



# Next-Concept Modeling: Fast, Parallel Coarse-to-Fine Text Generation

Hermann Kumbong<sup>\*1</sup>, Ishan S. Khare<sup>\*1</sup>,  
Iddah Mlauzi<sup>1</sup>, Andrew Woen<sup>2</sup>, Charles Ding<sup>1</sup>, E. Kelly Buchanan<sup>1</sup>,  
Simran Arora<sup>†3</sup>, Azalia Mirhoseini<sup>†1</sup>, and Christopher Ré<sup>†1</sup>

<sup>1</sup>Department of Computer Science, Stanford University

<sup>2</sup>Department of Electrical Engineering, Stanford University

<sup>3</sup>Together AI

 <https://github.com/HazyResearch/ncm>

## Abstract

Diffusion language models (DLMs) offer parallel sampling for faster generation than autoregressive models, an increasingly important advantage as inference becomes a dominant computational workload. However, DLMs degrade quickly with fewer sampling steps, process the full sequence at every step, and require many passes over the dataset to reach good quality. We introduce **Next-Concept Modeling (NCM)**, a two-stage, coarse-to-fine framework: a multiscale residual VQ-VAE encodes text into a hierarchy of discrete codes, and a transformer generates one level at a time, predicting all codes within a level in parallel. At matched parameter scales, **NCM** achieves better generative perplexity than the strongest DLM baseline on LM1B while running  $21\times$  faster. On OpenWebText, **NCM** is competitive with the strongest DLM while running  $10\times$  faster; on XSum, it matches the strongest DLM’s ROUGE-L while being  $31\times$  faster. On the reasoning task of Sudoku, **NCM** outperforms autoregressive and diffusion baselines across all difficulty levels, using fewer function evaluations.

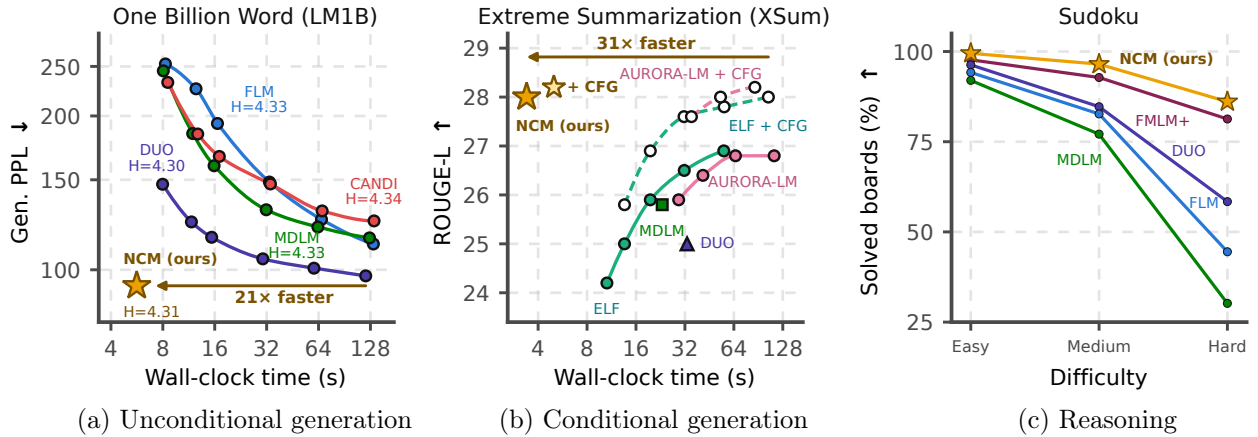


Figure 1: **Generation efficiency and reasoning quality.** (a,b) NCM achieves better or on-par quality with  $21\times$  and  $31\times$  lower wall-clock time on LM1B and XSum, respectively (Section 4.6).  $H$  denotes mean unigram entropy. (c) NCM maintains the highest Sudoku accuracy as difficulty increases, using fewer function evaluations (NFE) than the diffusion baselines.

<sup>\*</sup>Equal contribution. <sup>†</sup>Equal senior authorship.  
Correspondence to {kumbong, iskhare}@stanford.edu.

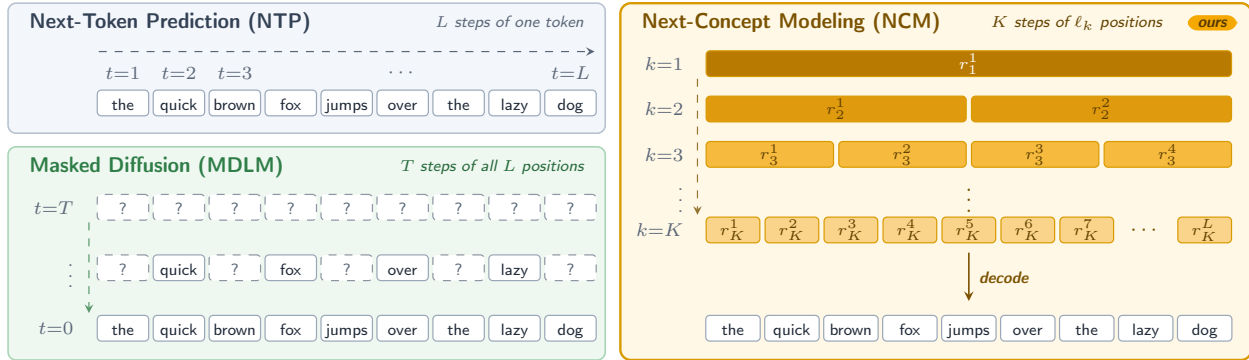


Figure 2: **Comparing text generation paradigms.** Next-token prediction generates one token per step, and masked diffusion (discrete diffusion) refines all  $L$  positions at every step. **NCM** instead generates discrete codes  $r_k^i$ , the  $i$ -th code of level  $k$ , from coarse to fine in  $K \ll L$  steps: each step predicts the  $\ell_k$  codes of one level in parallel, conditioned on the reconstruction  $\hat{z}^{(k-1)}$  of the earlier levels, and after the final level, the cumulative reconstruction is decoded into tokens.

## 1 Introduction

Diffusion language models (DLMs) are the leading non-autoregressive language modeling paradigm [1, 2, 3]. Their main appeal is their ability to produce many tokens at a time through iterative denoising, which can make generation substantially faster than in autoregressive (AR) models that generate one token at a time. This is becoming increasingly important, as language model inference is one of the dominant computational workloads [4, 5]. DLMs are moving from research to practice: open models such as LLaDA [6] scale to billions of parameters, commercial systems such as Mercury [7] and DiffusionGemma [8] serve diffusion models in production, and diffusion drafters like DFlash [9] accelerate AR models through speculative decoding.

The promise of DLMs is parallel sampling: generating a sequence in far fewer sequential steps than it has tokens. In practice however:

- **(i) Many sampling steps.** Diffusion models predict tokens in parallel but still perform successive denoising passes. Reducing these passes can substantially degrade generation quality (Figure 5), limiting the speed gains from parallelism.
- **(ii) Expensive sampling steps.** Processing the full sequence at every step limits speedups, and bidirectional attention prevents direct reuse of AR-style KV caching [10, 11].
- **(iii) Large training budgets.** DLM training often involves many passes over the same data. For example, strong diffusion baselines such as DUO and MDLM are trained on OpenWebText for 1M steps, processing over 500B tokens [12, 3, 13].

Prior work reduces the number of sampling steps mainly through few-step samplers and distillation, either by distilling a many-step teacher into a few-step student [12, 14] or by learning a direct map between noise levels [15]. These approaches come at the cost of additional training stages on top of an already expensive model. Few-step sampling also comes at a price: distilled models tend to be less diverse than the models they are distilled from. On standard benchmarks such as OpenWebText [16] and LM1B [17], they reach a lower generative perplexity at the expense of sample diversity, as measured by mean per-sample unigram entropy [15]. Moreover, these approaches leave the per-step

cost unchanged. Block-autoregressive diffusion enables KV caching at the cost of parallelism: tokens within each block can be denoised in parallel, but blocks must be generated sequentially, with each waiting for the previous block to finish [18, 19].

We introduce **Next-Concept Modeling (NCM)**, a two-stage language modeling framework. A multiscale residual VQ-VAE [20] encodes text into  $K$  levels of discrete codes, from short coarse representations to longer fine ones. A transformer generates these levels sequentially, predicting all codes within each level in parallel, following next-scale prediction for images [21, 22]. **NCM** achieves (i) high generation quality and diversity in  $K \ll L$  steps without distillation (Figure 5); (ii) lower sampling cost by processing short code sequences at early levels (Figure 1); and (iii) strong performance with fewer training passes over the data (Figure 7).

We evaluate **NCM** models at approximately 100M parameters against discrete-diffusion, flow-based, and hybrid baselines, following standard evaluation protocols [3, 12]. Across unconditional generation, conditional generation (translation and summarization), and reasoning tasks (Sudoku), we find:

- **(i) High quality in few steps.** Across three unconditional generation datasets (TinyStories, LM1B, and OpenWebText), **NCM** matches or outperforms strong diffusion baselines with approximately  $4\text{--}11\times$  fewer function evaluations (NFE), while maintaining sample entropy close to the data (Figure 5). On conditional generation (XSum summarization and WMT14 translation), it uses approximately  $6\text{--}16\times$  fewer NFE than unguided baselines, achieving the best summarization scores and outperforming all but one non-AR translation baseline (Table 2). On Sudoku, **NCM** outperforms AR and diffusion baselines across difficulty levels while using fewer NFE than any of them (Table 3).
- **(ii) Fast inference and effective test-time scaling.** Fewer sampling steps and lower per-step cost from shorter coarse levels yield  $10\text{--}31\times$  faster generation at comparable or better quality (Figures 1 and 5). Using the saved time to generate more candidates and select among them with MBR decoding yields about 7% higher ROUGE-L than the strongest summarization baseline within the same sampling-time budget (Figure 6).
- **(iii) Lower training budgets.** **NCM** achieves competitive generation quality with approximately  $4\text{--}5\times$  fewer training tokens than strong diffusion baselines on LM1B and OpenWebText, including both tokenizer and generator training (Figure 7; Appendix C).

We also show preliminary scaling results: increasing **NCM**’s model size improves generation quality on LM1B at a fixed training-token budget (Figure 6a). Qualitative probes suggest that coarse levels capture global structure while finer levels add detail (Appendix H, Figures 17–19).

## 2 Background & Related Work

**Diffusion & Flow-Based Models for Text Generation.** Text generation using Diffusion & Flow-Based Models is an iterative refinement process. Discrete methods operate on token sequences [1, 2, 3]; continuous methods evolve representations of tokens in an embedding space [13, 23, 24, 25]. Distillation/flow-map techniques enable reduction of the number of sampling steps [12, 14, 15]; however, full-sequence formulations still require processing a length- $L$  representation in every step. Block-Autoregressive Diffusion allows KV caching through sequential block generation: tokens within a block are generated in parallel; however, each block must be generated after the previous blocks [18, 19]. **NCM** modifies both the representation and the generation schedule. **NCM** generates a

|                                                                                                                                            |                   |
|--------------------------------------------------------------------------------------------------------------------------------------------|-------------------|
| <b>Translation</b>                                                                                                                         | <b>BLEU: 77.8</b> |
| <b>German:</b> Es besteht allerdings die Sorge, dass kleine und mittlere Unternehmen angreifbar gegenüber Hacking und Überwachung bleiben. |                   |
| <b>Reference:</b> However, there is concern that small and medium-sized companies remain vulnerable to hacking and surveillance.           |                   |

Decoded output after each sampling step, coarse → fine

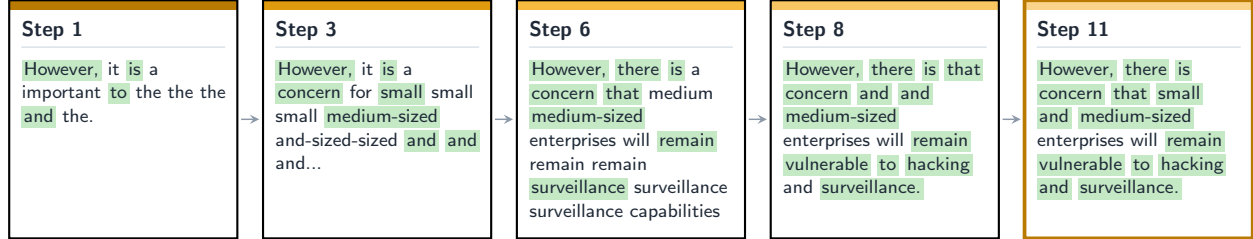


Figure 3: **NCM generation trajectory.** Output decoded after selected sampling steps of one German–English translation (WMT14), shown verbatim; the last step (gold frame) is the final output. Words matching the reference are shaded green. Coarse levels fix the high-level idea; finer levels fill in and correct the content. Appendix J.2 shows all sampling steps with their scores.

learned hierarchy of discrete codes from coarse to fine, and predicts all codes within each level in parallel. Generation therefore follows a short sequence of levels, with fewer positions processed at the early, coarse levels.

**Discrete latent variables and next scale prediction.** A number of image generative models use discrete tokenizers [20, 26, 27]. Residual quantization corrects for errors made at previous steps and uses those errors to create a new codebook [28] which has been adapted to neural audio codecs [29, 30]. Next-scale prediction generates such hierarchies coarse to fine, one scale at a time in parallel, for fast, high-quality images [21, 22]. **NCM** extends residual quantization and next-scale prediction to text through a hierarchy of discrete codes that determines the generation order. While latent diffusion models generate continuous representations of text [31, 32], **NCM** produces successive discrete levels. Adapting this approach to text requires learning useful coarse representations without a natural counterpart to image downsampling. Accurate reconstruction is also important: a single token error can alter meaning, prevent code from running, or invalidate a Sudoku solution.

**Hierarchical and concept-level text generation.** Several approaches organize language modeling around larger units and learned abstractions. MEGABYTE [33] combines global modeling of byte patches with local byte prediction, while H-Net [34] learns dynamic chunks within a multistage hierarchy. Large Concept Models [35] predict sentence representations. ConceptLM [36] predicts discrete concepts to guide subsequent token generation, and Dynamic Large Concept Models [37] learn variable-length concepts for computation in a compressed representation space. HDLM [38] progressively refines tokens through a hierarchy of semantic vocabularies. **NCM** represents an entire sequence through discrete levels of increasing resolution and generates these levels in order using a shared transformer. Predicting all codes within each level in parallel gives a short generation path, while the shorter coarse levels reduce computation during early steps.

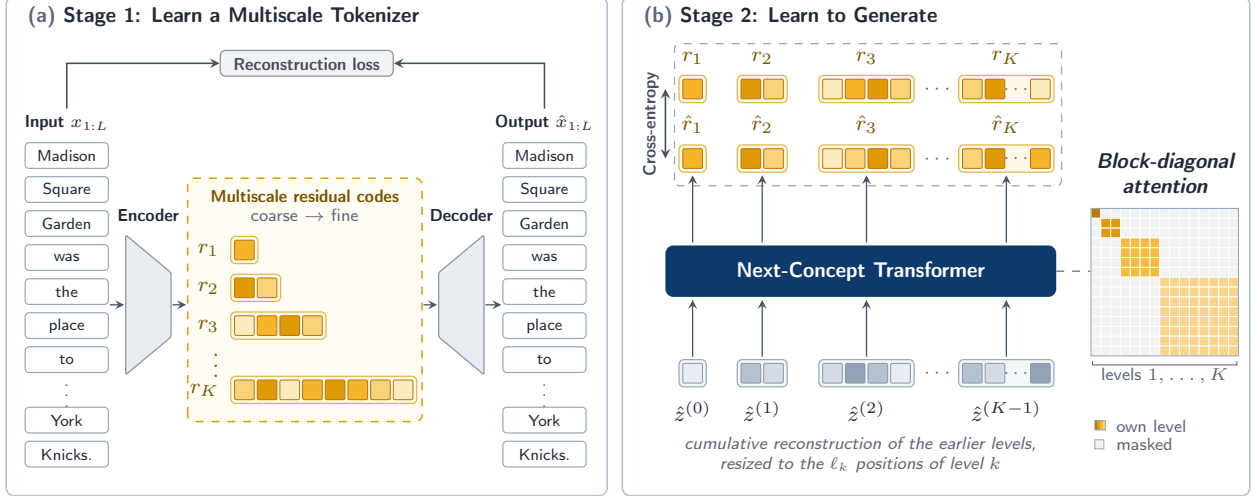


Figure 4: **NCM’s two-stage training.** (a) A multiscale residual VQ-VAE learns to encode text into a coarse-to-fine hierarchy of discrete codes  $r_1, \dots, r_K$  and reconstruct the input. (b) The **NCM** transformer predicts all codes within level  $k$  in parallel from the cumulative reconstruction  $\hat{z}^{(k-1)}$ , resized to  $\ell_k$  positions. In the unconditional setting, block-diagonal attention restricts each input block to itself. During training, all  $K$  levels are predicted in one forward pass using a cross-entropy loss. During generation, levels are predicted sequentially from coarse to fine.

### 3 Next-Concept Modeling

NCM generates text through a learned coarse-to-fine hierarchy of discrete codes. We first train a multiscale residual VQ tokenizer to represent text at progressively finer levels of detail, then freeze it and train a transformer to generate the hierarchy. At each level, the transformer predicts all codes in parallel, conditioned on the cumulative reconstruction of earlier levels. The tokenizer’s decoder maps the completed reconstruction back to text. Figure 3 previews this process through a German–English translation, showing how the output develops across levels. The term *concept* here refers to the information carried by the coarse levels of the hierarchy.

#### 3.1 Preliminaries

Given latent vectors  $z \in \mathbb{R}^{L \times d}$  and a codebook  $\mathcal{V} = \{e_1, \dots, e_{|\mathcal{V}|}\}$ , with entries  $e_j \in \mathbb{R}^d$ , *vector quantization* [20] replaces each vector  $z^i$  with its nearest codebook entry:

$$z_q^i = \arg \min_{e \in \mathcal{V}} \|z^i - e\|_2^2 \quad (1)$$

The selected codebook vectors form the quantized latent  $z_q \in \mathbb{R}^{L \times d}$ .

A single quantization step generally leaves an approximation error. *Residual quantization* [29] represents this error using additional codebooks, adding each quantized correction to the reconstruction. Starting from the residual  $z^{(0)} = z$  and the reconstruction  $\hat{z}^{(0)} = 0$ , let  $z^{(k),i}$  denote the residual vector at position  $i$  after level  $k$ , and  $\hat{z}^{(k),i}$  the corresponding cumulative reconstruction vector. Level  $k$  quantizes the residual into  $L$  codes  $r_k$  and updates the reconstruction and residual. Here  $e_k(r_k)$  denotes lookup in level  $k$ ’s codebook, with  $e_k(r_k)^i = e_{k,r_k^i}$ :

$$\hat{z}^{(k)} = \hat{z}^{(k-1)} + e_k(r_k), \quad z^{(k)} = z - \hat{z}^{(k)} \quad (2)$$

The sum  $\hat{z}^{(K)} = \sum_{k=1}^K e_k(r_k)$  approximates  $z$ , and each partial sum  $\hat{z}^{(k)}$ , the *cumulative reconstruction*, is a partial reconstruction using the first  $k$  levels. *Multiscale residual quantization* [21] uses progressively longer code sequences, with level  $k$  containing  $\ell_k$  codes. Its codebook vectors are mapped to a length- $L$  contribution by a reconstruction operator  $R_k$  before being added to the cumulative reconstruction, allowing shorter, coarse levels to be followed by finer levels that add detail.

### 3.2 Formulation

Let  $x \in \mathcal{T}^L$  be a sequence of  $L$  tokens from a text vocabulary  $\mathcal{T}$ . The tokenizer maps  $x$  to discrete codes at  $K$  levels,

$$r_{1:K} = (r_1, \dots, r_K), \quad r_k \in \{1, \dots, |\mathcal{V}_k|\}^{\ell_k}, \quad (3)$$

where  $\mathcal{V}_k$  is level  $k$ 's codebook,  $|\mathcal{V}_k|$  is its size,  $\ell_k$  is the number of codes at that level, and  $\ell_1 < \ell_2 < \dots < \ell_K = L$ . Each level encodes the residual left by the earlier ones. The generator predicts the levels from coarse to fine, conditioning on earlier levels through their cumulative reconstruction  $\hat{z}^{(k-1)}$ :

$$p_\theta(r_{1:K}) = \prod_{k=1}^K p_\theta(r_k \mid \hat{z}^{(k-1)}) \quad (4)$$

Conditional on this reconstruction, the codes within each level are sampled independently, allowing all positions to be generated in one forward pass:

$$p_\theta(r_k \mid \hat{z}^{(k-1)}) = \prod_{i=1}^{\ell_k} p_\theta(r_k^i \mid \hat{z}^{(k-1)}) \quad (5)$$

After all levels are generated, the tokenizer's decoder maps their cumulative reconstruction  $\hat{z}^{(K)}$  to token distributions at all  $L$  positions.

### 3.3 Multiscale Residual Tokenization

To obtain the hierarchy, we learn a multiscale residual VQ-VAE, using nested dropout to encourage informative partial reconstructions. An encoder  $E_\phi$  maps the input sequence  $x$  to  $L$  latent vectors of dimension  $d$ :

$$z = E_\phi(x) \in \mathbb{R}^{L \times d} \quad (6)$$

Starting from  $z^{(0)} = z$  and  $\hat{z}^{(0)} = 0$ , each level quantizes the residual left by the preceding levels. At level  $k$ , the residual is downsampled to  $\ell_k$  latent vectors, and each vector is assigned the index of its nearest entry in the codebook  $\mathcal{V}_k$ :

$$u_k = \text{Down}_k(z^{(k-1)}) \in \mathbb{R}^{\ell_k \times d}, \quad r_k^i = \arg \min_{j \in \{1, \dots, |\mathcal{V}_k|\}} \|u_k^i - e_{k,j}\|_2^2 \quad (7)$$

The reconstruction operator  $R_k$  maps the selected codebook vectors to a length- $L$  contribution, which we add to the cumulative reconstruction:

$$\hat{z}^{(k)} = \hat{z}^{(k-1)} + R_k(e_k(r_k)), \quad z^{(k)} = z - \hat{z}^{(k)} \quad (8)$$

Here  $e_k(r_k) \in \mathbb{R}^{\ell_k \times d}$  contains the selected codebook vectors, and both  $\hat{z}^{(k)}$  and  $z^{(k)}$  have shape  $L \times d$ .

The VQ-VAE decoder defines token distributions  $p_\psi(x^i \mid \hat{z}^{(k)})$  at all  $L$  positions from a cumulative reconstruction  $\hat{z}^{(k)}$ .

**Nested dropout.** We use nested dropout [39], which during training retains only the first  $k$  levels for a sampled cutoff  $k \in \{1, \dots, K\}$ , giving  $\hat{z}^{(k)} = \sum_{j=1}^k R_j(e_j(r_j))$ . This encourages earlier levels to capture meaningful information rather than relying on finer levels to carry it. During training, both the reconstruction cross-entropy and the VQ terms are evaluated at the sampled cutoff:

$$\mathcal{L}_{\text{tok}} = -\frac{1}{L} \sum_{i=1}^L \log p_{\psi}(x^i | \hat{z}^{(k)}) + \beta \text{MSE}(z, \text{sg}[\hat{z}^{(k)}]) + \text{MSE}(\text{sg}[z], \hat{z}^{(k)}) \quad (9)$$

The VQ terms combine a commitment loss and a latent quantization loss, both measured by mean squared error (MSE). Here sg denotes stop-gradient and  $\beta$  weights the commitment term. We use a straight-through estimator to pass reconstruction gradients through the discrete quantization step to the encoder.

### 3.4 Coarse-to-Fine Generation

With the tokenizer frozen, we train the **NCM** transformer on all levels in one forward pass (Algorithm 1); generation proceeds from coarse to fine in  $K$  steps (Algorithm 2).

**Conditioning Corruption.** To improve robustness to accumulating VQ code errors during generation, we train with a fraction of conditioning codes replaced by other codes from the same level’s codebook, keeping the ground-truth codes intact as labels. Replacement probabilities vary by level; ablations are given in Appendix F.3. Let  $\tilde{z}^{(k-1)}$  denote the resulting cumulative reconstruction. The training objective is

$$\mathcal{L}_{\text{NCM}} = -\sum_{k=1}^K w_k \sum_{i=1}^{\ell_k} \log p_{\theta}(r_k^i | \tilde{z}^{(k-1)}), \quad w_k = \frac{\ell_k^{-\alpha}}{\sum_{j=1}^K \ell_j^{1-\alpha}} \quad (10)$$

With  $\alpha = 0$ , all code positions have equal weight, and with  $\alpha = 1$ , each level has equal weight.

---

#### Algorithm 1 NCM training step

---

**Require:** Text  $x$ , frozen tokenizer,  $\theta$   
1: Encode  $x$  into target codes  $r_{1:K}$   
2: Uniformly replace codes to obtain  $\tilde{r}_{1:K}$   
3:  $\tilde{z}^{(0)} \leftarrow 0$   
4: **for**  $k = 1, \dots, K$  **do**  
5:   Resize and project  $\tilde{z}^{(k-1)}$  to  $\ell_k$  inputs  
6:    $\tilde{z}^{(k)} \leftarrow \tilde{z}^{(k-1)} + R_k(e_k(\tilde{r}_k))$   
7: **end for**  
8: Predict all levels in one forward pass  
9: Compute  $\mathcal{L}_{\text{NCM}}$  against  $r_{1:K}$   
10: Update  $\theta$  using the loss gradient

---



---

#### Algorithm 2 NCM generation

---

**Require:** Frozen tokenizer, generator  $p_{\theta}$   
1:  $\hat{z}^{(0)} \leftarrow 0$   
2: **for**  $k = 1, \dots, K$  **do**  
3:   Resize  $\hat{z}^{(k-1)}$  to  $\ell_k$  positions  
4:   Project the resized vectors to the hidden width  
5:   Predict code distributions in one forward pass  
6:   Sample all  $\ell_k$  codes  $r_k^i$  in parallel  
7:    $\hat{z}^{(k)} \leftarrow \hat{z}^{(k-1)} + R_k(e_k(r_k))$   
8: **end for**  
9: Decode  $\hat{z}^{(K)}$  to token distributions  
10: Select output tokens by argmax

---

**Conditional generation.** **NCM** supports prompt-conditioned generation through a prefix-LM attention pattern. Prompt tokens are embedded using the frozen VQ-VAE encoder’s token embedding table and projected into the hidden space. Prompt positions attend to each other; each target block attends to the prompt and itself, while prompt positions cannot attend to target blocks.

## 4 Experiments

We evaluate **NCM** along four axes:

- **Quality.** We first assess tokenizer quality through reconstruction accuracy and codebook utilization (§4.1). We then evaluate unconditional generation on TinyStories [40], One Billion Word (LM1B) [17], and OpenWebText (OWT) [16], measuring generative perplexity alongside sample diversity (§4.2). For conditional generation, we evaluate WMT14 German–English translation [41] using BLEU [42, 43] and XSum summarization [44] using ROUGE-1/2/L [45] (§4.3).
- **Efficiency.** We compare the number of function evaluations (NFE) and wall-clock time required to achieve a given generation quality, as well as the training-token budgets used to obtain these results (§4.6).
- **Reasoning.** We compare **NCM** with autoregressive and diffusion baselines on Sudoku, where successful generation requires satisfying interdependent constraints across the entire output (§4.4).
- **Scaling.** We investigate how scaling the model size improves generation quality and whether **NCM**’s fast sampling can translate into further quality gains through additional test-time compute (§4.5).

Across these evaluations, **NCM** achieves competitive generation quality with substantially lower sampling cost. We train a separate tokenizer and generator for each dataset. Unless otherwise stated, our generators have approximately 100M parameters, comparable to or smaller than the baselines. We follow established evaluation protocols from prior diffusion language modeling work where possible. Appendices B–E provide dataset, architecture, training, and baseline details, along with additional results.

### 4.1 Tokenizer Quality

|                   | Unconditional |       |             | Conditional |       | Reasoning (Sudoku) |        |        |
|-------------------|---------------|-------|-------------|-------------|-------|--------------------|--------|--------|
|                   | OWT           | LM1B  | TinyStories | XSum        | WMT14 | Easy               | Medium | Hard   |
| Recon. acc. (%) ↑ | 99.99         | 99.86 | 99.98       | 99.79       | 98.79 | 100.00             | 100.00 | 100.00 |
| Utilization (%)   | 83.10         | 89.31 | 90.12       | 86.20       | 61.89 | 80.73              | 83.85  | 80.73  |

Table 1: **Tokenizer reconstruction and codebook utilization.** Reconstruction accuracy measures exact token recovery on held-out text using the full hierarchy. Utilization is the mean percentage of codebook entries used across levels.

Reconstruction errors that are visually imperceptible in images can be consequential in text: e.g., one wrong digit can invalidate a Sudoku solution. Table 1 reports exact reconstruction accuracy. On Sudoku, where correctness requires exact recovery, the tokenizer achieves 100% reconstruction accuracy. On other tasks, like summarization and translation, token-level reconstruction accuracy is close to 100%. Observed tokenizer reconstruction errors often involve substitutions of related words or entities, with their effect on meaning depending on the context. We also report tokenizer codebook utilization in Table 1, with mean utilization ranging from 62% to 90% across datasets. Appendix G

provides further tokenizer analysis, including reconstruction errors and codebook utilization. We provide additional experiments in Appendix H to further understand the learned hierarchy.

## 4.2 Unconditional Generation

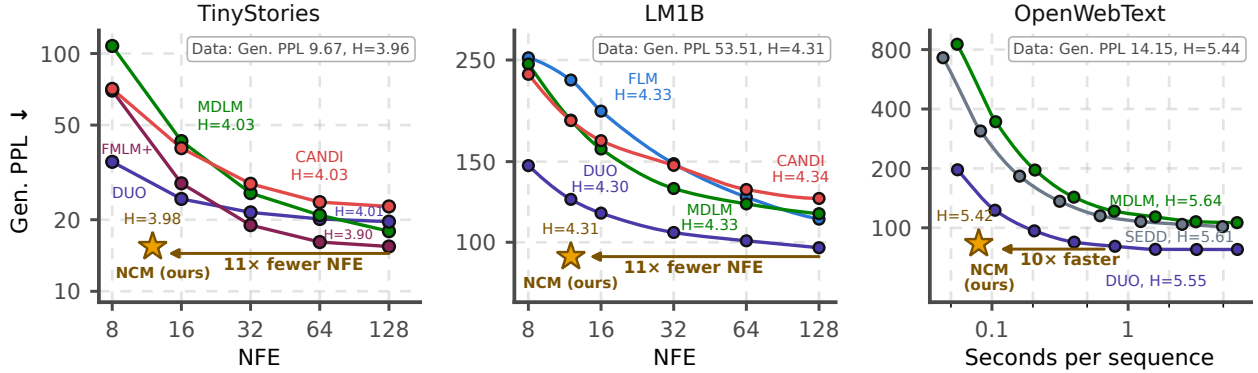


Figure 5: **Unconditional generation quality and sampling cost.** Generative perplexity (Gen. PPL; lower is better) against the number of function evaluations on TinyStories and LM1B, and against the sampling time per 1,024-token sequence (batch size 16) on OpenWebText. **NCM** achieves competitive or better Gen. PPL compared to strong diffusion baselines with substantially fewer function evaluations and less time.  $H$  denotes mean per-sample unigram entropy.

Following standard practice in the diffusion language modeling literature [3, 12, 46], we evaluate **NCM**’s unconditional generation quality using generative perplexity (Gen. PPL), with GPT-2-large as the reference model on TinyStories [40], LM1B [17], and OpenWebText (OWT) [16]. We report mean per-sample unigram entropy ( $H$ ) alongside Gen. PPL, since lower Gen. PPL can be obtained by reducing sample diversity [46]. For reference, we report both metrics on ground-truth text from each dataset. For a fair comparison, **NCM** is compared against baselines with sample diversity close to that of the ground-truth data. Figure 5 compares **NCM** with strong discrete and continuous diffusion language model baselines in terms of Gen. PPL at different numbers of function evaluations (NFE). Across the three datasets, **NCM** achieves competitive or better Gen. PPL with substantially fewer function evaluations, while maintaining sample entropy close to that of the data. Table 8 in Appendix E.2 lists all results with their entropies, and Appendix E.1 gives results under other reference scoring models.

## 4.3 Conditional Generation

We evaluate **NCM** on translation (WMT14 German–English, De–En) [41] and extreme summarization (XSum) [44], using BLEU [42, 43] and ROUGE-1/2/L [45] scores, respectively. We compare **NCM** with autoregressive baselines [13] and diffusion and flow models [3, 12, 47, 48, 13, 25] at comparable model sizes (Table 2). Without guidance, **NCM** achieves better ROUGE-1/2/L scores on XSum than the strongest diffusion baseline, AURORA-LM [25], while reducing the number of function evaluations (NFE) by 16 $\times$ . Among the non-AR baselines on WMT14, **NCM** trails only the strongest diffusion baseline, E2D2 [47], while using 5.8 $\times$  fewer NFE. **NCM**’s lower sampling cost allows us to match the baselines’ NFE budget using multiple candidates and minimum Bayes risk (MBR) decoding. **NCM** with MBR achieves better scores on all three XSum ROUGE metrics than the strongest diffusion baseline with guidance, AURORA-LM [25], while using fewer NFE (Table 2). We count each classifier-free guidance step as two function evaluations, since it requires

both a conditional and an unconditional generator call. We visualize generated samples in Figure 8 and Appendix J.

| Type                                 | Method                   | WMT14 De-En |                 |                  | XSum |                    |                    |                    |                  |
|--------------------------------------|--------------------------|-------------|-----------------|------------------|------|--------------------|--------------------|--------------------|------------------|
|                                      |                          | Size        | BLEU $\uparrow$ | NFE $\downarrow$ | Size | ROUGE-1 $\uparrow$ | ROUGE-2 $\uparrow$ | ROUGE-L $\uparrow$ | NFE $\downarrow$ |
| –                                    | AR <sup>†</sup>          | 99M         | 25.2            | 64               | 99M  | 30.5               | 10.2               | 24.4               | 64               |
| <i>Without guidance</i>              |                          |             |                 |                  |      |                    |                    |                    |                  |
| Diff.                                | MDLM <sup>†</sup>        | 99M         | 18.4            | <u>64</u>        | 99M  | 33.4               | 11.6               | 25.8               | <u>64</u>        |
|                                      | DUO <sup>†</sup>         | 205M        | 21.3            | <u>64</u>        | 205M | 31.4               | 10.1               | 25.0               | <u>64</u>        |
|                                      | E2D2 <sup>†</sup>        | 99M         | <b>24.8</b>     | <u>64</u>        | 99M  | 28.4               | 8.3                | 22.0               | <u>64</u>        |
|                                      | SeqDiffuSeq <sup>†</sup> | –           | 21.3            | <u>64</u>        | –    | 19.3               | 1.7                | 14.1               | <u>64</u>        |
| Flow                                 | ELF                      | 140M        | 17.2            | <u>64</u>        | 140M | 34.2               | 11.1               | 26.4               | <u>64</u>        |
|                                      | AURORA-LM                | –           | –               | –                | 130M | <u>34.7</u>        | <u>12.2</u>        | <u>26.6</u>        | 128              |
| Ours                                 | <b>NCM</b>               | 120M        | <u>22.6</u>     | <b>11</b>        | 88M  | <b>35.0</b>        | <b>12.8</b>        | <b>28.0</b>        | <b>8</b>         |
| <i>With guidance or MBR decoding</i> |                          |             |                 |                  |      |                    |                    |                    |                  |
| Flow                                 | ELF                      | 140M        | <b>26.4</b>     | <u>128</u>       | 140M | 36.0               | 12.2               | 27.8               | <u>128</u>       |
|                                      | AURORA-LM                | –           | –               | –                | 130M | <u>36.3</u>        | <u>13.4</u>        | <u>28.6</u>        | 184              |
| Ours                                 | <b>NCM + CFG</b>         | 120M        | 23.9            | <b>22</b>        | 88M  | 35.3               | 13.0               | 28.2               | <b>16</b>        |
|                                      | <b>NCM + MBR</b>         | 120M        | <u>25.2</u>     | 132              | 88M  | <b>37.3</b>        | <b>14.0</b>        | <b>29.6</b>        | <u>128</u>       |

Table 2: **Conditional generation quality and sampling cost.** BLEU on WMT14 De-En and ROUGE-1/2/L on XSum (higher is better). Size denotes task-specific generator parameters; NFE counts generator calls per output. Bold and underlined values mark the best and second-best results within each non-AR decoding setting, including ties. <sup>†</sup>: results from Hu et al. [13].

#### 4.4 Reasoning

We evaluate **NCM** on Sudoku, a reasoning task in which every cell of the solution must satisfy the row, column, and box constraints at once. Following the setup of S-FLM [50], we report the percentage of completely solved boards at three difficulty levels defined by the number of provided clues (40, 35, and 30 for easy, medium, and hard, respectively). **NCM** outperforms both the AR baselines and the strongest diffusion baseline, FMLM+ [49], across all three levels of difficulty while using fewer NFE (Table 3). **NCM**’s coarse-to-fine generation is well suited to tasks that require global planning: **NCM** refines a representation of the entire board across levels, allowing later levels to correct earlier partial solutions. In contrast, the AR baselines struggle with left-to-right generation, which commits to individual tokens without revising earlier decisions. Appendix J.7 shows a sample Sudoku solving trajectory across different time steps.

| Type  | Method     | Size | NFE $\downarrow$ | Accuracy (%) $\uparrow$ |             |             |
|-------|------------|------|------------------|-------------------------|-------------|-------------|
|       |            |      |                  | Easy                    | Med.        | Hard        |
| AR    | Sample     | 25M  | 128              | 13.9                    | 5.1         | 0.6         |
|       | Greedy     | 25M  | 128              | 14.6                    | 5.1         | 1.0         |
| Disc. | MDLM       | 29M  | 128              | 92.0                    | 77.1        | 30.2        |
|       | DUO        | 29M  | 128              | 96.3                    | 84.7        | 58.4        |
| Cont. | CANDI      | –    | 128              | 79.3                    | 45.9        | 16.7        |
|       | FLM        | 29M  | 128              | 94.2                    | 82.7        | 44.5        |
|       | S-FLM      | 29M  | 128              | 94.8                    | 85.2        | 45.0        |
|       | FMLM+      | 29M  | 16               | 97.7                    | 92.8        | 81.3        |
| Ours  | <b>NCM</b> | 25M  | <b>12</b>        | <b>99.5</b>             | <b>96.5</b> | <b>86.1</b> |

Table 3: **Sudoku.** Completely solved boards (%) by difficulty. Baseline results: Agarwal et al. [49]. **NCM** outperforms all baselines with fewer NFE.

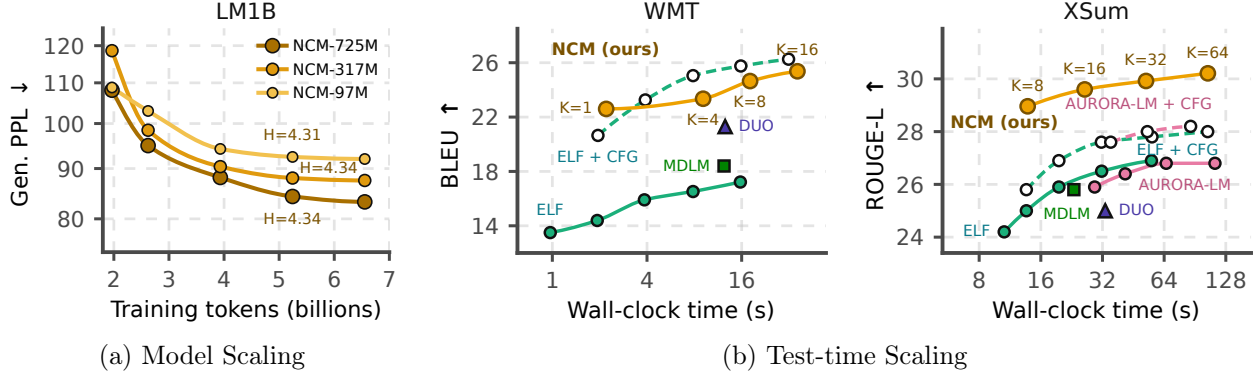


Figure 6: **Model and test-time scaling.** (a) Increasing generator size improves Gen. PPL while maintaining high sample entropy ( $H$ ) on LM1B at a fixed generator training-token budget; the shared tokenizer is excluded from this panel. (b) Generating more candidates and selecting with MBR decoding improves BLEU on WMT14 and ROUGE-L on XSum.  $K$  denotes the number of candidates; wall-clock time measures sampling cost. On XSum, **NCM** outperforms diffusion baselines at matched inference time.

## 4.5 Scaling

We study how **NCM** benefits from additional compute through larger models and increased inference budgets. We evaluate model scaling by increasing generator size and test-time scaling by selecting among multiple candidates using minimum Bayes risk (MBR) decoding.

**Model Scaling.** Scaling **NCM**'s model size improves both generation quality and sample diversity at a fixed training-token budget (Figure 6a). We train generators with 97M, 317M, and 725M parameters on LM1B, all sharing the same frozen tokenizer. These results suggest that **NCM** scales effectively, with potential for further gains from larger generators.

**Test-Time Scaling.** In Figure 1 and Table 2, we show that **NCM** matches or outperforms diffusion baselines while using significantly fewer function evaluations (NFEs) and less wall-clock time. A natural question is whether we can convert these efficiency gains into further improvements in quality. We generate multiple candidates and use minimum Bayes risk (MBR) decoding [51] to select the candidate with the highest average similarity to the others. On XSum, Figure 6b shows that **NCM** improves over diffusion baselines at matched inference time.

## 4.6 Efficiency

**Inference Efficiency.** In Figure 5, we show that **NCM** achieves lower or comparable generative perplexity than the strongest diffusion baselines using far fewer function evaluations (NFE). However, NFE alone does not capture the full inference efficiency picture, since the cost per step differs across different diffusion language models; many DLMs process the full sequence at every step and use samplers that can introduce additional overhead. **NCM**, on the other hand, processes very few tokens at the coarse levels of its hierarchy, saving compute at these early steps. We benchmark end-to-end wall-clock time for generating 1,024 sequences at batch size 16 on a single NVIDIA H100 SXM 80GB GPU (Figure 1). On LM1B, **NCM** uses  $11\times$  fewer NFE than the strongest diffusion language model baseline, DUO, and runs  $21\times$  faster at better generative perplexity. On XSum, it uses  $16\times$  fewer NFE than the strongest diffusion baseline, AURORA-LM, and runs  $31\times$  faster at

comparable ROUGE. The speedup gains come from both fewer NFE and a lower average cost per step compared to the DLM baselines.

**Lower Training Budgets.** NCM achieves strong unconditional generation results with fewer training tokens than the diffusion baselines. On LM1B, the tokenizer and generator together process 13.36B full-length token positions, approximately  $4.9\times$  fewer than the baselines’ 66B (Figure 7). We count repeated passes over the data in this budget; the calculation is given in Appendix C.2. These budgets measure total token exposure. One possible explanation is that NCM’s fixed coarse-to-fine order simplifies learning relative to the many generation orders learned by diffusion baselines.

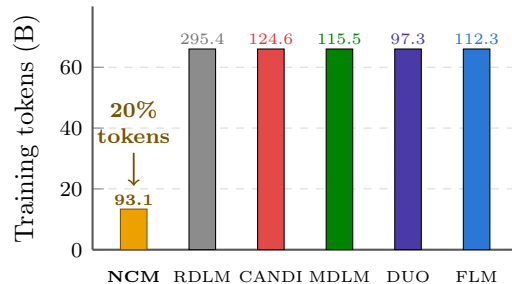


Figure 7: **Training tokens on LM1B.** NCM uses  $4.9\times$  fewer tokens and reaches a lower Gen. PPL (above each bar).

## 4.7 Qualitative Samples

Figure 8 shows NCM samples across unconditional and conditional settings. The conditional outputs remain anchored to their input contexts. Additional samples are provided in Appendix J.

|                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                       |  |                                                                    |
|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|--|--------------------------------------------------------------------|
| <b>Summarization</b>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                  |  | <b>ROUGE-1:</b> 48.5   <b>ROUGE-2:</b> 12.9   <b>ROUGE-L:</b> 36.4 |
| <b>Context:</b> The 33-year-old has featured only twice for the Foxes this term, having signed a new one-year deal with the Premier League newcomers in the summer. Former Blackpool forward Taylor-Fletcher scored three goals in 23 games for his parent club last season. He joined the 13th-placed Owls prior to Saturday’s Championship game against Norwich City.                                                                                                                                                                                                                               |  |                                                                    |
| <b>Reference:</b> Leicester City striker Gary Taylor-Fletcher has joined Sheffield Wednesday on an initial month-long loan.                                                                                                                                                                                                                                                                                                                                                                                                                                                                           |  |                                                                    |
| <b>Generated:</b> Sheffield Wednesday have signed Leicester striker James Taylor-Fletcher on loan until the end of the season.                                                                                                                                                                                                                                                                                                                                                                                                                                                                        |  |                                                                    |
| <b>Translation</b>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                    |  | <b>BLEU:</b> 51.9                                                  |
| <b>German:</b> Leider gibt es kein Wundermittel zur Vorbeugung von Krebs.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                             |  |                                                                    |
| <b>Reference:</b> Unfortunately, there is no miracle recipe for preventing cancer.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                    |  |                                                                    |
| <b>Generated:</b> Unfortunately, there is no miracle cure to prevent cancer.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                          |  |                                                                    |
| <b>Conditional Generation</b>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                         |  | <b>Prompt</b>   <b>Generated</b>                                   |
| <b>Sample 1:</b> An Egyptian court has ordered a probe into fresh clashes between police and protesters in Cairo. “There are strikes at the Suez Canal,” witnesses said. “It’s going to be happening in the streets. It’s the new regime.” Those police, responding to the protests, are now getting the money out of the government, he said. . .                                                                                                                                                                                                                                                    |  |                                                                    |
| <b>Sample 2:</b> But Church explosion and billions of thought, frustration, morals, and religion had itself bound inextricably in the fabric of beauty rather than reason. The decay of society fell into the wealth of divine advancement at hand. The power of social engineering perused human godhood into a sense of direction.                                                                                                                                                                                                                                                                  |  |                                                                    |
| <b>Unconditional Generation</b>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                       |  | <b>Gen. PPL:</b> 55.6   <b>Entropy:</b> 4.39                       |
| <b>Generated:</b> [CLS] at the far end of the series around 18 games. [CLS] "it could be good if he doesn't simply name the party that has won his past two on the other side and the corners, " he said. [CLS] "if you are in power then you can put your players down and work together and try find out the changes with which we need. [CLS] i work in the various schools or into children's school. [CLS] saunders, who was also hit after britain's most successful player in 30 years, had been for six appearances since starting 11 with an 11–14 major league record, a 28-year high [CLS] |  |                                                                    |

Figure 8: **Qualitative samples from NCM.** Unconditional generation samples are on LM1B. Conditional generation samples use prompts from a held-out OWT validation set. Summarization and translation samples are on XSum and WMT14 De–En, respectively. Additional samples are provided in Appendix J.

## 5 Conclusion

We introduce **Next-Concept Modeling (NCM)**, a two-stage framework that first learns a discrete coarse-to-fine representation of text and then trains a transformer to generate its levels in order. Predicting all codes within each level in parallel provides a short sequential generation path. On unconditional generation across TinyStories, LM1B, and OpenWebText, **NCM** achieves competitive or better Gen. PPL than strong diffusion baselines with approximately  $4\text{--}11\times$  fewer function evaluations (NFE), while maintaining sample entropy close to the data. Generation is  $10\text{--}21\times$  faster on LM1B and OpenWebText. On conditional generation, **NCM** uses approximately  $6\text{--}16\times$  fewer NFE than unguided diffusion baselines on WMT14 translation and XSum summarization, achieving competitive translation quality and the best summarization scores. On XSum, it runs  $31\times$  faster than the strongest diffusion baseline. It also uses approximately  $4\text{--}5\times$  fewer training tokens than strong diffusion baselines on LM1B and OpenWebText, including both tokenizer and generator training. Finally, preliminary scaling results on LM1B show that increasing the generator size improves generation quality at a fixed training-token budget.

## Acknowledgments

We thank Yasa Baig, Dan Biderman, Bradley Brown, Francois Chaubard, Mayee Chen, Vivien Cheng, Catherine Deng, Owen Dugan, Sabri Eyuboglu, Simon Guo, Neel Guha, Junmiao Hu, Hangoo Kang, Jacky Kwok, Jerry Liu, Avanika Narayan, Divya Nori, Jon Saad-Falcon, Ben Spector, Stuart Sul, Tarun Suresh, Shayan Talaei, Roberto Torres, Ateh Wangia, and Michael Zhang for their helpful feedback and discussions.

We gratefully acknowledge the support of NIH under Nos. U54EB020405 (Mobilize) and R01GM124443 (SimTK); NSF under Nos. CCF2247015 (Hardware-Aware), CCF1763315 (Beyond Sparsity), CCF1563078 (Volume to Velocity), 1937301 (RTML), 1900638 (Open Federated Virtual Assistants), and 2028602 (PPoSS); DARPA under No. HR00112520038 (Fallingwater); US DEVCOM ARL under Nos. W911NF-23-2-0184 (Long-context) and W911NF-21-2-0251 (Interactive Human-AI Teaming); NSTXL under No. N00164-23-9-G057-01 (Energy-Efficient AI Hardware); ONR under Nos. N000142312633 (Deep Signal Processing), N000142212426 (Misinformation Analysis), N000142012480 (Non-Euclidean Geometry), and N000142012275 (Data Programming); Stanford HAI under Nos. 247183, 215955, and 345077 (Evo); Google DeepMind; Google Research; Google Cloud; NXP, Xilinx, LETI-CEA, Intel, IBM, Microsoft, NEC, Toshiba, TSMC, ARM, Hitachi, BASF, Accenture, Ericsson, Qualcomm, Analog Devices, Salesforce, Total, the Laude Institute, Prime Intellect, Anthropic, the HAI-GCP Cloud Credits for Research program, the Stanford Data Science Initiative (SDSI), the Stanford Marlowe Computing Platform, Lambda Labs, Modal, Zoltan’s Flops, and members of the Stanford DAWN project: Meta, Google, and VMWare; and members of the Stanford SEAMS project: IBM and Felicis. HK is supported by the Knight Hennessy Scholars Program. IK is supported in part by the NSF GRFP. The U.S. Government is authorized to reproduce and distribute reprints for Governmental purposes notwithstanding any copyright notation thereon. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the views, policies, or endorsements, either expressed or implied, of NIH, ONR, DARPA, or the U.S Government.

## References

- [1] Jacob Austin, Daniel D. Johnson, Jonathan Ho, Daniel Tarlow, and Rianne van den Berg. Structured denoising diffusion models in discrete state-spaces. In A. Beygelzimer, Y. Dauphin, P. Liang, and J. Wortman Vaughan, editors, *Advances in Neural Information Processing Systems*, 2021.
- [2] Aaron Lou, Chenlin Meng, and Stefano Ermon. Discrete diffusion modeling by estimating the ratios of the data distribution. arXiv preprint arXiv:2310.16834, 2024.
- [3] Subham Sekhar Sahoo, Marianne Arriola, Yair Schiff, Aaron Gokaslan, Edgar Marroquin, Justin T Chiu, Alexander Rush, and Volodymyr Kuleshov. Simple and effective masked diffusion language models. arXiv preprint arXiv:2406.07524, 2024.
- [4] Woosuk Kwon, Zhuohan Li, Siyuan Zhuang, Ying Sheng, Lianmin Zheng, Cody Hao Yu, Joseph E. Gonzalez, Hao Zhang, and Ion Stoica. Efficient memory management for large language model serving with PagedAttention. arXiv preprint arXiv:2309.06180, 2023.
- [5] Reiner Pope, Sholto Douglas, Aakanksha Chowdhery, Jacob Devlin, James Bradbury, Anselm Levskaya, Jonathan Heek, Kefan Xiao, Shivani Agrawal, and Jeff Dean. Efficiently scaling transformer inference. arXiv preprint arXiv:2211.05102, 2022.
- [6] 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. arXiv preprint arXiv:2502.09992, 2025.
- [7] Inception Labs, Samar Khanna, Siddhant Kharbanda, Shufan Li, Harshit Varma, Eric Wang, Sawyer Birnbaum, Ziyang Luo, Yanis Miraoui, Akash Palrecha, Stefano Ermon, Aditya Grover, and Volodymyr Kuleshov. Mercury: Ultra-fast language models based on diffusion. arXiv preprint arXiv:2506.17298, 2025.
- [8] DiffusionGemma Team, Adrien Ali Taïga, James Assiene, Daniele Calandriello, Rahma Chaabouni, João Gante, Tamara von Glehn, Nate Keating, Chris Knutsen, Martin Kukla, et al. DiffusionGemma technical report. *arXiv preprint arXiv:2608.00146*, 2026.
- [9] Jian Chen, Yesheng Liang, and Zhijian Liu. DFlash: Block diffusion for flash speculative decoding. *arXiv preprint arXiv:2602.06036*, 2026.
- [10] 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 *International Conference on Learning Representations*, volume 2026, pages 57027–57051, 2026.
- [11] Xinyin Ma, Runpeng Yu, Gongfan Fang, and Xinchao Wang. dKV-Cache: The cache for diffusion language models. *Advances in Neural Information Processing Systems*, 38:149009–149033, 2026.
- [12] Subham Sekhar Sahoo, Justin Deschenaux, Aaron Gokaslan, Guanghan Wang, Justin T Chiu, and Volodymyr Kuleshov. The diffusion duality. In *Forty-second International Conference on Machine Learning*, 2025.
- [13] Keya Hu, Linlu Qiu, Yiyang Lu, Hanhong Zhao, Tianhong Li, Yoon Kim, Jacob Andreas, and Kaiming He. ELF: Embedded language flows. *arXiv preprint arXiv:2605.10938*, 2026.

- [14] Justin Deschenaux and Caglar Gulcehre. Beyond autoregression: Fast LLMs via self-distillation through time. In *International Conference on Learning Representations*, volume 2025, pages 5007–5045, 2025.
- [15] Chanhuk Lee, Jaehoon Yoo, Manan Agarwal, Sheel Shah, Jerry Huang, Aditi Raghunathan, Seunghoon Hong, Nicholas M. Boffi, and Jinwoo Kim. Flow map language models: One-step language modeling via continuous denoising. *arXiv preprint arXiv:2602.16813*, 2026.
- [16] Aaron Gokaslan and Vanya Cohen. OpenWebText corpus. <http://Skylion007.github.io/OpenWebTextCorpus>, 2019.
- [17] Ciprian Chelba, Tomas Mikolov, Mike Schuster, Qi Ge, Thorsten Brants, Phillipp Koehn, and Tony Robinson. One billion word benchmark for measuring progress in statistical language modeling. *arXiv preprint arXiv:1312.3005*, 2013.
- [18] Marianne Arriola, Aaron Gokaslan, Justin T. Chiu, Zhihan Yang, Zhixuan Qi, Jiaqi Han, Subham Sekhar Sahoo, and Volodymyr Kuleshov. Block diffusion: Interpolating between autoregressive and diffusion language models. In *The Thirteenth International Conference on Learning Representations*, 2025. URL <https://arxiv.org/abs/2503.09573>.
- [19] Chengyue Wu, Hao Zhang, Shuchen Xue, Shizhe Diao, Yonggan Fu, Zhijian Liu, Pavlo Molchanov, Ping Luo, Song Han, and Enze Xie. Fast-dLLM v2: Efficient block-diffusion LLM. In *The Fourteenth International Conference on Learning Representations*, 2026. URL <https://arxiv.org/abs/2509.26328>.
- [20] Aaron van den Oord, Oriol Vinyals, and Koray Kavukcuoglu. Neural discrete representation learning. *arXiv preprint arXiv:1711.00937*, 2018.
- [21] Keyu Tian, Yi Jiang, Zehuan Yuan, Bingyue Peng, and Liwei Wang. Visual autoregressive modeling: Scalable image generation via next-scale prediction. *Advances in neural information processing systems*, 37:84839–84865, 2024.
- [22] Hermann Kumbong, Xian Liu, Tsung-Yi Lin, Xihui Liu, Ziwei Liu, Daniel Y Fu, Ming-Yu Liu, Christopher Re, and David W Romero. HMAR: Efficient hierarchical masked autoregressive image generation. *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2025.
- [23] Sander Dieleman, Laurent Sartran, Arman Roshannai, Nikolay Savinov, Yaroslav Ganin, Pierre H. Richemond, Arnaud Doucet, Robin Strudel, Chris Dyer, Conor Durkan, Curtis Hawthorne, Rémi Leblond, Will Grathwohl, and Jonas Adler. Continuous diffusion for categorical data, 2022. URL <https://arxiv.org/abs/2211.15089>.
- [24] Patrick Pynadath, Jiaxin Shi, and Ruqi Zhang. CANDI: Hybrid discrete-continuous diffusion models. *arXiv preprint arXiv:2510.22510*, 2025.
- [25] Jiajun Liang, Yucheng Liao, Yukang Cao, Jiazhe Wei, Ken Li, Wende Tan, Jiankun Zhang, ZY Cui, Jingkan Yang, Liucheng Guo, et al. AURORA-LM: Autoencoding unified representation for continuous-latent diffusion language modeling. *arXiv preprint arXiv:2608.02602*, 2026.
- [26] Ali Razavi, Aaron van den Oord, and Oriol Vinyals. Generating diverse high-fidelity images with VQ-VAE-2. *arXiv preprint arXiv:1906.00446*, 2019.

- [27] Patrick Esser, Robin Rombach, and Björn Ommer. Taming transformers for high-resolution image synthesis. *arXiv preprint arXiv:2012.09841*, 2021.
- [28] Doyup Lee, Chiheon Kim, Saehoon Kim, Minsu Cho, and Wook-Shin Han. Autoregressive image generation using residual quantization. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 11523–11532, 2022.
- [29] Neil Zeghidour, Alejandro Luebs, Ahmed Omran, Jan Skoglund, and Marco Tagliasacchi. SoundStream: An end-to-end neural audio codec. *arXiv preprint arXiv:2107.03312*, 2021.
- [30] Alexandre Défossez, Jade Copet, Gabriel Synnaeve, and Yossi Adi. High fidelity neural audio compression. *Transactions on Machine Learning Research*, 2023. ISSN 2835-8856. Featured Certification, Reproducibility Certification.
- [31] Justin Lovelace, Varsha Kishore, Chao Wan, Eliot Shekhtman, and Kilian Q. Weinberger. Latent diffusion for language generation. In *Advances in Neural Information Processing Systems*, volume 36, 2023.
- [32] Yizhe Zhang, Jiatao Gu, Zhuofeng Wu, Shuangfei Zhai, Joshua Susskind, and Navdeep Jaitly. PLANNER: Generating diversified paragraph via latent language diffusion model. In *Advances in Neural Information Processing Systems*, volume 36, 2023.
- [33] Lili Yu, Dániel Simig, Colin Flaherty, Armen Aghajanyan, Luke Zettlemoyer, and Mike Lewis. MEGABYTE: Predicting million-byte sequences with multiscale transformers. *Advances in Neural Information Processing Systems*, 36:78808–78823, 2023.
- [34] Sukjun Hwang, Brandon Wang, and Albert Gu. Dynamic chunking for end-to-end hierarchical sequence modeling. In *International Conference on Learning Representations*, volume 2026, pages 149273–149313, 2026.
- [35] Loïc Barrault, Paul-Ambroise Duquenne, Maha Elbayad, Artyom Kozhevnikov, Belen Alastruey, Pierre Andrews, Mariano Coria, Guillaume Couairon, Marta R Costa-jussà, David Dale, et al. Large concept models: Language modeling in a sentence representation space. *arXiv preprint arXiv:2412.08821*, 2024.
- [36] Yulian Liu, Yunchong Song, Yixuan Wang, Kewen Ge, Alex Lamb, Qipeng Guo, Kai Chen, Bowen Zhou, and Zhouhan Lin. Next concept prediction in discrete latent space leads to stronger language models. *arXiv preprint arXiv:2602.08984*, 2026.
- [37] Xingwei Qu, Shaowen Wang, Zihao Huang, Kai Hua, Fan Yin, Rui-Jie Zhu, Jundong Zhou, Qiyang Min, Zihao Wang, Yizhi Li, et al. Dynamic large concept models: Latent reasoning in an adaptive semantic space. *arXiv preprint arXiv:2512.24617*, 2025.
- [38] Cai Zhou, Chenyu Wang, Dinghuai Zhang, Shangyuan Tong, Yifei Wang, Stephen Bates, and Tommi Jaakkola. Next semantic scale prediction via hierarchical diffusion language models. *Advances in Neural Information Processing Systems*, 38:41496–41531, 2026.
- [39] Xiang Li, Kai Qiu, Hao Chen, Jason Kuen, Jiuxiang Gu, Bhiksha Raj, and Zhe Lin. ImageFolder: Autoregressive image generation with folded tokens. *arXiv preprint arXiv:2410.01756*, 2024.
- [40] Ronen Eldan and Yuanzhi Li. Tinystories: How small can language models be and still speak coherent english? *arXiv preprint arXiv:2305.07759*, 2023.

- [41] Ondřej Bojar, Christian Buck, Christian Federmann, Barry Haddow, Philipp Koehn, Johannes Leveling, Christof Monz, Pavel Pecina, Matt Post, Herve Saint-Amand, Radu Soricut, Lucia Specia, and Aleš Tamchyna. Findings of the 2014 workshop on statistical machine translation. In Ondřej Bojar, Christian Buck, Christian Federmann, Barry Haddow, Philipp Koehn, Christof Monz, Matt Post, and Lucia Specia, editors, *Proceedings of the Ninth Workshop on Statistical Machine Translation*, pages 12–58, Baltimore, Maryland, USA, June 2014. Association for Computational Linguistics. doi: 10.3115/v1/W14-3302.
- [42] Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. BLEU: a method for automatic evaluation of machine translation. In *Proceedings of the 40th annual meeting of the Association for Computational Linguistics*, pages 311–318, 2002.
- [43] Matt Post. A call for clarity in reporting BLEU scores. In Ondřej Bojar, Rajen Chatterjee, Christian Federmann, Mark Fishel, Yvette Graham, Barry Haddow, Matthias Huck, Antonio Jimeno Yepes, Philipp Koehn, Christof Monz, Matteo Negri, Aurélie Névél, Mariana Neves, Matt Post, Lucia Specia, Marco Turchi, and Karin Verspoor, editors, *Proceedings of the Third Conference on Machine Translation: Research Papers*, pages 186–191, Brussels, Belgium, October 2018. Association for Computational Linguistics. doi: 10.18653/v1/W18-6319.
- [44] Shashi Narayan, Shay B. Cohen, and Mirella Lapata. Don’t give me the details, just the summary! Topic-aware convolutional neural networks for extreme summarization. In Ellen Riloff, David Chiang, Julia Hockenmaier, and Jun’ichi Tsujii, editors, *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 1797–1807, Brussels, Belgium, October–November 2018. Association for Computational Linguistics. doi: 10.18653/v1/D18-1206.
- [45] Chin-Yew Lin. ROUGE: A package for automatic evaluation of summaries. In *Text Summarization Branches Out*, pages 74–81, Barcelona, Spain, July 2004. Association for Computational Linguistics.
- [46] Kaiwen Zheng, Yongxin Chen, Hanzi Mao, Ming-Yu Liu, Jun Zhu, and Qinsheng Zhang. Masked diffusion models are secretly time-agnostic masked models and exploit inaccurate categorical sampling. In *International Conference on Learning Representations*, 2025. URL <https://arxiv.org/abs/2409.02908>.
- [47] Marianne Arriola, Yair Schiff, Hao Phung, Aaron Gokaslan, and Volodymyr Kuleshov. Encoder-decoder diffusion language models for efficient training and inference. In *The Thirty-ninth Annual Conference on Neural Information Processing Systems*, 2025.
- [48] Hongyi Yuan, Zheng Yuan, Chuanqi Tan, Fei Huang, and Songfang Huang. SeqDiffuSeq: Text diffusion with encoder-decoder transformers. arXiv preprint arXiv:2212.10325, 2023.
- [49] Manan Agarwal, Sheel Shah, Chanhuk Lee, Jaehoon Yoo, Jerry Huang, Seunghoon Hong, Aditi Raghunathan, Jinwoo Kim, and Nicholas M Boffi. Posterior refinement: Fast language generation via any-order flow maps. *arXiv preprint arXiv:2606.24773*, 2026.
- [50] Justin Deschenaux and Caglar Gulcehre. Language modeling with hyperspherical flows. *arXiv preprint arXiv:2605.11125*, 2026.
- [51] Amanda Bertsch, Alex Xie, Graham Neubig, and Matthew R Gormley. It’s MBR all the way down: Modern generation techniques through the lens of minimum Bayes risk. In *Proceedings of the Big Picture Workshop*, pages 108–122, 2023.

- [52] Subham Sekhar Sahoo, Zhihan Yang, Yash Akhauri, Johnna Liu, Deepansha Singh, Zhoujun Cheng, Zhengzhong Liu, Eric Xing, John Thickstun, and Arash Vahdat. Esoteric language models: A family of any-order diffusion LLMs. *arXiv preprint arXiv:2506.01928*, 2025.

# Appendix Contents

|                                                                  |           |
|------------------------------------------------------------------|-----------|
| <b>A Probabilistic Formulation and ELBO</b>                      | <b>21</b> |
| A.1 Setup . . . . .                                              | 21        |
| A.2 Evidence Lower Bound . . . . .                               | 21        |
| <b>B Datasets</b>                                                | <b>22</b> |
| <b>C Architecture Details</b>                                    | <b>22</b> |
| C.1 Multiscale VQVAE Tokenizer . . . . .                         | 23        |
| C.2 Generator (NCM) . . . . .                                    | 24        |
| <b>D Baselines</b>                                               | <b>24</b> |
| D.1 Unconditional Generation . . . . .                           | 24        |
| D.2 Conditional Generation . . . . .                             | 25        |
| D.3 Reasoning . . . . .                                          | 25        |
| <b>E Additional Results</b>                                      | <b>25</b> |
| E.1 Generative Perplexity Under Other Reference Models . . . . . | 25        |
| E.2 Unconditional Generation . . . . .                           | 26        |
| <b>F Ablations</b>                                               | <b>26</b> |
| F.1 Latent Code Hierarchy . . . . .                              | 26        |
| F.2 NCM Transformer . . . . .                                    | 28        |
| F.3 Code Corruption . . . . .                                    | 29        |
| F.4 Sampling . . . . .                                           | 30        |
| <b>G Tokenizer Analysis</b>                                      | <b>31</b> |
| <b>H Probing the Hierarchy</b>                                   | <b>32</b> |
| <b>I Limitations &amp; Future Work</b>                           | <b>35</b> |
| <b>J Additional Qualitative Samples</b>                          | <b>35</b> |
| J.1 WMT14 De–En . . . . .                                        | 35        |
| J.2 Generation Trajectory . . . . .                              | 36        |

|     |                       |    |
|-----|-----------------------|----|
| J.3 | XSum . . . . .        | 36 |
| J.4 | TinyStories . . . . . | 37 |
| J.5 | LM1B . . . . .        | 38 |
| J.6 | OpenWebText . . . . . | 38 |
| J.7 | Sudoku . . . . .      | 40 |

## A Probabilistic Formulation and ELBO

We formulate **NCM** as a hierarchical latent-variable model and derive an evidence lower bound (ELBO) on its text log-likelihood. We consider unconditional generation with deterministic nearest-neighbor code assignments.

### A.1 Setup

Following Section 3, let  $x \in \mathcal{T}^L$  be a sequence of  $L$  tokens from the text vocabulary  $\mathcal{T}$ . The tokenizer assigns  $x$  a hierarchy  $r_{1:K}^*(x) = (r_1^*, \dots, r_K^*)$ , where  $r_k^* \in \{1, \dots, |\mathcal{V}_k|\}^{\ell_k}$  and  $\ell_1 < \dots < \ell_K = L$ . We fix the tokenizer’s parameters and use a fixed rule to break nearest-neighbor ties, so this assignment is deterministic. For any code hierarchy  $r_{1:K}$ , define its cumulative reconstruction by

$$\hat{z}^{(k)}(r_{1:k}) = \sum_{j=1}^k R_j(e_j(r_j)), \quad \hat{z}^{(0)} = 0, \quad (11)$$

where  $e_j$  is the codebook lookup and  $R_j$  is the reconstruction operator of Section 3.3. The decoder defines a likelihood over text,

$$p_\psi(x \mid r_{1:K}) = \prod_{n=1}^L p_\psi(x^n \mid \hat{z}^{(K)}(r_{1:K})), \quad (12)$$

and the generator defines a prior over hierarchies,

$$p_\theta(r_{1:K}) = \prod_{k=1}^K p_\theta(r_k \mid \hat{z}^{(k-1)}) = \prod_{k=1}^K \prod_{i=1}^{\ell_k} p_\theta(r_k^i \mid \hat{z}^{(k-1)}), \quad (13)$$

where  $\hat{z}^{(k-1)} = \hat{z}^{(k-1)}(r_{1:k-1})$ . Each factor is normalized for every preceding hierarchy, so their product defines a normalized prior. The joint distribution and marginal text likelihood are

$$p_{\theta,\psi}(x, r_{1:K}) = p_\psi(x \mid r_{1:K}) p_\theta(r_{1:K}), \quad p_{\theta,\psi}(x) = \sum_{r_{1:K}} p_\psi(x \mid r_{1:K}) p_\theta(r_{1:K}). \quad (14)$$

### A.2 Evidence Lower Bound

Use the tokenizer’s deterministic assignment as a point-mass variational distribution,

$$q(r_{1:K} \mid x) = \mathbf{1}\{r_{1:K} = r_{1:K}^*(x)\}. \quad (15)$$

This distribution has zero entropy. Its ELBO therefore reduces to the log joint probability of the tokenizer’s assigned hierarchy:

$$\begin{aligned} \mathcal{L}_{\text{ELBO}}(x) &\equiv \mathbb{E}_{q(r_{1:K} \mid x)} [\log p_{\theta,\psi}(x, r_{1:K}) - \log q(r_{1:K} \mid x)] \\ &= \log p_\psi(x \mid r_{1:K}^*) + \log p_\theta(r_{1:K}^*) \\ &\leq \log \sum_{r_{1:K}} p_\psi(x \mid r_{1:K}) p_\theta(r_{1:K}) = \log p_{\theta,\psi}(x). \end{aligned} \quad (16)$$

The inequality follows because the marginal likelihood sums nonnegative terms and includes the term for  $r_{1:K}^*$ . Writing  $\hat{z}_*^{(k)} = \hat{z}^{(k)}(r_{1:k}^*)$  and substituting (12) and (13) gives

$$\boxed{\mathcal{L}_{\text{ELBO}}(x) = \underbrace{\sum_{n=1}^L \log p_\psi(x^n \mid \hat{z}_*^{(K)})}_{\text{reconstruction}} + \underbrace{\sum_{k=1}^K \sum_{i=1}^{\ell_k} \log p_\theta(r_k^{*i} \mid \hat{z}_*^{(k-1)})}_{\text{generation}}} \quad (17)$$

where  $r_k^{*i}$  is the  $i$ th code at level  $k$  in the assigned hierarchy. The gap to the log-likelihood is exactly

$$\begin{aligned}\log p_{\theta,\psi}(x) - \mathcal{L}_{\text{ELBO}}(x) &= D_{\text{KL}}(q(r_{1:K} \mid x) \parallel p_{\theta,\psi}(r_{1:K} \mid x)) \\ &= -\log p_{\theta,\psi}(r_{1:K}^* \mid x) \geq 0.\end{aligned}\tag{18}$$

Thus, the bound is tight if and only if the posterior assigns probability one to the tokenizer’s hierarchy. With finite softmax logits, the prior and decoder assign positive probability to every hierarchy and text sequence, so the bound and gap are finite.

For a frozen tokenizer and decoder, only the generation term depends on  $\theta$ . With clean conditioning and  $\alpha = 0$ , the generator loss in Section 3.4 is the negative generation term divided by the total number of codes  $N = \sum_{k=1}^K \ell_k$ ; minimizing it therefore maximizes this ELBO over  $\theta$ . Conditioning corruption and nonuniform level weighting modify that objective. The bound uses the complete hierarchy; nested dropout and the quantization losses used to train the tokenizer are not included in it.

## B Datasets

We train a separate tokenizer and generator for each of six datasets, grouped by task as in Section 4. Table 4 summarizes them.

| Dataset                      | Abbreviation | Task          | Text tokenizer           | Context $c$   | Target $L$ |
|------------------------------|--------------|---------------|--------------------------|---------------|------------|
| OpenWebText [16]             | OWT          | Unconditional | GPT-2                    | Null prefix   | 1,024      |
| One Billion Word [17]        | LM1B         | Unconditional | <b>bert-base-uncased</b> | Null prefix   | 128        |
| TinyStories [40]             | –            | Unconditional | Custom BPE               | Null prefix   | 128        |
| Extreme Summarization [44]   | XSum         | Summarization | GPT-2                    | Article (512) | 48         |
| WMT 2014 German–English [41] | WMT14        | Translation   | <b>Qwen3-0.6B</b>        | German (96)   | 64         |
| Sudoku [50]                  | –            | Reasoning     | FMLM+ [49]               | Puzzle        | 89         |

Table 4: **Datasets.** Target length  $L$  is the number of target tokens, which equals the length  $\ell_K$  of the finest level; context lengths are in tokens. “–” indicates that we use the full dataset name rather than a separate abbreviation.

## C Architecture Details

A tokenizer and a generator are trained for each dataset (Appendix B): the tokenizer first, and it is then frozen as the generator trains. For XSum and WMT14, the generator is initially trained with unpaired text, then with source–target pairs. The target length schedule and level schedule remain unchanged. Table 5 describes the tokenizers’ architectures and training. Table 6 provides details about the generators’ architectures, training, and sampling. The datasets are listed in the same order as in Table 1. All experiments employed seed 42, BF16 mixed-precision and `torch.compile`.

## C.1 Multiscale VQVAE Tokenizer

| Setting                | OWT                | LM1B               | TinyStories        | XSum               | WMT14              | Sudoku             |
|------------------------|--------------------|--------------------|--------------------|--------------------|--------------------|--------------------|
| <i>Architecture</i>    |                    |                    |                    |                    |                    |                    |
| Text vocab. size       | 50,304             | 30,528             | 8,192              | 50,304             | 151,680            | 12                 |
| Target length $L$      | 1,024              | 128                | 128                | 48                 | 64                 | 89                 |
| Levels $K$             | 32                 | 12                 | 12                 | 8                  | 11                 | 12                 |
| Codebook vocab. $V_k$  | 4,096              | 8,192              | 4,096              | 4,096              | 4,096              | 16                 |
| Latent width $d$       | 384                | 384                | 384                | 384                | 384                | 256                |
| Encoder blocks         | 0                  | 0                  | 0                  | 0                  | 0                  | 0                  |
| Decoder blocks         | 1                  | 4                  | 1                  | 1                  | 1                  | 1                  |
| Decoder attn. heads    | 6                  | 6                  | 6                  | 6                  | 6                  | 4                  |
| Dropout                | 0.3                | 0.1                | 0.3                | 0.1                | 0.1                | 0.3                |
| Parameters             | 111.2M             | 111.40M            | 36.43M             | 60.63M             | 144.00M            | 14.90M             |
| <i>Training</i>        |                    |                    |                    |                    |                    |                    |
| Training updates       | 78,432             | 25,957             | 30,000             | 120,001            | 60,000             | 10,000             |
| Global batch size      | 768                | 2,048              | 4,096              | 4,096              | 4,096              | 1,024              |
| Peak learning rate     | $3 \times 10^{-4}$ | $3 \times 10^{-4}$ | $5 \times 10^{-4}$ | $5 \times 10^{-4}$ | $5 \times 10^{-4}$ | $5 \times 10^{-4}$ |
| Learning-rate floor    | $3 \times 10^{-5}$ | $3 \times 10^{-5}$ | $5 \times 10^{-5}$ | $5 \times 10^{-5}$ | $5 \times 10^{-5}$ | $5 \times 10^{-5}$ |
| Warmup updates         | 1,000              | 1,000              | 1,000              | 1,000              | 1,000              | 500                |
| AdamW $\beta_1$        | 0.9                | 0.9                | 0.9                | 0.9                | 0.9                | 0.9                |
| AdamW $\beta_2$        | 0.95               | 0.99               | 0.95               | 0.95               | 0.95               | 0.99               |
| Weight decay           | 0.01               | 0.05               | 0.01               | 0.01               | 0.01               | 0.01               |
| Gradient clipping norm | 1                  | 1                  | 1                  | 1                  | 1                  | 1                  |
| Codebook EMA decay     | 0.99               | 0.99               | 0.95               | 0.95               | 0.95               | 0.99               |
| Nested dropout         | 0.5                | 0.7 / 0.3          | 0.5                | 0.5                | 0.5                | 0.5                |

Table 5: **Tokenizers.** Text vocabulary size includes unused padding rows. Nested dropout is the probability of dropping the finest levels; LM1B trains on the complete and a truncated hierarchy, weighted 0.7 and 0.3.

## C.2 Generator (NCM)

| Setting                | OWT                | LM1B               | TinyStories        | XSum               | WMT14              | Sudoku             |
|------------------------|--------------------|--------------------|--------------------|--------------------|--------------------|--------------------|
| <i>Architecture</i>    |                    |                    |                    |                    |                    |                    |
| Parameters             | 94.65M             | 97.07M             | 88.61M             | 88.02M             | 119.96M            | 25.46M             |
| Transformer blocks     | 12                 | 12                 | 12                 | 12                 | 12                 | 8                  |
| Hidden width           | 768                | 768                | 768                | 768                | 768                | 512                |
| Attention heads        | 12                 | 12                 | 12                 | 12                 | 12                 | 8                  |
| Prefix length $P$      | 1                  | 1                  | 1                  | 512                | 96                 | 91                 |
| RoPE base              | $10^5$             | $10^4$             | $10^5$             | $10^5$             | $10^5$             | $10^4$             |
| Dropout                | 0.1                | 0.1                | 0.1                | 0.3                | 0.1                | 0.1                |
| <i>Training</i>        |                    |                    |                    |                    |                    |                    |
| Training updates       | 250,000            | 100,000            | 50,000             | 350,000            | 150,000            | 50,000             |
| Global batch size      | 256                | 512                | 2,048              | 512                | 2,048              | 1,024              |
| Peak learning rate     | $3 \times 10^{-4}$ | $3 \times 10^{-4}$ | $3 \times 10^{-4}$ | $3 \times 10^{-4}$ | $3 \times 10^{-4}$ | $3 \times 10^{-4}$ |
| Learning-rate floor    | $3 \times 10^{-5}$ | $3 \times 10^{-5}$ | $3 \times 10^{-5}$ | $3 \times 10^{-5}$ | $3 \times 10^{-5}$ | $3 \times 10^{-5}$ |
| Warmup updates         | 500                | 1,000              | 500                | 500                | 500                | 1,000              |
| AdamW $\beta_1$        | 0.9                | 0.9                | 0.9                | 0.9                | 0.9                | 0.9                |
| AdamW $\beta_2$        | 0.95               | 0.95               | 0.95               | 0.95               | 0.95               | 0.99               |
| Weight decay           | 0.1                | 0.1                | 0.1                | 0.1                | 0.01               | 0.01               |
| Gradient clipping norm | 1                  | 1                  | 1                  | 1                  | 1                  | 1                  |

Table 6: **Generators.** The prefix holds the source (XSum, WMT14), the puzzle (Sudoku), or a null token. Training settings for XSum and WMT14 describe conditional training. “–” means not applicable.

**Training-token accounting.** Budgets count full-length text positions, including padding and repeated data exposure. For LM1B, tokenizer and generator training total  $25,957 \times 2,048 \times 128 + 100,000 \times 512 \times 128 \approx 13.36\text{B}$  positions (6.80B+6.55B). For OpenWebText, the corresponding total is  $78,432 \times 768 \times 1,024 + 250,000 \times 256 \times 1,024 \approx 127.22\text{B}$  positions (61.68B+65.54B), approximately  $4\times$  fewer than the 500B-plus tokens used by MDLM and DUO [3, 12]. Scaling plots exclude the shared tokenizer. These budgets measure token exposure, not FLOPs.

Each generator block contains self-attention followed by a feed-forward layer. The resized reconstruction  $\hat{z}^{(k-1)}$  and prompt embeddings are projected to the hidden width before the first block. Attention heads have dimension 64, equal to the hidden width divided by the number of heads.

## D Baselines

This appendix states where every baseline number in Section 4 comes from and how the baselines were sampled.

### D.1 Unconditional Generation

**LM1B.** All LM1B baselines (RDLM, CANDI, MDLM, DUO, and FLM) use the checkpoints of Lee et al. [15], who trained them for the same 1M iterations on 66B tokens. We sample each with  $T \in \{8, 12, 16, 32, 64, 128\}$  steps under the protocol of Section 4.2. RDLM is omitted from Figure 5 because its Gen. PPL lies far above the other curves.

**TinyStories.** We train the AR, DUO, MDLM, and CANDI baselines ourselves using the same transformer architecture and training settings as the TinyStories **NCM** generator (Table 6). For FMLM+, we use the publicly released checkpoint [49].

**OpenWebText.** The MDLM, DUO, SEDD, and Eso-LM baselines use publicly available checkpoints [3, 12, 2, 52]. We sample each with  $T \in \{8, 16, \dots, 1,024\}$  steps under the same protocol.

## D.2 Conditional Generation

The MDLM, DUO, E2D2, and SeqDiffuSeq results of Table 2 are taken from Hu et al. [13] (marked <sup>†</sup>), with their configurations. We use publicly available checkpoints of ELF (with and without guidance) and AURORA-LM [25] to reproduce their quality results and measure sampling time under the setup of Section 4.6. For conditional baselines without released checkpoints, we benchmark sampling time using randomly initialized weights with the same model architecture and task-specific sampling configuration reported for those tasks. These runs are used only for timing; their quality results are taken from the published results.

## D.3 Reasoning

We follow the Sudoku evaluation setup of S-FLM [50], reporting the percentage of completely solved boards at three difficulty levels: easy, medium, and hard, with 40, 35, and 30 provided clues, respectively. The baseline results in Table 3 are taken from Agarwal et al. [49].

# E Additional Results

This appendix checks that the LM1B comparison does not depend on the scoring model (Appendix E.1) and collects unconditional generation results from Section 4 (Appendix E.2).

## E.1 Generative Perplexity Under Other Reference Models

To check whether the LM1B comparison depends on the reference model, we evaluate the same checkpoints and sampling settings as Figure 5 under four additional autoregressive models. Table 7 averages five sampling seeds with 256 samples each. **NCM** has the lowest average Gen. PPL, using approximately  $5\times$  fewer training tokens and 12 sampling steps rather than 128.

| Model      | Size | Tokens | Mistral-7B ↓ | Llama-3.1-8B ↓ | Gemma-2-9B ↓  | Qwen-3-14B ↓ | Avg. ↓       |
|------------|------|--------|--------------|----------------|---------------|--------------|--------------|
| RDLM       | 108M | 66B    | 179.54       | 315.40         | 1293.67       | 379.00       | 541.90       |
| CANDI      | 139M | 66B    | 67.75        | 99.96          | 208.59        | 125.87       | 125.54       |
| MDLM       | 139M | 66B    | 63.63        | 91.84          | 186.53        | 115.25       | 114.31       |
| DUO        | 139M | 66B    | <b>55.08</b> | <u>77.24</u>   | <u>141.16</u> | <u>96.59</u> | <u>92.52</u> |
| FLM        | 139M | 66B    | 68.08        | 89.20          | 150.11        | 110.98       | 104.59       |
| <b>NCM</b> | 97M  | 13.36B | <u>59.69</u> | <b>75.23</b>   | <b>106.41</b> | <b>91.64</b> | <b>83.24</b> |

Table 7: **LM1B Gen. PPL** (lower is better). Baselines use 128 steps; **NCM** uses 12. Size: generator parameters; Tokens: training budget (including tokenizer training for **NCM**); Avg.: unweighted mean.

## E.2 Unconditional Generation

Table 8 collects the unconditional results of all three datasets at the largest NFE reported for each baseline. We report generative perplexity under GPT-2-large alongside mean per-sample unigram entropy  $H$  to assess sample quality and diversity together. The ground-truth data are included as a reference for both metrics.

| Type  | Method | TinyStories |           |      | LM1B |           |      | OpenWebText |           |      |
|-------|--------|-------------|-----------|------|------|-----------|------|-------------|-----------|------|
|       |        | NFE↓        | Gen. PPL↓ | $H$  | NFE↓ | Gen. PPL↓ | $H$  | NFE↓        | Gen. PPL↓ | $H$  |
| –     | Data   | –           | 9.67      | 3.96 | –    | 53.51     | 4.31 | –           | 14.15     | 5.44 |
| AR    | Sample | 128         | 8.89      | 4.01 | –    | –         | –    | 1024        | 35.45     | 5.58 |
|       | Greedy | 128         | 5.34      | 4.06 | –    | –         | –    | 1024        | 1.07      | 1.65 |
| Disc. | MDLM   | 128         | 17.89     | 4.03 | 128  | 115.45    | 4.33 | 1024        | 105.15    | 5.63 |
|       | Duo    | 128         | 19.63     | 4.01 | 128  | 97.30     | 4.30 | 1024        | 77.69     | 5.55 |
|       | SEDD   | –           | –         | –    | –    | –         | –    | 1024        | 101.43    | 5.61 |
|       | Eso-LM | –           | –         | –    | –    | –         | –    | 1024        | 131.96    | 5.65 |
| Cont. | RDLM   | –           | –         | –    | 128  | 295.40    | 4.35 | –           | –         | –    |
|       | CANDI  | 128         | 22.74     | 4.03 | 128  | 124.64    | 4.34 | 1024        | 143.13    | 5.71 |
|       | FLM    | 128         | 57.50     | 4.17 | 128  | 112.33    | 4.33 | 1024        | 62.23     | 5.33 |
|       | S-FLM  | 128         | 95.25     | 4.08 | –    | –         | –    | 1024        | 123.87    | 5.52 |
|       | FMLM+  | 128         | 15.43     | 3.90 | –    | –         | –    | 128         | 66.60     | 5.21 |
| Ours  | NCM    | 12          | 15.41     | 3.98 | 12   | 93.05     | 4.31 | 32          | 83.12     | 5.42 |

Table 8: **Unconditional generation on TinyStories, LM1B, and OpenWebText.** Gen. PPL under GPT-2-large and mean per-sample unigram entropy  $H$ ; the Data row scores ground-truth text; “–” means no result on that dataset. Lower Gen. PPL at an entropy below that of the data, as for FLM and FMLM+ on OpenWebText, partly reflects reduced diversity.

## F Ablations

We ablate the key design choices in **NCM**: the latent code hierarchy (Appendix F.1), the NCM transformer (Appendix F.2), code corruption (Appendix F.3), and sampling (Appendix F.4). Unless otherwise stated, we run our ablations on TinyStories [40]. Each experiment changes one design choice and keeps every other setting fixed.

### F.1 Latent Code Hierarchy

#### F.1.1 Number of Levels

The number of levels  $K$  sets the number of generation steps. We compare ladders on TinyStories with  $K \in \{4, 8, 16, 32\}$ . Each uses a linear schedule starting at 1 and ending at 128.

**Setup.** Each ladder has its own tokenizer with 4,096 codes per level, trained for 40k steps. We then train an NCM transformer on top of each tokenizer for 10k steps. We do not use code corruption, so the ladder is the only meaningful difference between runs. We compute Gen. PPL using 512 generated samples and multiple sampling seeds, with GPT-2-Large as the reference model. Reference text has Gen. PPL 9.67 and unigram entropy  $H = 3.96$ .

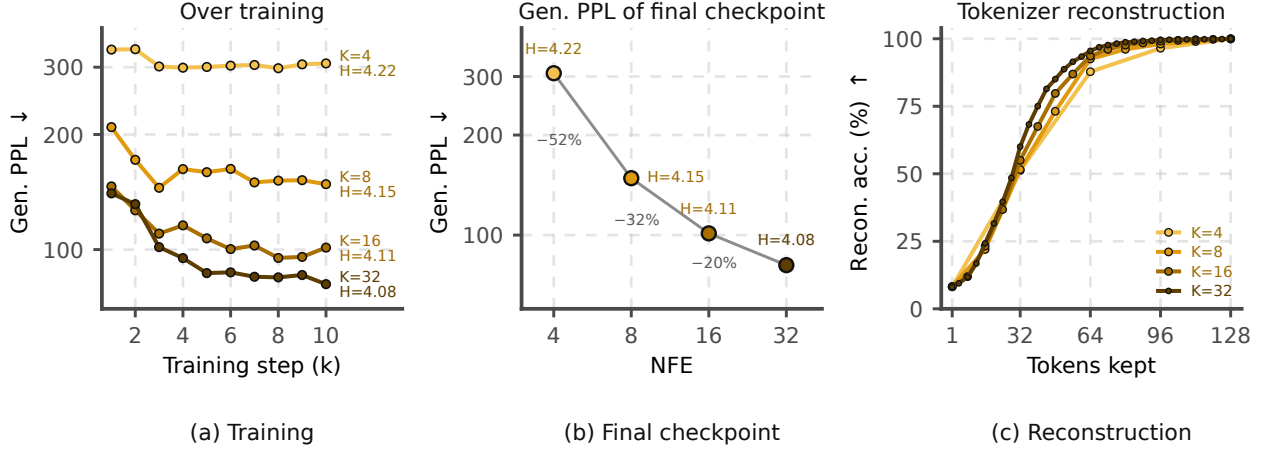


Figure 9: **Ablation on the number of levels in the hierarchy.** (a) Gen. PPL against training steps. (b) Gen. PPL at the final checkpoint. (c) Cumulative reconstruction accuracy per level.

Gen. PPL decreases as the number of levels in the hierarchy grows (Figure 9). It continues to improve at the largest hierarchy tested, but with diminishing returns: further increases in the number of levels yield smaller gains. Panel (c) shows cumulative reconstruction accuracy; longer hierarchies achieve slightly better final reconstruction, although all four reconstruct at least 99.95% of tokens. These ablations do not use code corruption, so errors can accumulate across levels; adding corruption may narrow the gaps between hierarchies.

### F.1.2 Level Schedule

The level schedule determines how the number of codes grows from coarse to fine. We consider six schedules: linear, quadratic, square-root, logarithmic, exponential, and cosine. Each has  $K = 12$  levels, starting at 1 code and ending at 128; code lengths are rounded to integers. Figure 10 compares Gen. PPL over training, Gen. PPL of the final checkpoint at 10k steps, and the schedules themselves.

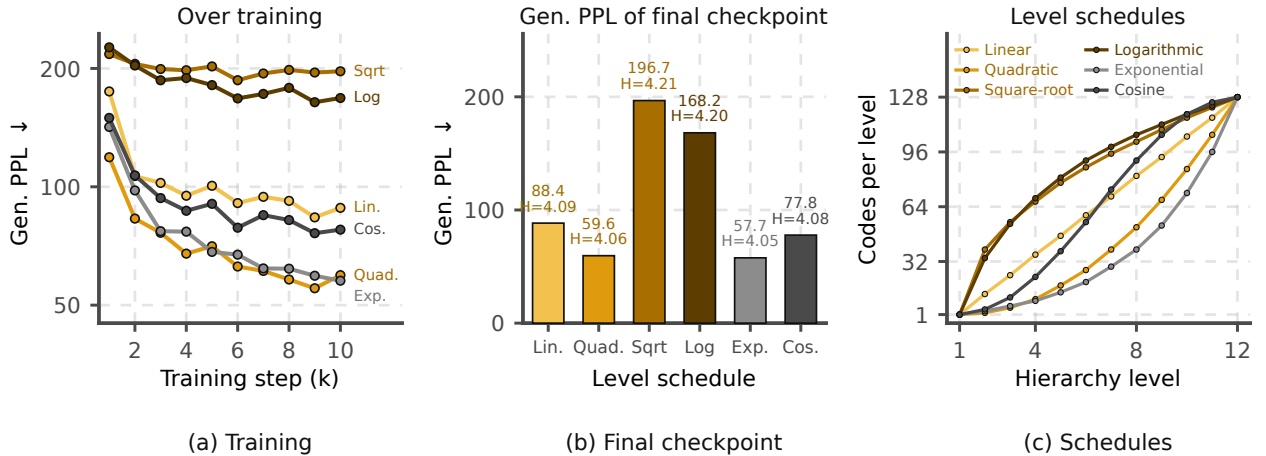


Figure 10: **Ablation on the level schedule.** (a) Gen. PPL against training steps. (b) Gen. PPL of the final checkpoint at 10k steps (bars). (c) Code lengths for six level schedules.  $H$  is the final per-sample unigram entropy.

## F.2 NCM Transformer

### F.2.1 Attention Pattern

For comparison purposes on TinyStories, we used the exact same 12-level tokenizer, generator, and selected conditioning-corruption recipe (mean rate 0.50; Appendix F.3) as in the previous example. The input blocks at each level are aggregated from the earlier levels. Block-causal attention also includes information from earlier blocks; a two-level window includes the preceding block. Block-diagonal attention is the best option for generating low PPL and fast sampling time with good preservation of entropy (Table 9). We therefore use this in **NCM**.

| Attention                 | Gen. PPL ↓   | $H$  | Time (s) ↓  |
|---------------------------|--------------|------|-------------|
| Block-causal              | 19.63        | 4.01 | 7.90        |
| Sliding window (2 levels) | 17.84        | 3.99 | 6.83        |
| Block-diagonal (ours)     | <b>15.41</b> | 3.98 | <b>5.65</b> |

Table 9: **Attention ablation on TinyStories.** Quality uses the selected corruption recipe;  $H$  is mean unigram entropy. Times are for 1,024 TinyStories samples at 12 steps with batch size 16 (Section 4.6); block-causal and sliding-window times are  $1.40\times$  and  $1.21\times$  the block-diagonal time.

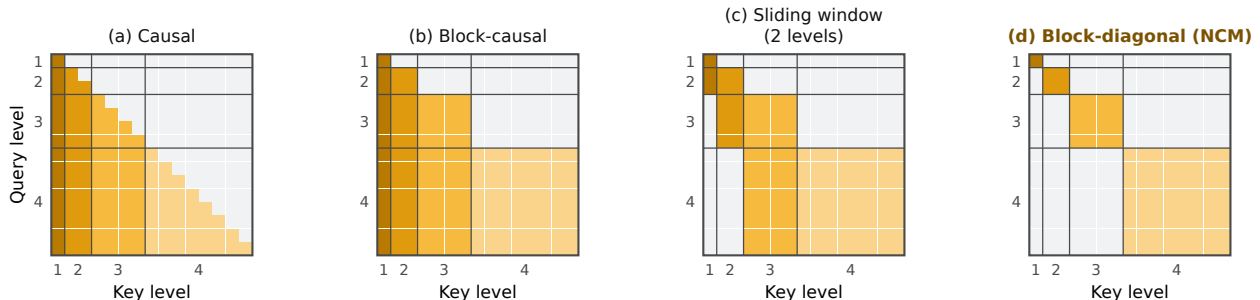


Figure 11: **Attention over cumulative-reconstruction blocks.** Allowed query-key pairs are colored by key level. (a) Causal (reference). (b) Block-causal. (c) Two-level sliding window. (d) Block-diagonal (**NCM**): attention stays within the current block, whose input already summarizes earlier levels.

### F.2.2 Context Encoder

To support conditional generation, **NCM** uses a prefix LM. One design decision is how to encode the prefix. On XSum, we compare the frozen encoder from pretrained T5-small with the encoder from our VQ-VAE (Table 10). T5-small achieves slightly higher ROUGE scores: +0.10 ROUGE-1, +0.15 ROUGE-2, and +0.24 ROUGE-L. However, it adds 35.3M parameters, approximately a 40% increase over our 88M model, and introduces a dependency on an externally pretrained language model. We therefore use the encoder from our VQ-VAE in all reported experiments.

| Metric     | T5-small<br>encoder | Ours<br>(VQ-VAE) | $\Delta$ |
|------------|---------------------|------------------|----------|
| ROUGE-1 ↑  | 34.89               | 34.79            | +0.10    |
| ROUGE-2 ↑  | 12.92               | 12.77            | +0.15    |
| ROUGE-L ↑  | 28.19               | 27.95            | +0.24    |
| Parameters | 123.4M              | <b>88.0M</b>     | +35.3M   |

Table 10: **Context encoder: frozen T5-small vs. our VQ-VAE encoder.** Both use the constant-LR recipe and greedy decoding at 8 NFE, on the full XSum test set.

### F.2.3 Loss Weighting

The loss weights level  $k$  by  $w_k \propto \ell_k^{-\alpha}$ :  $\alpha = 0$  weights every code equally, as the ELBO does (Appendix A.2), and  $\alpha = 1$  every level equally. On TinyStories (setup of Appendix F.3), raising  $\alpha$  from 0 to 1.5 moves the coarse levels’ share of the loss from 19% to 63% and lowers their validation loss, yet Gen. PPL after 10k steps stays at 18.9–19.0 for  $\alpha \leq 1$  and rises slightly to 19.4 at 1.5 (Figure 12).

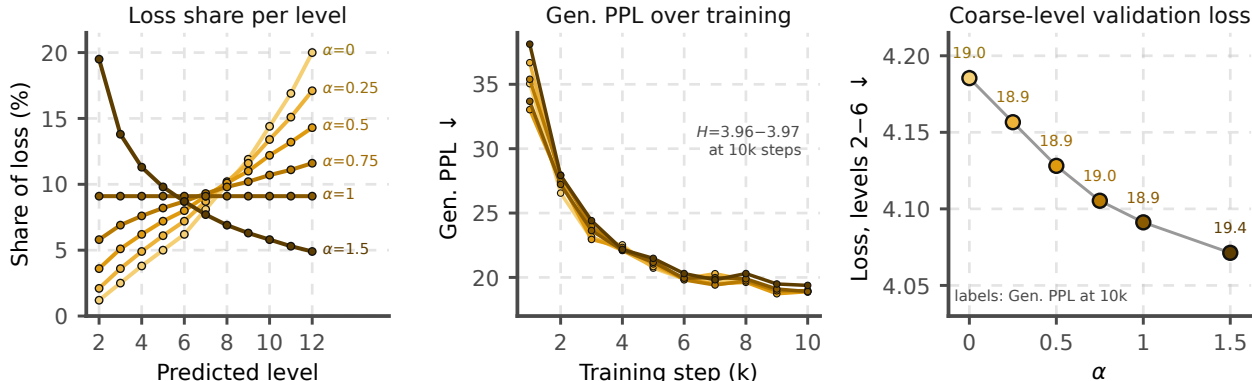


Figure 12: **Loss weighting** on TinyStories. **Left:** loss share per level. **Middle:** Gen. PPL during training. **Right:** coarse-level validation loss, labelled with Gen. PPL after 10k steps.

### F.3 Code Corruption

During training, NCM uses teacher forcing and conditions on clean input codes; during inference, it conditions on its own generated codes. This difference creates opportunities for errors to accumulate across levels. Since codes within each level are generated in parallel, multiple errors can enter the reconstruction used to generate the next level. We address this mismatch by making the model resilient to incorrect input codes: during training, we replace some input codes with entries sampled uniformly from the same level’s codebook, at a different rate per level. Targets remain clean.

This ablation and the loss-weighting ablation (Appendix F.2.3) use TinyStories (128 tokens) and the 12-level tokenizer of Table 5. Each 88.6M NCM transformer is trained for 10,000 steps with a batch size of 1,024. Gen. PPL is averaged over five sampling seeds with 256 samples each. The degree of corruption matters: moderate corruption greatly improves generation quality, while excessive corruption weakens the conditioning signal and degrades quality (Figure 13). The type of corruption, including how rates vary across levels and which replacement codes are selected, also affects how closely training resembles generation. We use a linearly decreasing corruption rate across levels. Corruption based on the model’s actual errors may better match inference conditions; we leave this to future work.

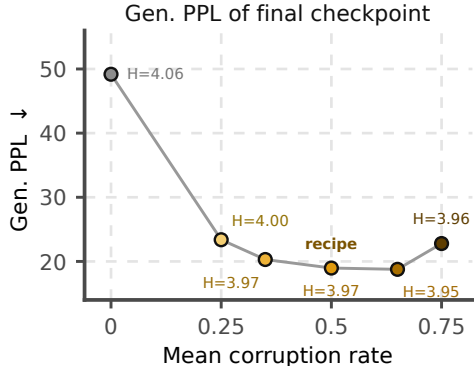


Figure 13: **Effect of code corruption.** Gen. PPL after 10k steps vs. mean corruption rate. Nonzero rates use a shifted linear decreasing schedule;  $H$  is entropy.

## F.4 Sampling

The quality of NCM’s generations may be enhanced during inference time without retraining. The two types of decoding choices we examine in Table 2 are Classifier-Free Guidance and Minimum Bayes Risk (MBR) Decoding.

**Classifier-Free Guidance.** For Classifier-Free Guidance, we use VAR’s approach [21] to allow for Classifier-Free Guidance in multiscale generations by combining conditional and unconditional logits from each scale using a guidance weight  $w$ . As illustrated in Figure 14, Classifier-Free Guidance results in less improvement on XSum summarization compared to WMT14 translation.

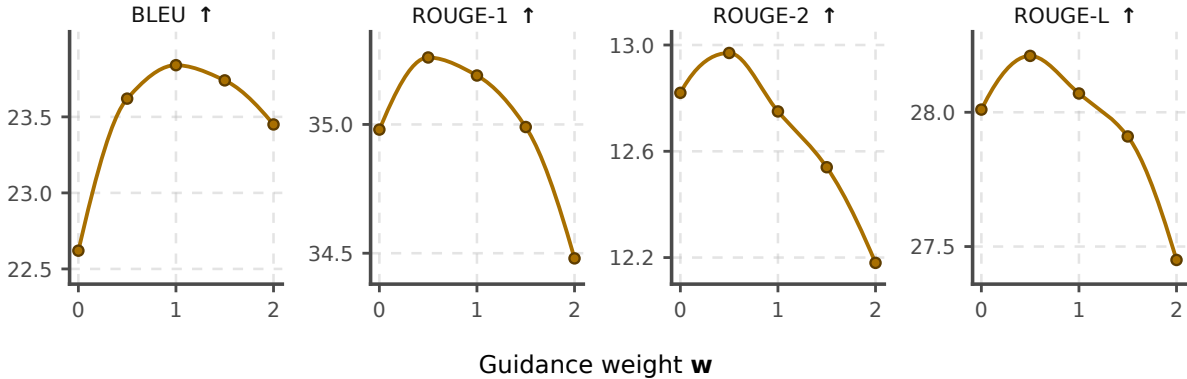


Figure 14: **Effect of classifier-free guidance.** De–En BLEU and XSum ROUGE-1/2/L as a function of the guidance weight  $w$ . All three ROUGE metrics peak at  $w=0.5$ , the setting reported in Table 2; BLEU, which is length-corrected, peaks later at  $w=1.0$ .

**Minimum Bayes Risk Decoding.** We draw  $K$  candidates and keep the one with the highest average similarity to the rest. Figure 15 sweeps  $K$ : all three metrics rise monotonically with diminishing returns, gaining 4.48 ROUGE-1 in total from  $K=1$  to  $K=256$ . The main-table settings use  $K=16$  on XSum (128 NFE) and  $K=12$  on WMT14 (132 NFE), yielding 29.6 ROUGE-L and 25.2 BLEU, respectively.

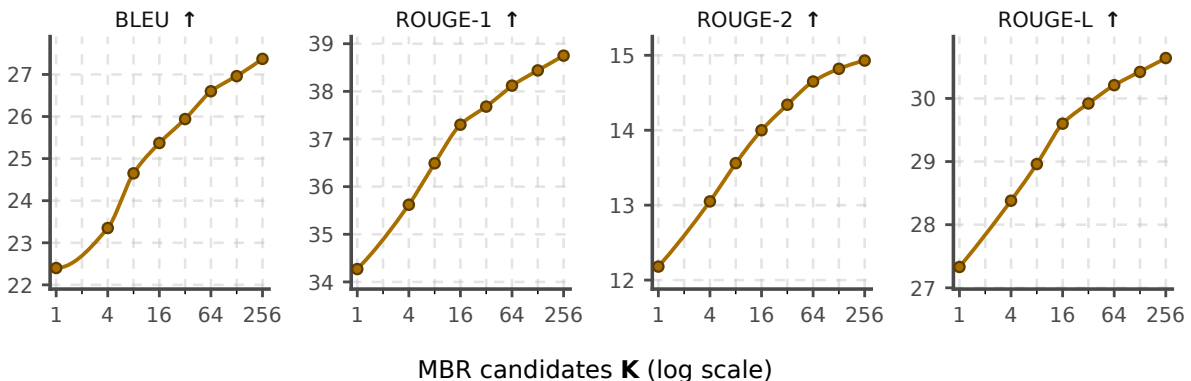


Figure 15: **Effect of MBR candidate count.** De–En BLEU and XSum ROUGE-1/2/L against the number of sampled candidates  $K$ , on a log-spaced axis. Quality improves as more candidates are sampled. NCM’s low sampling cost allows multiple candidates within the NFE budget that diffusion baselines use for a single sample, enabling extra inference compute to be used for sampling and verification through MBR selection.

## G Tokenizer Analysis

**Reconstruction & Utilization.** Figure 16 shows how the reconstruction builds up across the hierarchy of the OpenWebText (OWT) tokenizer. After each level  $k$ , we decode the cumulative reconstruction  $\hat{z}^{(k)}$  and measure exact token accuracy on held-out data. We also report the fraction of each level’s codebook entries used and its effective codebook utilization, measured by codebook perplexity. Reconstruction accuracy increases across levels, reaching 99.99% at full resolution.

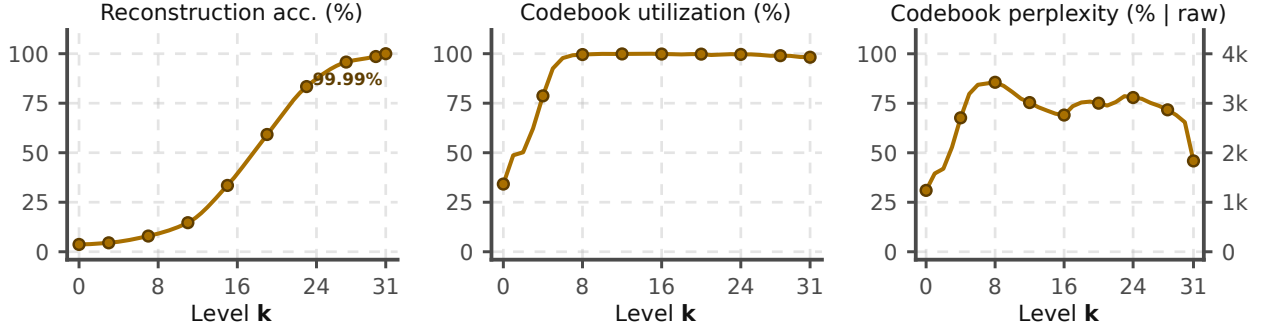


Figure 16: **Reconstruction and utilization by level.** Left: exact token accuracy after decoding levels 0 through  $k$  at the final tokenizer checkpoint. Middle: percentage of the entries of each level’s codebook selected at least once. Right: effective codebook utilization, measured by the perplexity of each level’s code distribution, as a percentage of the codebook size.

**Tokenizer Failure Cases.** We investigate typical tokenizer reconstruction errors using the OWT tokenizer as a case study. As reported in Table 1, it achieves 99.99% exact token reconstruction accuracy, corresponding to approximately 0.01% incorrectly reconstructed tokens. Table 11 summarizes representative errors, including substitutions of named entities and numerals, as well as errors in diacritics, usernames, markup, and word boundaries.

| Category        | Original → reconstruction, in context                                                                                                                |
|-----------------|------------------------------------------------------------------------------------------------------------------------------------------------------|
| Named entities  | ... City Metro Sydney <del>Sydney</del> <b>Melbourne</b> Melbourne Brisbane <del>Adelaide</del> <b>Adelaide</b> Adelaide Perth ...                   |
| Diacritics      | ... ev sahibi <del>takım</del> <b>takām</b> skoru ... geriye <del>düştü</del> <b>düştüü</b> ...                                                      |
| Numerals        | ... 1957 <del>1956</del> <b>1957</b> 1955 <del>1954</del> <b>1955</b> 1953 ... 1949 <del>1948</del> <b>1947</b> <del>1947</del> <b>1946</b> 1946 ... |
| Usernames       | ... 10 maxion 10 fishingcat 10 <del>da_lan_xiong_mao</del> <b>da_op_xiong_mao</b> ...                                                                |
| Markup          | ... < head > <del>&lt;</del> < > title > <del>&lt;</del> #BlackoutUK ...                                                                             |
| Word boundaries | ... Biological and Physical <del>SciencesJournal</del> <b>Sciences Journal</b> of Chemical ...                                                       |

Table 11: **Tokenizer reconstruction errors** on OpenWebText, by category. Struck-through grey: original; red: its reconstruction.

## H Probing the Hierarchy

We use the 12-level TinyStories model. TinyStories [40] contains short, simple stories that make changes easy to follow. We encode 128-token stories with the VQ-VAE to obtain codes at each level of the hierarchy.

**Shared Structure, Different Realizations.** We fix the first three levels of the hierarchy and randomly sample the remaining nine levels using two seeds. As shown in Figure 17, the outputs tend to preserve the broad theme of the source story while varying names, objects, and wording. Story A retains a girl seeking warmth and a garment that moves; Story B retains Tim’s alarm at something in the sky.

|                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                        |
|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <p><b>Source story A</b> 128 tokens</p> <p>One day, a little girl went outside to play. It was very cold, and she started to shiver. She saw an enormous tree with big leaves. The girl thought the leaves could be a wrap to keep her warm. So, she took one leaf and wrapped it around her. As she played, she found a big box under the tree. She opened the box and saw many wraps inside. She was very happy and took one wrap to wear. Now, she was warm and not shivering anymore. But then, something unexpected happened. The wraps in the box started to move! They were not wraps at all, but a</p> | <p><b>Source story B</b> 128 tokens</p> <p>One day, a boy named Tim saw a weird jet in the sky. It was big, shiny, and made a loud noise. Tim was scared and ran to his mom. "Mom, look at the weird jet!" he said. Mom looked up and saw it too. She told Tim not to worry, but he was still scared. Tim went to his friend, Sue. "Sue, did you see the weird jet?" he asked. Sue nodded and said, "Yes, it was big and loud." They both wanted to know what it was. They decided to ask Mr. Brown, who knew a lot about planes</p>   |
| <p><b>Story A, seed 1</b> 44% changed</p> <p>One day, a girl named Sue went outside to play. It was a cold, and Sue started to shiver. She saw a big tree with many leaves. Sue thought it would be a shelter to stay dry. So, she put her scarf and wrapped it around her neck. As Sue was walking, she found a small box under the tree. She opened the box and found many colorful hats inside. Sue was very happy. She decided to wear the scarf to play. She was not so cold anymore. But then, something unexpected happened. The scarf started to move! It was not just a scarf at all - a scarf</p>    | <p><b>Story B, seed 1</b> 51% changed</p> <p>One day, a boy named Tim saw a terrible cloud in the sky. It was big and made a loud noise. It was thunder. Tim was scared. "Mom, look at the big cloud!" he said. His mom looked up and saw it too. "Yes, Tim, the thunder is not scary," she said.. Tim’s friend, Sue, came over to. "Tim, come see the big thunder?" Sue asked. Tim said, "I think think it’s so big and scary to see!" They both decided to ask their teacher, Mr. Fox, about the thunder. Mr.</p>                    |
| <p><b>Story A, seed 2</b> 42% changed</p> <p>One day, a girl named Sue went to play outside. It was a cold day, and she started to shiver again. She saw a big tree, and thought it would be a nice to keep her warm. So, she took a sweater and wrapped it around her. As she played, she found a cold cold box under a tree. Sue opened the box and saw many pretty dresses. She was very happy. She decided to wear the sweater to play outside. She was not cold anymore. But then, something unexpected happened. The sweater started to move away! It was not a sweater at all! It was</p>               | <p><b>Story B, seed 2</b> 54% changed</p> <p>One day, a boy named Tim saw a weird thing in the sky. It was a big lightning and loud lightning outside. Tim was scared. He said, "Mom, look at the big lightning!" he said. His mom looked up and saw it too. She saw lightning too, Tim. It is not safe." Tim felt sad. Tim’s friend, Sam, came to play. "Did you see the lightning, Tim?" Sam asked. Tim nodded. "I saw a big, dark cloud in the sky!" They both knew what what to do. They decided to ask their friend, Mr. Owl.</p> |

Figure 17: **Shared structure, different realizations.** Each column shows one source story and two samples generated with its coarse codes fixed. Text is shown verbatim to the 128-token limit, including unfinished endings. Yellow highlights added or replaced words relative to the corresponding source; deletions are not marked.

**Mixing Coarse Representations from Different Stories.** We encode pairs of 128-token stories from TinyStories with the VQ-VAE, mix their coarse representations, and use NCM to generate the remaining finer levels conditioned on the mixture. Figure 18 shows two source pairs. Stories 1 and 2 blend drawing and billboard details with the cold-weather and moving-object narrative. Stories 5 and 6 begin with Tim and the jet before shifting to Max and the rain.

| Stories 1 and 2                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                |  | Stories 5 and 6                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                    |  |
|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|--|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|--|
| <p><b>Source A: Story 2</b> 128 tokens</p> <p>Sara and Tom were twins who liked to draw. They were gifted at making pictures with crayons and paper. They drew animals, flowers, cars, and houses. They wanted to share their pictures with everyone. One day, they saw a big billboard near their home. It was old and faded. It had a picture of a soda that no one drank anymore. The billboard was rotting and falling apart. "Let's draw on the billboard!" Sara said. "We can make it pretty and new!" "Good idea!" Tom said. "We can use our crayons and tape!" They took their crayons and paper and</p>                               |  | <p><b>Source A: Story 5</b> 128 tokens</p> <p>One day, a boy named Tim saw a weird jet in the sky. It was big, shiny, and made a loud noise. Tim was scared and ran to his mom. "Mom, look at the weird jet!" he said. Mom looked up and saw it too. She told Tim not to worry, but he was still scared. Tim went to his friend, Sue. "Sue, did you see the weird jet?" he asked. Sue nodded and said, "Yes, it was big and loud." They both wanted to know what it was. They decided to ask Mr. Brown, who knew a lot about planes</p>                                                                                                            |  |
| <p><b>Source B: Story 1</b> 128 tokens</p> <p>One day, a little girl went outside to play. It was very cold, and she started to shiver. She saw an enormous tree with big leaves. The girl thought the leaves could be a wrap to keep her warm. So, she took one leaf and wrapped it around her. As she played, she found a big box under the tree. She opened the box and saw many wraps inside. She was very happy and took one wrap to wear. Now, she was warm and not shivering anymore. But then, something unexpected happened. The wraps in the box started to move! They were not wraps at all, but a</p>                              |  | <p><b>Source B: Story 6</b> 128 tokens</p> <p>Once upon a time there were two puppies, one was named Spot and the other was named Max. Spot was so cheerful and playful, he loved to run around outside and enjoy the sunshine. Max was quite different though, he was much more shy and preferred to stay inside all day. One day it was raining outside so Spot and Max couldn't go out and play. Even though Spot was sad he still decided to try and cheer Max up. He started to display such a playful attitude, trying to invite Max to come out and play even though it was raining. However, Max wanted to stay inside and remain safe</p> |  |
| <p><b>Mixed coarse representations</b></p> <p>Sara was a little girl who liked to draw. It was cold, but she loved her crayons. She drew flowers, houses houses, and big trees. She wanted to share her crayons to keep everyone warm. So, she took a big billboard near her house and wrapped it. As she drew, she found a beautiful picture under a tree. She it billboard and saw many colors inside. She was very happy and took one picture on the billboard Sara. Now, she could make it pretty anymore. But then, something unexpected happened. The pictures in the picture started to move! They were not their lines at all, but</p> |  | <p><b>Mixed coarse representations</b></p> <p>One day a boy named Tim saw a weird jet in the sky. It was big, shiny, and made a loud noise. Tim was scared and ran to his mom. "Mom, look at the weird jet!" he said. Mom looked up and saw it too. She told Tim not to worry, but it was raining outside, and Max couldn't go out and play. Even though he was scared, he wanted to try and cheer Max up. He started to have such such a playful friend, Tim, waiting Max to come out and play even though it was raining. Eventually, Max wanted to stay inside and stay stay</p>                                                                |  |

Figure 18: **Mixing coarse representations.** Each column shows two sources and one output with newly generated finer levels. Green (source A) and yellow (source B) highlight selected shared wording and details. All text is verbatim, including unfinished endings.

**Perturbing Early Codes.** We perturb a story’s coarse representation by replacing two codes with random codes observed at the same level across validation stories, then generate the finer levels. Using the same sampling seed, we compare outputs with and without these replacements. Figure 19 shows that changing two coarse codes can alter characters, objects, and events while preserving parts of the narrative.

|                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                 |      |            |      |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                          |      |      |      |            |
|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|------|------------|------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|------|------|------|------------|
| Story 1: original validation text                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                               |      |            |      |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                          |      |      |      | 128 tokens |
| One day, a little girl went outside to play. It was very cold, and she started to shiver. She saw an enormous tree with big leaves. The girl thought the leaves could be a wrap to keep her warm. So, she took one leaf and wrapped it around her. As she played, she found a big box under the tree. She opened the box and saw many wraps inside. She was very happy and took one wrap to wear. Now, she was warm and not shivering anymore. But then, something unexpected happened. The wraps in the box started to move! They were not wraps at all, but a |      |            |      |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                          |      |      |      |            |
| Position 0                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                      | 1    | 2          | 3    | 4                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                        | 5    | 6    | 7    |            |
| 3165 → 512                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                      | 3094 | 410 → 2383 | 963  | 3438                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                     | 370  | 3334 | 2301 |            |
| Clean, seed 1                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                   |      |            |      | Perturbed, seed 1                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                        |      |      |      |            |
| One day, a lovely cat named Lily went outside to play. She liked to play in the sun. She saw a big box on the ground. Lily thought it would be a good place to hide. So she decided to keep it. While playing, Lily found a small key. She put the key on. She opened the box and found a toy inside. Inside, she found a real toy cat! Lily was so happy to find it. She wanted to show friends her new toy all the time. But then, something unexpected happened. The toy cat started to move! It was not a real cat, but real                                |      |            |      | One day, a girl named Sue went to play outside. She liked to dig in the sun. She saw a key on the ground. She picked it up and thought it would be fun. So, she decided to keep it. WhileWhile Sue was digging, she found a small key key on it. She opened the key and found a secret box inside. Inside the box, she saw many fake toys inside She was so happy and surprised. She wanted to play with her new toys all day long. But then, something unexpected happened. The fake toys started to talk! It was not a real toy, but a |      |      |      |            |
| Story 2: original validation text                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                               |      |            |      |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                          |      |      |      | 128 tokens |
| Sara and Tom were twins who liked to draw. They were gifted at making pictures with crayons and paper. They drew animals, flowers, cars, and houses. They wanted to share their pictures with everyone. One day, they saw a big billboard near their home. It was old and faded. It had a picture of a soda that no one drank anymore. The billboard was rotting and falling apart. "Let's draw on the billboard!" Sara said. "We can make it pretty and new!" "Good idea!" Tom said. "We can use our crayons and tape!" They took their crayons and paper and  |      |            |      |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                          |      |      |      |            |
| Position 0                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                      | 1    | 2          | 3    | 4                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                        | 5    | 6    | 7    |            |
| 1474 → 219                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                      | 3164 | 3742       | 2611 | 4055 → 2225                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                              | 1178 | 2580 | 3910 |            |
| Clean, seed 1                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                   |      |            |      | Perturbed, seed 1                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                        |      |      |      |            |
| Tom and Lily were twins who liked to play with cardboard. They had a lot of toys: cars, dolls, blocks and more. They wanted more and more, but they always wanted to play with. One day, they found a new box in the garage. It was big and red and shiny. It was a box of cardboard that Mom wrapped. Tom wondered what it was for. "Let's go play with it!" Lily said. "Maybe we can make a spaceship!" Tom said. They ran to the living room and tore the cardboard into pieces. They made circles, lines, squares and                                       |      |            |      | Ben and Lily were twins who liked to play with their toys. They had a lot of toys: cars, dolls, dolls, blocks, books, books and more, but they always wanted more and. One day, they found a new box in the garage. It was big and red and shiny. It had many lights and stickers inside. Ben and Lily wanted to see what was inside. "Let's play with the tape!" Ben said. "Maybe we can make something!" Lily said. They ran to the living room and cut the tape into pieces. They made circles, stickers, and stickers                |      |      |      |            |

Figure 19: **Replacing two early codes.** Each example shows the original story, its level-1 code replacements, and paired clean and perturbed outputs with the same seed. Yellow marks word-level differences on both sides; punctuation-only changes are unmarked. Code positions index eight equal regions of the 128-token grid. Text is verbatim, including unfinished endings.

## I Limitations & Future Work

NCM shows promising results on unconditional generation and downstream conditional generation tasks such as summarization and translation. Several directions remain for further improvement. NCM fixes both the hierarchy depth  $K$  and the per-level lengths  $\ell_k$ , so even short continuations traverse the full hierarchy; an input-dependent schedule would adapt compute to example complexity. NCM relies on a separately trained multiscale VQ-VAE tokenizer, decoupling representation learning from the generative objective. An end-to-end training approach could benefit NCM by aligning representation learning with the generative objective. Our main experiments use generators of roughly 100M parameters, and our scaling study extends to 725M parameters on LM1B. Scaling to larger datasets and model sizes is another exciting direction. Generations are generally fluent but exhibit occasional artifacts at coarse-level transitions, presenting another opportunity to further improve NCM.

## J Additional Qualitative Samples

Figure 8 presents short samples from WMT14 German–English, XSum, LM1B, and OWT. In this section, we give three additional samples each for WMT14 German–English, XSum, TinyStories, LM1B, and OpenWebText, with the relevant metrics reported where available, and we follow NCM through one hard Sudoku puzzle step by step (§J.7).

### J.1 WMT14 De–En

| Sample 1                                                                                                                                                                          |  | BLEU: 30.6 |
|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|--|------------|
| <b>German:</b> Volkswirtschaftler sorgen sich auch wegen der hohen Verbraucherverschuldung in Thailand, das diese Woche knapp aus einer technischen Rezession herausgekommen ist. |  |            |
| <b>Reference:</b> Economists are also worried about high levels of consumer debt in Thailand, which this week narrowly emerged from technical recession.                          |  |            |
| <b>Generated:</b> Economists also worry about high consumer debt in Thailand, which has just come out of a technical recession this week.                                         |  |            |

| Sample 2                                                                                                                                                                                |  | BLEU: 55.7 |
|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|--|------------|
| <b>German:</b> Außerdem erlebte er auf seiner Bahnfahrt von Titisee-Neustadt nach Freiburg die bereits unternommenen Anstrengungen für den Ausbau des öffentlichen Personennahverkehrs. |  |            |
| <b>Reference:</b> In addition to this, on his train journey from Titisee-Neustadt to Freiburg, he experienced first-hand the efforts already undertaken to develop public transport.    |  |            |
| <b>Generated:</b> Furthermore, on his train journey from Titisee-Neustadt to Freiburg, he experienced the efforts already made to develop local public transport.                       |  |            |

| Sample 3                                                                                                                                                                                    |  | BLEU: 18.6 |
|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|--|------------|
| <b>German:</b> Samsung, Huawei und HTC stellen Handys her, die mit Googles Betriebssystem Android arbeiten, das in scharfem Wettbewerb zu den Mobilprodukten von Apple und Microsoft steht. |  |            |
| <b>Reference:</b> Samsung, Huawei and HTC all manufacture phones that operate on Google’s Android operating system, which competes fiercely with Apple and Microsoft mobile products.       |  |            |
| <b>Generated:</b> Samsung, BMW and HTC make mobile phones that work with Android’s operating operating system, which is firmly competing with Apple and and Microsoft products.             |  |            |

## J.2 Generation Trajectory

Table 12 shows the German source, reference, final generated translation, and level-by-level outputs of a single generation run. Figure 3 shows selected levels from the same run. Each generated row decodes the cumulative reconstruction after that level. Intermediate quality is not monotonic.

| Translation trajectory                                                                                                                     |                                                                                                                                                | Final BLEU: 77.8 |
|--------------------------------------------------------------------------------------------------------------------------------------------|------------------------------------------------------------------------------------------------------------------------------------------------|------------------|
| <b>German:</b> Es besteht allerdings die Sorge, dass kleine und mittlere Unternehmen angreifbar gegenüber Hacking und Überwachung bleiben. |                                                                                                                                                |                  |
| <b>Reference:</b> However, there is concern that small and medium-sized companies remain vulnerable to hacking and surveillance.           |                                                                                                                                                |                  |
| <b>Generated:</b> However, there is concern that small and medium-sized enterprises will remain vulnerable to hacking and surveillance.    |                                                                                                                                                |                  |
| Level                                                                                                                                      | Decoded output after this level                                                                                                                | BLEU ↑           |
| 1                                                                                                                                          | However, it is a important to the the the and the.                                                                                             | 6.0              |
| 2                                                                                                                                          | However, it is a small to to small and and small, and small and and.                                                                           | 7.2              |
| 3                                                                                                                                          | However, it is a concern for small small small medium-sized and-sized-sized and and and...                                                     | 6.3              |
| 4                                                                                                                                          | However, there is a medium of medium-sized enterprises remain remain remain remain to to to to.                                                | 16.0             |
| 5                                                                                                                                          | However, there is a concern of medium medium-sized-sized enterprises remain remain remain and and orientation.                                 | 17.5             |
| 6                                                                                                                                          | However, there is a concern that medium medium-sized enterprises will remain remain remain surveillance surveillance surveillance capabilities | 17.7             |
| 7                                                                                                                                          | However, there is that that and and medium-sized enterprises will remain remain vulnerable surveillance and surveillance.                      | 24.2             |
| 8                                                                                                                                          | However, there is that concern and and medium-sized enterprises will remain vulnerable to hacking and surveillance.                            | 51.7             |
| 9                                                                                                                                          | However, there is that that small and medium-sized enterprises will remain vulnerable to to and surveillance.                                  | 38.9             |
| 10                                                                                                                                         | However, there is concern that small and medium-sized enterprises will remain vulnerable to to surveillance surveillance.                      | 56.3             |
| 11                                                                                                                                         | <b>However, there is concern that small and medium-sized enterprises will remain vulnerable to hacking and surveillance.</b>                   | <b>77.8</b>      |

Table 12: **German–English translation trajectory.** Each row decodes the cumulative reconstruction after one level of a single NCM generation; text is shown as generated, and BLEU is computed per level against the reference. The final level (shaded) is the output.

## J.3 XSum

| Sample 1                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                         | ROUGE-1: 80.00   ROUGE-2: 71.43   ROUGE-L: 80.00 |
|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|--------------------------------------------------|
| <b>Context:</b> The Brazil international, who had two years remaining on his current deal, scored 13 goals in 59 games for the Blues last season. Willian, 27, joined from Russian side Anzhi Makhachkala for a reported £30m in 2013. “It’s a dream to continue with this club and over the next four years I will try to help the team win more trophies,” he said. Willian, who has scored 19 goals in 140 matches for Chelsea, was named player of the year last season as the club finished 10th in the Premier League. He has won 30 caps for Brazil since making his debut in 2011, scoring six goals. Last week, Chelsea signed Belgium striker Michy Batshuayi from Marseille for a fee believed to be worth £33m, their first capture under new manager Antonio Conte. |                                                  |
| <b>Reference:</b> Chelsea midfielder Willian has signed a new four-year contract with the club.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                  |                                                  |
| <b>Generated:</b> Chelsea midfielder Willian has signed a new four-year contract to stay with the Premier League champions.                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                      |                                                  |

**Sample 2****ROUGE-1:** 73.3 | **ROUGE-2:** 57.1 | **ROUGE-L:** 73.3

**Context:** The incident happened in Gerard Street, Ashton-in-Makerfield in Wigan at about 14:45 BST on Saturday. The woman, who has not been named but was in her 70s, was critically injured and died at the scene. Gerard Street was closed for a time while investigations were carried out. Police have appealed for any witnesses to contact them.

**Reference:** A woman has died after being hit by a lorry in Greater Manchester, police have said.

**Generated:** A woman has died after she was hit by a car in Greater Manchester.

**Sample 3****ROUGE-1:** 58.82 | **ROUGE-2:** 12.50 | **ROUGE-L:** 41.18

**Context:** The scene from Alice in Wonderland in Wardie Primary’s assembly hall dates back to 1936 and was recently restored with a Heritage Lottery Fund.

A mother has lodged a complaint about the image describing it as racist.

Edinburgh City Council said it understands the offensiveness of the image but said it does not reflect the attitudes of the school.

An Edinburgh City Council spokesman said: “The Alice in Wonderland mural at Wardie Primary School was painted in 1936 and is of both historical and artistic importance as evidenced by the fact it recently received full Heritage Lottery Funding support to restore the work.

“While we understand the offensiveness of the image, it is in no way indicative of the attitudes of either the school or the council.

“Our equalities policies and approaches are robustly multi-cultural and anti-racist, promoting diversity and good relationships among pupils.”

A Police Scotland spokesman said: “Police in Edinburgh have received a complaint in relation to a mural at a primary school in the Trinity area.

“Officers are now liaising with Edinburgh City Council education department with regards to this matter.

“Police Scotland treats all reports relating to hate incidents extremely seriously and will thoroughly investigate whenever a report of this nature is made.”

**Reference:** Police are investigating a complaint about a mural at an Edinburgh primary school which features a goliwog.

**Generated:** A police investigation is under way after a mural was painted at a primary school in Edinburgh.

## J.4 TinyStories

**Sample 1****Gen. PPL:** 10.2 | **H:** 3.98

Once upon a time, there was a little boy named Tim. Tim loved to play with his yellow blocks. One day, he wanted to build a tower with his blocks. He was happy. Tim’s best friend, Tom, came over to play. Tom asked Tim, “Can I play with your blocks?” Tim said, “No,, you cannot play with my blocks. But we can get us to play for you.” They played and had fun. They put the yellow blocks and made an even bigger tower. It was tall and colorful. Tim and Tom had so much fun. They built the tower and made...

**Sample 2****Gen. PPL:** 14.1 | **H:** 3.88

Once upon a time, there was a little dog named Sam. Sam was a big dog who liked to pull rope. He often liked to watch his pants spin and wagging his tail very much. One day, Sam wanted his rope to his side, but the rope was too high for him to move. Sam felt sad and did not know what to do. His friend, Tom the cat, saw what was wrong. Sam asked him back for help. Sam and Sam tried to pull the rope together. They pulled and pulled, but the rope would not move. Sam was sad and he could not get to his side.

**Sample 3****Gen. PPL:** 17.9 | **H:** 4.03

Once upon a time, there was a small brother named Sam. One day, Tim and Sam to the table. He saw something unusual in the kitchen. It was a little egg out of toast. Tim said, “Look, Sam! A little little egg!” Sam looked and stared at the tree. Suddenly, it started to crack. The egg began to shake. Tim was scared. He did not want to lose the eggs or the cake. He watched as the little cake landed on the tree branch. Tim picked it up and smiled. “Look, Mom! The cake will fall back!” he said. His mom came...

## J.5 LM1B

### Sample 1

Gen. PPL: 39.4 | H: 4.29

[CLS] swine flu cases have been confirmed across the united states. [CLS] there are actually more people that see it—since you don't worry about other things. [CLS] the labor department says. [CLS] the two proposals are approved by the end of the week. [CLS] he soon said that he always felt him about health care which was not a practical issue and insisted on comprehensive reforms to improve a child's health. [CLS] the euro also added 1.33 per cent to \$1.46829 against the dollar and gained 0.8 percent tob72,54 against the yen and slipped 2.4 percent barrel to [CLS]

### Sample 2

Gen. PPL: 66.1 | H: 4.39

[CLS] to be welcome—the united nations security resolution that was imposed by russia earlier this month, the united states, european union and the baltic states to take sanctions against iran, whose nuclear program also is its west and nuclear programme. [CLS] they will cover the amount of state subsidies on power and equipment. [CLS] any power cuts rates will be likely on those who are still in talks over a deal late thursday, according to a government report released late tuesday. [CLS] conservative advocates have done their effort to helping people who have fallen before given to come into life while others has in particular turn them off work, he said. [CLS] more than two [CLS]

### Sample 3

Gen. PPL: 72.0 | H: 4.35

[CLS] that individuals should be given an perspective on where the things are right now, and when we know things are doing well you know that the federal government should still take measures to get us back over the next three months. [CLS] "it is certainly saying, by 50 it isn't going to be put this week in charge, but this has been a ridiculous prospect.. 'i thought would say we have been very good before election," he added. [CLS] gs said the profit dropped 8 percent to £96.7 million, and revenues grew 9 percent from \$10bn to £2.7m. [CLS] " [CLS]

## J.6 OpenWebText

We provide excerpts of 1,024-token unconditional generations from our NCM model trained on OWT. Samples are shortened for space and readability; ellipses mark omissions. Gen. PPL and entropy are computed on the full samples.

### Sample 1

Gen. PPL: 47.8 | H: 5.40

Compise of Doctor is the world's first shot villain in the Marvel Marvelion and appears to have host a full trailer for the beloved new Marvel franchise and Arrow in Daredevil. He has the upcoming show scenes in November 2015. Welcome to the famous show, as well as those who joined Marvel films. "Dark franchise is a wonderful, kind of exciting," Batman said in an interview. JoseRamos told reporters in a conference. "I feel like I never happen, but never is a dream. I hope I do it." ... "It's wonderful, wonderful actors," he said. "You can't watch it and watch a video on screen." Watch the full video of Batman's future in Flash or Flash? ... It's a difficult matter to imagine how it can make it easier for you to talk about content, since it's hard to work on what works. Here's more information about the future, in that context of out aspects of "wots," and the products that are merely cultural historical references to them online. Do you think the way they keep their web communication online? ... Facebook's content is called "The Internet," a web company called Media tool. It allows users to speed up interviews with some of the top web sites, including web sites, said Thomas William Abbartis, who just appeared on an online strategy that seemed to focus its attention to the web, its products and its groups. ... Most of these companies are based on social sites, and Dr. Ababbartis's blog on Facebook's new site is to popularize web sites on its,, efficient manner. ...

**Sample 2****Gen. PPL: 26.4 | H: 5.27**

... The growth is slowing down, the job creation is ramping up, and still again, a quarter of the Oregonians are stuck in part-time or without not work at all.

A new report shows Walmart's workforce gained economic growth this fall, dropping up an estimated 5,300 jobs last year alone, after increasing improvements for recovery. Instead, employment has declined, from 7.1 percent in August to to 7.3 percent in November, a five-year low.

But seasonally 272,195,000 people were unemployed in November, and just as many people were underemployed -- working part-time, but they could not get full-time jobs.

"Even though we're seeing [a] jobs recovery, there's still a lot of people who need to be working more than they are," said Nicke Belele, a Colorado statistician. ...

"It's just just about the quality of the jobs out there, it's the quality of jobs," State Department Mark McMullen said. "We still have some work to do in terms of the quality of jobs."

The November jobs report, issued Tuesday morning, will be the date of the last year. McMullen said it makes clear that Oregon has undergone rapid economic growth this fall, but the damage left by the workforce still can't be developed. The longer they are slightly underizedized, the bigger the focus on Oregon's economic growth potential. ...

And things could turn for things worse quickly. An estimated 17,000 Oregonians will run out of jobless unemployment in two weeks, when the long-term recovery emerged. ...

**Sample 3****Gen. PPL: 41.5 | H: 5.12**

Three Canadian music star Thomas Smith partnered with Graham Graham's Matthew Helons. ...

Steve Feich forward: I am going to start a year later this year, so are your thoughts have changed completely?

Since then, I understand that I've managed to grow up as a person as a decent person. ...

What are your thoughts on my career with last year?

It was so important. It was important for a whole lot. But I just can't do just any other than everyone. But I was for the whole comfortable feeling, and that's what I was happy. I wanted to do. ...

How can you change your career, the enthusiasm in your season, and how too much do you feel when you're in the team?

There's a lot of opportunities and a lot of opportunities. I know that I would be happy, and I know a lot that they're actually the ones, and the ones who do, but I don't say what I wanted to do to. ...

How was it hard for my professional career? Do you think you spend your life and present for your life?

Well, I thought it to be fairly unfair at risk of being college football. I have always been informed, busy, and playing while working hard and as a user. As a result, it's too hard to get in the right five hours and yourself being focused on your sweet school school at the football level. It's so much harder than the other player.

What do you have to do?

I have a whole moment at the good balance, for which I'm grateful for you for what's going into my life. ...

## J.7 Sudoku

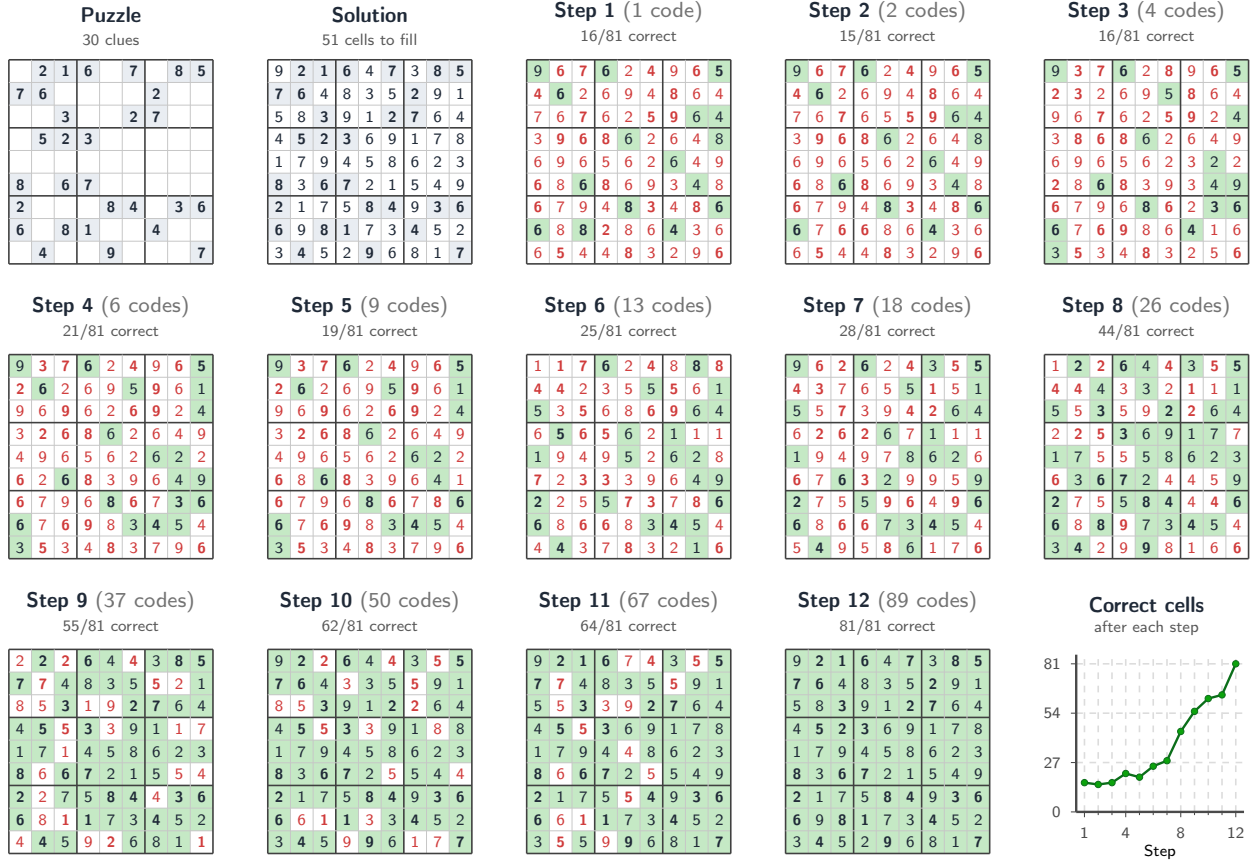


Figure 20: **Solving a hard Sudoku puzzle step by step.** Top left: the puzzle, with its 30 clues shaded, and its solution. Step  $k$  decodes the codes that **NCM** has generated in its first  $k$  steps, where step  $k$  generates level  $k$  of the code hierarchy; its header gives the number of codes at that level and how many of the 81 cells match the solution. Green cells are correct, red digits are wrong, and bold digits mark the clue positions. Bottom right: correct cells after each step. Across levels, **NCM** iteratively refines the decoded board until it reaches the correct solution.