. *arXiv preprint arXiv:2602.04770*, 2026.

Shansan Gong, Mukai Li, Jiangtao Feng, Zhiyong Wu, and LingPeng Kong. Diffuseq: Sequence to sequence text generation with diffusion models. *arXiv preprint arXiv:2210.08933*, 2022.

Shansan Gong, Shivam Agarwal, Yizhe Zhang, Jiacheng Ye, Lin Zheng, Mukai Li, Chenxin An, Peilin Zhao, Wei Bi, Jiawei Han, Hao Peng, and Lingpeng Kong. Scaling diffusion language models via adaptation from autoregressive models. In *The Thirteenth International Conference on Learning Representations*, 2025. URL https://openreview.net/forum?id=j1tSLYKwg8.

Arthur Gretton, Karsten M. Borgwardt, Malte J. Rasch, Bernhard Schölkopf, and Alexander Smola. A kernel two-sample test. *Journal of Machine Learning Research*, 13(25):723–773, 2012. URL https://www.jmlr.org/papers/v13/gretton12a.html.

Xiaochuang Han, Sachin Kumar, and Yulia Tsvetkov. Ssd-lm: Semi-autoregressive simplex-based diffusion language model for text generation and modular control. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 11575–11596, 2023.

Kaiming He, Haoqi Fan, Yuxin Wu, Saining Xie, and Ross Girshick. Momentum contrast for unsupervised visual representation learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 9729–9738, 2020.

Emiel Hoogeboom, Didrik Nielsen, Priyank Jaini, Patrick Forré, and Max Welling. Argmax flows and multinomial diffusion: Learning categorical distributions. *Advances in neural information processing systems*, 34:12454–12465, 2021.

Emiel Hoogeboom, David Ruhe, Jonathan Heek, Thomas Mensink, and Tim Salimans. Beyond single tokens: Distilling discrete diffusion models via discrete mmd. *arXiv preprint arXiv:2603.20155*, 2026.

Edward J. Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. Lora: Low-rank adaptation of large language models. In *International Conference on Learning Representations*, 2022.

Naman Jain, King Han, Alex Gu, Wen-Ding Li, Fanjia Yan, Tianjun Zhang, Sida Wang, Armando Solar-Lezama, Koushik Sen, and Ion Stoica. LiveCodeBench: Holistic and contamination free evaluation of large language models for code. In *International Conference on Learning Representations*, 2025. URL https://openreview.net/forum?id=chfJJYC3iL.

Xiang Li, John Thickstun, Ishaan Gulrajani, Percy S Liang, and Tatsunori B Hashimoto. Diffusion-lm improves controllable text generation. *Advances in neural information processing systems*, 35: 4328–4343, 2022.

Aaron Lou, Chenlin Meng, and Stefano Ermon. Discrete diffusion modeling by estimating the ratios of the data distribution. *arXiv preprint arXiv:2310.16834*, 2023.

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. *Advances in Neural Information Processing Systems*, 38:50608–50646, 2026.

Daisuke Oba, Hiroki Furuta, and Naoaki Okazaki. Drifting objectives for refining discrete diffusion language models. *arXiv preprint arXiv:2605.19470*, 2026.

Penghui Qi, Xiangxin Zhou, Zichen Liu, Tianyu Pang, Chao Du, Min Lin, and Wee Sun Lee. Rethinking the trust region in LLM reinforcement learning. *arXiv preprint arXiv:2602.04879*, 2026. doi: 10.48550/arXiv.2602.04879. URL https://arxiv.org/abs/2602.04879.

Yuji Ren, Chenkai Xu, Zhuocheng Gong, Jianguo Li, and Zhijie Deng. Cforce: Boosting parallel decoding for dllms via consistency forcing. *arXiv preprint arXiv:2608.13925*, 2026.

Joshua David Robinson, Ching-Yao Chuang, Suvrit Sra, and Stefanie Jegelka. Contrastive learning with hard negative samples. In *International Conference on Learning Representations*, 2021. URL https://openreview.net/forum?id=CR1XOQ0UTh-.

Subham S Sahoo, Marianne Arriola, Yair Schiff, Aaron Gokaslan, Edgar Marroquin, Justin T Chiu, Alexander Rush, and Volodymyr Kuleshov. Simple and effective masked diffusion language models. *Advances in Neural Information Processing Systems*, 37:130136–130184, 2024.

Jiaxin Shi, Kehang Han, Zhe Wang, Arnaud Doucet, and Michalis K. Titsias. Simplified and generalized masked diffusion for discrete data. In *Advances in Neural Information Processing Systems*, volume 37, 2024. doi: 10.52202/079017-3277. URL https://proceedings.neurips.cc/paper_files/paper/2024/hash/bad233b9849f019aead5e5cc60cef70f-Abstract-Conference.html.

Aaron van den Oord, Yazhe Li, and Oriol Vinyals. Representation learning with contrastive predictive coding. In *arXiv preprint arXiv:1807.03748*, 2018.

Tongzhou Wang and Phillip Isola. Understanding contrastive representation learning through alignment and uniformity on the hypersphere. In *Proceedings of the 37th International Conference on Machine Learning*, volume 119 of *Proceedings of Machine Learning Research*, pp. 9929–9939, 2020. URL https://proceedings.mlr.press/v119/wang20k.html.

Xun Wang, Haozhi Zhang, Weilin Huang, and Matthew R. Scott. Cross-batch memory for embedding learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 6388–6397, 2020. URL https://openaccess.thecvf.com/content_CVPR_2020/html/Wang_Cross-Batch_Memory_for_Embedding_Learning_CVPR_2020_paper.html.

Yihong Wu, Liheng Ma, Lingfeng Xiao, Muzhi Li, Xinyu Wang, Yingxue Zhang, and Jian-Yun Nie. Rethinking groups in critic-free RLVR. *arXiv preprint arXiv:2606.17250*, 2026a. doi: 10.48550/arXiv.2606.17250. URL https://arxiv.org/abs/2606.17250.

Zhirong Wu, Yuanjun Xiong, Stella X. Yu, and Dahua Lin. Unsupervised feature learning via non-parametric instance discrimination. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 3733–3742, 2018. URL https://openaccess.thecvf.com/content_cvpr_2018/html/Wu_Unsupervised_Feature_Learning_CVPR_2018_paper.html.

Zirui Wu, Lin Zheng, Jiacheng Ye, Shansan Gong, Xueliang Zhao, Yansong Feng, Wei Bi, and Lingpeng Kong. Dreamreasoner-8b: Block-size curriculum learning for diffusion reasoning models, 2026b. URL https://arxiv.org/abs/2606.19257.

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.

Tianwei Yin, Michaël Gharbi, Taesung Park, Richard Zhang, Eli Shechtman, Frédo Durand, and William T. Freeman. Improved distribution matching distillation for fast image synthesis. In *Advances in Neural Information Processing Systems*, volume 37, 2024a. doi: 10.52202/079017-1505. URL https://papers.nips.cc/paper_files/paper/2024/hash/54dcf25318f9de5a7a01f0a4125c541e-Abstract-Conference.html.

Tianwei Yin, Michaël Gharbi, Richard Zhang, Eli Shechtman, Frédo Durand, William T. Freeman, and Taesung Park. One-step diffusion with distribution matching distillation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 6613–6623, 2024b. URL https://openaccess.thecvf.com/content/CVPR2024/html/Yin_One-step_Diffusion_with_Distribution_Matching_Distillation_CVPR_2024_paper.html.

Jure Zbontar, Li Jing, Ishan Misra, Yann LeCun, and Stephane Deny. Barlow twins: Self-supervised learning via redundancy reduction. In *Proceedings of the 38th International Conference on Machine Learning*, volume 139 of *Proceedings of Machine Learning Research*, pp. 12310–12320, 2021. URL https://proceedings.mlr.press/v139/zbontar21a.html.

Yifan Zhang and Math-AI Team. American invitational mathematics examination (AIME) 2024. Hugging Face dataset, 2024. URL https://huggingface.co/datasets/math-ai/aime24.

Yifan Zhang and Math-AI Team. American invitational mathematics examination (AIME) 2025. Hugging Face dataset, 2025. URL https://huggingface.co/datasets/math-ai/aime25.

Xueliang Zhao, Wei Wu, Jian Guan, Zhuocheng Gong, and Lingpeng Kong. PromptCoT 2.0: Scaling prompt synthesis for large language model reasoning, 2025. URL https://arxiv.org/abs/2509.19894.

Huangjie Zheng, Shansan Gong, Ruixiang Zhang, Tianrong Chen, Jiatao Gu, Mingyuan Zhou, Navdeep Jaitly, and Yizhe Zhang. Continuously augmented discrete diffusion model for categorical generative modeling. In *The Fourteenth International Conference on Learning Representations*, 2026. URL https://iclr.cc/virtual/2026/poster/10010203.

Cai Zhou, Chenxiao Yang, Yi Hu, Chenyu Wang, Chubin Zhang, Muhan Zhang, Lester Mackey, Tommi Jaakkola, Stephen Bates, and Dinghuai Zhang. Coevolutionary continuous discrete diffusion: Make your diffusion language model a latent reasoner, 2025. URL https://arxiv.org/abs/2510.03206.

# A TRAINING ALGORITHM

Algorithms 1 and 2 give the training steps. The Drifting direction follows Deng et al. (2026), and the probability-weighted token embeddings follow Oba et al. (2026), as do the separate clean and generated reference banks and stopped target. Here CE remains active, Drifting follows the selected schedule, and features use the chosen view construction. All reference weights are uniform. The MDLM sampler is unchanged, and restartable checkpoints save both banks. Unlike hard-negative mining in contrastive learning (Robinson et al., 2021), generated references do not receive semantic class labels. The false-negative problem studied by Chuang et al. (2020) is a caution, not a correction used here; we exclude negative pairs with the same sample ID.

**Feature-space field.** For a fixed view $r$ and temperature $\tau$, let $\mathcal{P}$ and $\mathcal{N}$ be its valid positive and negative reference sets, and let $I_r$ be the valid anchors with both reference sides. The vector $v_{jr}$ is reference $j$ in view $r$; weights $w_{ij}^{+}$ and $w_{ij}^{-}$ are computed by Algorithm 2. The weighted attraction–repulsion displacement, its batch RMS scale, and the field averaged over temperatures are defined by

$$\begin{aligned} D_{ir\tau} &= \sum_{j\in\mathcal{P}} w_{ij}^{+}(v_{jr}-h_{ir}) - \sum_{j\in\mathcal{N}} w_{ij}^{-}(v_{jr}-h_{ir}), \\ s_{r\tau} &= \sqrt{\mathrm{mean}_{i\in I_r}\,\|D_{ir\tau}\|_2^2+\epsilon}, \\ V_{ir} &= \begin{cases} \dfrac{1}{|\mathcal{T}|}\displaystyle\sum_{\tau\in\mathcal{T}} \frac{D_{ir\tau}}{s_{r\tau}}, & i\in I_r, \\ 0, & i\notin I_r. \end{cases} \end{aligned} \tag{12}$$

For our experiments, $\mathcal{T}=\{0.02, 0.05, 0.2\}$. Algorithm 2 specifies the weight construction, reference filtering, and same-ID masking.

**Algorithm 1** One global-batch update with CE and feature-space Drifting

**Require:** Clean sequences $\{(x_i, \mathrm{id}_i)\}_{i=1}^{B}$, denoiser $f_\theta$, frozen encoder $\phi$, feature views $\mathcal{R}$
**Require:** FIFO banks $\mathcal{Q}^{+}$ and $\mathcal{Q}^{-}$ (capacity $Q$ per side), $\lambda_{\mathrm{CE}}, \lambda_D, \alpha, \mathcal{T}$
**for** $i = 1, \ldots, B$ **do**
  Sample response corruption rate $\rho_i$ and mask $\mathcal{M}_i$; form corrupted input $\bar{x}_i$
  $\ell_i \leftarrow f_\theta(\bar{x}_i)$ and $p_{it} \leftarrow \mathrm{softmax}(\ell_{it})$
  $c_i \leftarrow \sum_{t\in\mathcal{M}_i} -\log p_{it}(x_{it})/\rho_i$
  $\widetilde{e}_{it} \leftarrow p_{it}E$ for $t \in \mathcal{M}_i$; otherwise $\widetilde{e}_{it} \leftarrow E[\bar{x}_{it}]$
  $h_{ir} \leftarrow \phi_r(\widetilde{e}_i)$ and $u_{ir} \leftarrow \mathrm{sg}(\phi_r(E[x_i]))$ for $r \in \mathcal{R}$
**end for**
$\mathcal{L}_{\mathrm{CE}} \leftarrow (\sum_i c_i)/(\sum_i |\,\mathrm{response}(x_i)|)$
Gather live $h$ with gradients, and gather detached $u, h$ and validity masks across devices
Set $m_i^{+} \leftarrow 1$ and $m_i^{-} \leftarrow 1$ for all $i$
$\mathcal{P}_{\mathrm{cur}} \leftarrow \{(u_i, 1, \mathrm{id}_i)\}$; $\mathcal{N}_{\mathrm{cur}} \leftarrow \{(\mathrm{sg}(h_i), m_i^{-}, \mathrm{id}_i)\}$
$\mathcal{P} \leftarrow \mathcal{P}_{\mathrm{cur}} \cup (\mathcal{Q}^{+} \setminus \mathrm{IDs}_{\mathrm{cur}})$; $\mathcal{N} \leftarrow \mathcal{N}_{\mathrm{cur}} \cup (\mathcal{Q}^{-} \setminus \mathrm{IDs}_{\mathrm{cur}})$
$V \leftarrow \textsc{WeightedField}(h, \mathcal{P}, \mathcal{N}; \mathcal{T})$ ▷ Algorithm 2
$h_{ir}^{\star} \leftarrow \mathrm{sg}(h_{ir} + \alpha V_{ir})$
$\mathcal{L}_D \leftarrow (2B|\mathcal{R}|)^{-1} \sum_{i,r:\,\mathrm{valid}} \|h_{ir} - h_{ir}^{\star}\|_2^2$
Backpropagate $\lambda_{\mathrm{CE}}\mathcal{L}_{\mathrm{CE}} + \lambda_D\mathcal{L}_D$; average and clip generator gradients
Update $\theta$ if the generator gradient norm is nonzero
After a successful update, commit detached $\mathcal{P}_{\mathrm{cur}}, \mathcal{N}_{\mathrm{cur}}$ to their FIFO banks; retain the latest $Q$ valid samples per side

**Algorithm 2** Drifting field with fixed, weighted references across feature views

**Require:** Live anchors $h_{ir}$; detached banks $\mathcal{P}, \mathcal{N}$ with validity masks, sample IDs, and weights; temperatures $\mathcal{T}$

**for** each view $r \in \mathcal{R}$ **do**
  Keep valid positive and negative references of nonzero weight
  Let $I_r$ be anchors with a valid view and both reference sides; set $V_{ir} \leftarrow 0$ otherwise
  **for** $\tau \in \mathcal{T}$ **do**
    Concatenate both sides as $\{(v_j, m_j)\}$ and set $S_{ij} \leftarrow -\|h_{ir} - v_{jr}\|_2^2/\tau$ for $i \in I_r$
    Mask $S_{ij} \leftarrow -\infty$ for same-ID *negative* pairs
    $R_{ij} \leftarrow \mathrm{softmax}_j(S)_{ij}$; $C_{ij} \leftarrow \mathrm{softmax}_{i \in I_r}(S)_{ij}$
    $a_{ij} \leftarrow \sqrt{R_{ij} C_{ij}}\, m_j / \max_k m_k$; split into $a^+$ and $a^-$
    $M_i^{\pm} \leftarrow \sum_j a_{ij}^{\pm}$; $w_{ij}^+ \leftarrow a_{ij}^+ M_i^-$, $w_{ij}^- \leftarrow a_{ij}^- M_i^+$
    $D_{ir\tau} \leftarrow \sum_{j \in \mathcal{P}} w_{ij}^+ (v_{jr} - h_{ir}) - \sum_{j \in \mathcal{N}} w_{ij}^- (v_{jr} - h_{ir})$
    $s_{r\tau} \leftarrow \sqrt{\mathrm{mean}_{i \in I_r} \|D_{ir\tau}\|_2^2 + \epsilon}$
  **end for**
  $V_{ir} \leftarrow |\mathcal{T}|^{-1} \sum_{\tau \in \mathcal{T}} D_{ir\tau}/s_{r\tau}$ for $i \in I_r$
**end for**
**return** $\mathrm{sg}(V)$, with zero field on invalid views

## B QUEUE-CAPACITY RESULTS

Table 4: LiveCodeBench pass@1 (%) by queue and decoding threshold; TPF at $\tau = .90$ (339 tasks).

| $Q$ | $\tau = .70$ | $\tau = .80$ | $\tau = .90$ | $\tau = .95$ | $\mathrm{TPF}_{.90}$ |
|---|---|---|---|---|---|
| 1024 | <u>33.92</u> | **43.95** | 43.36 | 47.20 | **1.692** |
| 2048 | **36.28** | 39.23 | <u>45.72</u> | <u>47.49</u> | <u>1.638</u> |
| 4096 | 29.79 | <u>42.77</u> | **47.49** | **48.08** | 1.630 |

Table 5: Math queue-capacity results for multiscale batch-relative models (1,879 problems; corruption $U[0.1, 1.0]$). TPF is tokens per forward; mean NFE is forward calls per problem.

| Step | $Q$ | Overall | GSM8K | MATH-500 | AIME24 | AIME25 | TPF | Mean NFE |
|---|---|---|---|---|---|---|---|---|
| 5,000 | 1024 | 91.06 | 91.89 | <u>92.40</u> | 63.33 | <u>60.00</u> | 2.0619 | 477.2 |
| 5,000 | 2048 | <u>91.80</u> | <u>92.57</u> | **92.80** | <u>70.00</u> | **63.33** | 2.0168 | 467.3 |
| 5,000 | 4096 | **92.07** | **92.87** | **92.80** | **73.33** | **63.33** | 2.0104 | 459.0 |
| 10,000 | 1024 | 91.22 | 91.96 | **93.40** | 66.67 | 46.67 | 1.9498 | 462.8 |
| 10,000 | 2048 | <u>91.54</u> | <u>92.12</u> | <u>93.20</u> | <u>70.00</u> | **60.00** | 2.0314 | 464.7 |
| 10,000 | 4096 | **91.86** | **92.65** | 93.00 | **73.33** | <u>56.67</u> | 1.9868 | 480.8 |

The 40k-example code runs use multiscale features, uniform reference weights, one seed and data order, and corruption sampled from $U[0.1, 1.0]$. Their active training method and controlled-local HumanEval/MBPP evaluation match apart from $Q$; the earlier $Q = 1024$ run has different inactive KL metadata. Entries are correct/total at $\tau = 0.95$. HumanEval gains five correct solutions between the smallest and largest queues, while MBPP loses four. Even in this pilot, the effect depends on the task. This table covers only HumanEval and MBPP; the full-data LiveCodeBench sweep above is a separate setting.

## C GRADIENT-ALIGNMENT PROBE

Figure 1(d) compares cosine similarity between Drifting and each KL gradient over trainable LoRA parameters. We probe updates 500, 2,500, and 5,000 (circles, squares, and triangles), using the same four batches of eight code examples at corruption rates $0.2$, $0.5$, and $0.8$, using 1,024 historical clean references and 1,024 historical generated references. This gives 36 conditions per KL variant;

horizontal offsets only separate points. The probes make no optimizer update. Token and feature KL use separately trained checkpoints, so the plot is a descriptive comparison rather than a matched causal test. Feature KL has a higher median cosine with Drifting at every measured step, consistent with their shared feature-space supervision and its distinction from token-level anchoring.

## D DENOISING-DIRECTION DIAGNOSTICS

**Denoising directions.** The figure shows stepwise feature-space directions for CE, each KL loss, Drifting, and their combinations.

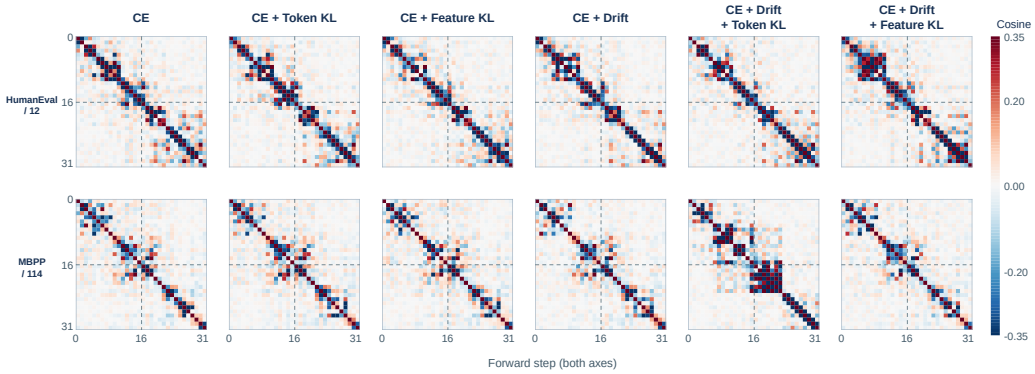


Figure D.1: Feature-direction cosines over 32 measured denoising forwards for HumanEval/12 and MBPP/114. The models share code seed 5678 and the same training budget; this figure uses $\tau = 1.0$. Dashed lines mark the first forward after at least half the masks are resolved. The diagonal compares each step with itself, and other colors are clipped at $\pm 0.35$. Two fixed prompts do not establish a benchmark-level quality difference.

## E DECODING-THRESHOLD RESULTS

Table 6: Full-coverage controlled-local pass@1 and equal-weight mean tokens per forward pass (TPF) across decoding thresholds (HumanEval 164, MBPP 500, LiveCodeBench 339). Higher TPF is better.

| **CE continuation** | | | | | **LD2LM + token KL** | | | | |
|---|---|---|---|---|---|---|---|---|---|
| $\tau$ | HE | MBPP | LCB | Mean TPF | $\tau$ | HE | MBPP | LCB | Mean TPF |
| 0.70 | 70.73 | 59.00 | 38.64 | **2.798** | 0.70 | 71.34 | 59.20 | 36.87 | **2.819** |
| 0.80 | **74.39** | 60.20 | 42.48 | 2.452 | 0.80 | **76.22** | 60.80 | 44.84 | 2.467 |
| 0.90 | 73.17 | **60.40** | 44.54 | 2.122 | 0.90 | 75.00 | 60.80 | 48.38 | 2.148 |
| 0.95 | **74.39** | **60.40** | **44.84** | 1.873 | 0.95 | 75.61 | **61.40** | **51.33** | 1.900 |