UNDERSTANDING ENCODING IN CONTINUOUS DIFFUSION LANGUAGE MODELS

Anonymous authors
Paper under double-blind review

ABSTRACT

Continuous language models encode discrete tokens as continuous vectors, but it remains unclear whether encoding design should follow the same principles as in discrete models. In this work, we first show theoretically that encoding has an additional target-defining role in continuous models. Controlled experiments further show how this role leads encoding to affect the two model families in qualitatively different ways. Building on this distinction, we systematically study encoding design and identify geometry, dimension, and encoding–model compatibility as three key considerations. Guided by these insights, we progressively refine a continuous language model's encoding and neural parameterization, achieving a GenPPL of 15.8 on OpenWebText at approximately 130M parameters and outperforming representative state-of-the-art diffusion language models of comparable scale. Our findings establish encoding as a distinct design problem in continuous language modeling and demonstrate the value of systematic encoding analysis.

1 INTRODUCTION

Continuous language models (CLMs) have emerged as a promising alternative for language modeling, offering flexible continuous trajectories, a mature toolbox of few-step sampling methods, and potential for multimodal generation (Gulrajani & Hashimoto, 2023; Chen et al., 2026; Hu et al., 2026; Yang et al., 2026). Despite recent progress, a substantial performance gap remains in many settings relative to mainstream discrete language models (DLMs), including autoregressive models (Achiam et al., 2023; Guo et al., 2025) and masked diffusion models (Nie et al., 2025; Ye et al., 2025). Unlike these DLMs, CLMs model the discrete distribution over token sequences indirectly by approximating a continuous surrogate target and decoding continuous states into tokens. This surrogate is constructed from the discrete data distribution through an *encoding* that maps tokens into a continuous vector space. Although prior work has explored encoding choices for CLMs (Li et al., 2022; Strudel et al., 2022; Dieleman et al., 2022), a systematic understanding of how encoding requirements differ between CLMs and DLMs remains limited. This raises an important but underexplored question: *does encoding design require different principles for CLMs and DLMs?*

We address this question in Section 3.1 by decomposing training objectives to distinguish encoding's representation role in both DLMs and CLMs from its additional target-defining role in CLMs. Controlled experiments in Section 3.2 further show that encoding affects the two model families in qualitatively different ways, demonstrating the practical importance of this distinction.

Motivated by this observation, we revisit the encoding problem for CLMs in Section 4 without relying on intuitions inherited from discrete representation learning. Systematic experiments across vocabulary sizes reveal geometric and dimensional effects that differ markedly from those in discrete models. Together with our analysis of jointly learned encodings, these results identify three key design considerations: encoding geometry, encoding dimension, and encoding–model compatibility.

These findings guide the real-vocabulary experiments in Section 5, where we progressively improve a CLM by refining its encoding and neural parameterization. On OpenWebText, our approximately 130M-parameter model achieves a GenPPL of 15.8, outperforming representative state-of-the-art diffusion language models of comparable scale. These results demonstrate the practical value of treating encoding as a distinct design problem for CLMs.

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

Our contributions are summarized as follows:

- We distinguish encoding's representation and target-defining roles through a decomposition of discrete and continuous language model training objectives, and use controlled experiments to show that encoding affects the two model families in qualitatively different ways.
- We identify encoding geometry, encoding dimension, and encoding–model compatibility as three key design considerations for CLMs, revealing requirements that differ markedly from those of discrete representation learning.
- We apply these insights to real-vocabulary language modeling, using jointly learned encodings and encoding-aware neural parameterizations to substantially improve generation.

2 PRELIMINARIES

Let $X = (X_1, \ldots, X_n) \in \Omega^n$ denote a token sequence over the vocabulary $\Omega = \{1, \ldots, K\}$. All models considered here aim to approximate the discrete target distribution P on Ω^n with a model distribution Q. We use p and q for the corresponding continuous target and model distributions.

2.1 DISCRETE LANGUAGE MODELS

DLMs parameterize Q^{DLM} directly on Ω^n, using numerical encodings to represent token identities. Standard one-hot encoding deterministically maps each token X_i to the basis vector $\boldsymbol{e}^{(X_i)} \in \mathbb{R}^K$, which a learned linear projection then transforms into an internal embedding. We distinguish encoding from embedding throughout the paper and explain its significance in Section 3.

Specifically, autoregressive (AR) models (Brown et al., 2020; Achiam et al., 2023) factorize Q^{AR} from left to right and minimize the expected negative log-likelihood $\mathcal{L}_{\mathrm{AR}}$. By contrast, masked diffusion models (MDMs) (Lou et al., 2024; Ou et al., 2025; Sahoo et al., 2024) define a fixed forward masking process $R(Z^{1:T} \mid X)$, where T is the number of masking steps and $Z^0 = X$. Each state Z^t is a sequence over $\Omega \cup \{\mathrm{M}\}$, with the mask symbol $\mathrm{M} \notin \Omega$. The reverse model predicts clean tokens from masked states, generating samples by progressively unmasking a fully masked sequence. The resulting joint $Q^{\mathrm{MDM}}(X, Z^{1:T})$ is trained using an expected negative evidence lower bound (NELBO), denoted by $\mathcal{L}_{\mathrm{MDM}}$. The explicit forms of both objectives are given in Appendix A.1.

2.2 CONTINUOUS LANGUAGE MODELS

CLMs construct Q^{CLM} indirectly through a continuous model $q(\boldsymbol{z}^0)$ that approximates a continuous surrogate target $p(\boldsymbol{z}^0)$. A discrete decoder $D(X \mid \boldsymbol{z}^0)$ then maps continuous states $\boldsymbol{z}^0 \in \mathbb{R}^{nd}$ to token sequences, yielding

$$Q^{\mathrm{CLM}}(X) = \int q(\boldsymbol{z}^0) D(X \mid \boldsymbol{z}^0)\, d\boldsymbol{z}^0, \tag{1}$$

where d is the encoding dimension per token. The joint $q(\boldsymbol{z}^0)D(X \mid \boldsymbol{z}^0)$ also defines the endpoint posterior $q(\boldsymbol{z}^0 \mid X)$ through Bayes' rule.

To learn $q(\boldsymbol{z}^0)$, continuous diffusion models (CDMs) (Sohl-Dickstein et al., 2015; Ho et al., 2020; Li et al., 2022) progressively add noise in the continuous space through a fixed forward process $r(\boldsymbol{z}^{1:T} \mid \boldsymbol{z}^0)$. This defines the target path density $p(\boldsymbol{z}^{0:T}) = p(\boldsymbol{z}^0) r(\boldsymbol{z}^{1:T} \mid \boldsymbol{z}^0)$. The model learns a reverse denoising process $q(\boldsymbol{z}^{0:T})$ to approximate this path distribution, thereby obtaining its endpoint marginal $q(\boldsymbol{z}^0)$. Together with D, it is trained using the discrete-data NELBO $\mathcal{L}_{\mathrm{CLM}}$, whose explicit form is also given in Appendix A.1.

Notably, the continuous surrogate target $p(\boldsymbol{z}^0)$ is itself induced by encoding. Here encoding is represented by a conditional distribution with density $e(\boldsymbol{z}^0 \mid X)$, which associates each discrete sequence with continuous states. The resulting target is therefore

$$p(\boldsymbol{z}^0) = \sum_{X \in \Omega^n} P(X) e(\boldsymbol{z}^0 \mid X). \tag{2}$$

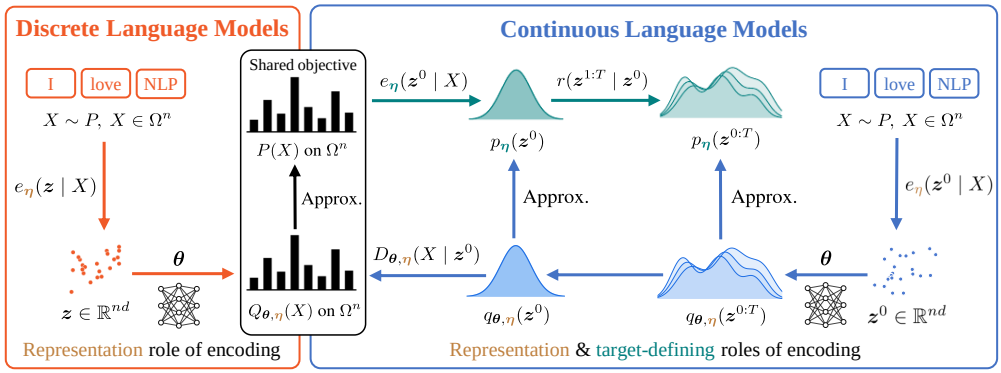

Figure 1: Encoding roles in DLMs and CLMs. Both model families approximate the discrete target P with a model distribution $Q_{\boldsymbol{\theta},\eta}$. In CLMs, encoding additionally defines continuous targets $p_{\boldsymbol{\eta}}$ for q. The approximation arrows summarize distribution matching, not individual NELBO terms.

A common construction of $e(\boldsymbol{z}^0 \mid X)$ adds Gaussian noise to a deterministic encoding map $\mathcal{E} : \Omega^n \to \mathbb{R}^{nd}$. Without initial noise at $\boldsymbol{z}^0$, e reduces to a deterministic encoding, as in DLMs. Choices of $\mathcal{E}$ include one-hot encodings (Richemond et al., 2022; Graves et al., 2023), fixed pretrained token embeddings (Strudel et al., 2022), jointly learned token encodings (Li et al., 2022; Wu et al., 2023; Gong et al., 2022; Dieleman et al., 2022; Gulrajani & Hashimoto, 2023; Yang et al., 2026), or pretrained contextual sequence encodings (Zhou et al., 2025; Hu et al., 2026).

3 The Target-Defining Role of Encoding in CLMs

Section 2 introduces encoding as a common component of DLMs and CLMs. We now distinguish its *representation* role from its additional *target-defining* role in CLMs, as illustrated in Figure 1. The two roles are highlighted in brown and teal, respectively. We formalize this distinction in Section 3.1 and then examine its empirical consequences through controlled experiments in Section 3.2.

3.1 Theoretical Analysis

To distinguish the two encoding roles more clearly, we denote encoding parameters by $\boldsymbol{\eta}$ and all remaining neural parameters by $\boldsymbol{\theta}$. The discrete model distributions are therefore $Q^{\mathrm{DLM}}_{\boldsymbol{\theta},\boldsymbol{\eta}}$ and $Q^{\mathrm{CLM}}_{\boldsymbol{\theta},\boldsymbol{\eta}}$. Throughout, the colors distinguish different roles of the same $\boldsymbol{\eta}$, not separate parameter sets.

In DLMs, encoding serves as an input representation by telling the neural network which token is present. It can therefore affect $Q^{\mathrm{DLM}}_{\boldsymbol{\theta},\boldsymbol{\eta}}$ through representation learning and optimization. However, the discrete target P is defined on Ω^n without reference to any encoding.

For CLMs, the situation is different. The continuous model $q_{\boldsymbol{\theta},\boldsymbol{\eta}}(\boldsymbol{z}^0)$ approximates an encoding-induced target $p_{\boldsymbol{\eta}}(\boldsymbol{z}^0)$. As Equation 2 shows, this target is constructed from P through $e_{\boldsymbol{\eta}}(\boldsymbol{z}^0 \mid X)$, so **it is only defined once the encoding is specified**. Encoding therefore has an additional target-defining role: beyond providing a neural representation, it specifies the continuous target and the space in which noising, denoising, and sampling take place.

The following theorem makes the two encoding roles more explicit by decomposing the objectives in Section 2 into discrete distributional error and variational gaps. We consider the AR, MDM, and diffusion-based CLM formulations in Section 2, with fixed forward processes R, r and a decoder D conditioned only on $\boldsymbol{z}^0$. All distributions are assumed to be normalized, with finite relevant Kullback–Leibler (KL) divergences.

Theorem 1 (Encoding-dependent NELBO decomposition; proof in Appendix A.3)**.** *Under the preceding assumptions, the expected objectives satisfy*

$$\begin{aligned}
\mathcal{L}_{\mathrm{AR}} &= H(P) + D_{\mathrm{KL}}\big(P \,\big\|\, Q^{\mathrm{AR}}_{\boldsymbol{\theta},\eta}\big)\,, \\
\mathcal{L}_{\mathrm{MDM}} &= H(P) + D_{\mathrm{KL}}\big(P \,\big\|\, Q^{\mathrm{MDM}}_{\boldsymbol{\theta},\eta}\big) + \underbrace{\mathbb{E}_{X\sim P}\big[D_{\mathrm{KL}}\big(R(Z^{1:T} \mid X) \,\big\|\, Q^{\mathrm{MDM}}_{\boldsymbol{\theta},\eta}(Z^{1:T} \mid X)\big)\big]}_{\textit{path gap}}, \\
\mathcal{L}_{\mathrm{CLM}} &= H(P) + D_{\mathrm{KL}}\big(P \,\big\|\, Q^{\mathrm{CLM}}_{\boldsymbol{\theta},\eta}\big) + \underbrace{\mathbb{E}_{\boldsymbol{z}^0\sim p_{\eta}}\big[D_{\mathrm{KL}}\big(r(\boldsymbol{z}^{1:T} \mid \boldsymbol{z}^0) \,\big\|\, q_{\boldsymbol{\theta},\eta}(\boldsymbol{z}^{1:T} \mid \boldsymbol{z}^0)\big)\big]}_{\textit{path gap}} \\
&\quad + \underbrace{\mathbb{E}_{X\sim P}\big[D_{\mathrm{KL}}\big(e_{\eta}(\boldsymbol{z}^0 \mid X) \,\big\|\, q_{\boldsymbol{\theta},\eta}(\boldsymbol{z}^0 \mid X)\big)\big]}_{\textit{encoding gap}}.
\end{aligned} \tag{3}$$

Here $H(P)$ is the target entropy, constant with respect to the model and encoding parameters, and the first KL term is the discrete distributional error we ultimately seek to minimize. The nonnegative path gaps compare the forward path distributions with the corresponding model path posteriors, conditioned on samples $X \sim P$ for MDMs and $\boldsymbol{z}^0 \sim p_{\eta}$ for CLMs. The encoding gap similarly measures the discrepancy between the encoding density and the model endpoint posterior.

Equation 3 reorganizes the objectives in Section 2 to expose encoding dependence. For AR and MDM, encoding enters only through $Q_{\boldsymbol{\theta},\eta}$, while the target P remains encoding-independent. These decompositions therefore reflect the representation role without introducing an encoding-defined target. In CLMs, by contrast, the encoding gap requires the model posterior $q_{\boldsymbol{\theta},\eta}(\boldsymbol{z}^0 \mid X)$ to match the conditional target $e_{\eta}(\boldsymbol{z}^0 \mid X)$. The continuous path gap is also averaged over p_{η}, so encoding determines the endpoint states on which path posteriors are compared. These dependencies make the target-defining role explicit: encoding introduces continuous matching requirements not determined by $Q^{\mathrm{CLM}}_{\boldsymbol{\theta},\eta}$ alone. Their gradients need not align with the gradient of the discrete KL, allowing encoding to influence optimization through target construction as well as neural representation.

To see how this distinction affects learning, consider training with two deterministic encodings related by a bijection between their codebooks. In DLMs, the target P remains unchanged, and a sufficiently expressive network parameterized by $\boldsymbol{\theta}$ can in principle learn the inverse map, absorbing the change in η into its internal representation. In CLMs, however, such compensation within the network does not generally undo the changes to e_{η} and p_{η}. This is why an effective neural embedding need not be an effective encoding for continuous modeling. Section 3.2 further examines these different effects of encoding through controlled experiments.

3.2 Empirical Evidence

Having distinguished the representation and target-defining roles of encoding in Section 3.1, we now examine their practical implications. We compare AR, MDM, and a diffusion-based CLM in controlled experiments with aligned vocabulary sizes and Transformer backbones.

We use character-level Text8 (Mahoney, 2011), with $K = 27$ tokens comprising the 26 English letters and the space character, and grayscale CIFAR-10 (Krizhevsky, 2009) as an additional reference whose natural intensity order makes encoding effects easier to examine. We convert CIFAR-10 to grayscale and discretize its pixel intensities into 27 bins, so both datasets are modeled as discrete token sequences with the same vocabulary size. Each experiment uses $N = 5000$ training examples and aligned Transformer backbones, with the CLM implemented as a DDPM (Ho et al., 2020) on the chosen encoding. Further details are given in Appendix B.1.

We first compare one-hot and scalar encodings for both modalities. One-hot encoding maps each token to a 27-dimensional basis vector, whereas scalar encoding assigns one scalar per token, ordered alphabetically for text and by grayscale intensity for images. For images, we additionally use Scalar-Shuffle, which applies a fixed permutation to the token-to-scalar assignment, preserving the scalar values, token identities, and dimension while placing widely separated intensities close together in the encoding space and disrupting the natural intensity order. Appendix C details the construction and additional image-modality shuffle results.

Table 1: Controlled encoding experiments with $K = 27$. (a) Model-family comparison on Text8 and grayscale CIFAR-10, measured by spelling accuracy (SA) and FID (Heusel et al., 2017), respectively. (b) Encoding-geometry comparison for CLM and AR on Text8; d is the encoding dimension.

(a) Model-family comparison

Text8 SA (↑)			
Encoding	AR	MDM	CLM
One-hot	89.29	71.82	71.73
Scalar	88.90	67.61	56.15
CIFAR-10 FID (↓)			
Encoding	AR	MDM	CLM
One-hot	17.00	21.43	24.11
Scalar	17.57	22.31	21.64
Scalar-shuffle	17.36	22.51	111.74

(b) Encoding geometry

Model	Encoding	d	SA (↑)
CLM	Circular	2	64.03
	Clustered-circle	2	57.90
	Scalar	1	56.15
	Scalar-zero-pad	2	58.54
AR	Circular	2	88.66
	Clustered-circle	2	89.53
	Scalar	1	88.90
	Scalar-zero-pad	2	89.16



Figure 2: Samples from the controlled grayscale CIFAR-10 experiment in Section 3.2: (a) validation images; (b,c) samples generated with scalar-shuffle encoding by AR and CLM, respectively.

Table 1(a) shows markedly different encoding effects across model families. On Text8, scalar encoding has a modest effect on DLMs, whereas CLM suffers a substantial drop in spelling accuracy (SA). On grayscale CIFAR-10, scalar encoding remains competitive for CLM, consistent with preserving the intensity order; however, shuffling this order severely degrades CLM generation while AR and MDM remain relatively stable. Figure 2 visualizes this contrast in generated samples.

These results support the distinction between the two encoding roles and demonstrate its practical importance. In DLMs, η affects $Q^{\mathrm{DLM}}_{\boldsymbol{\theta},\eta}$ through the representation role alone. With a small vocabulary and training set, the relatively large backbone can in principle absorb encoding changes by learning an inverse mapping, consistent with the limited sensitivity of AR and MDM while P remains fixed. In contrast, unsuitable encodings can severely degrade CLM generation even in this small-data setting. As discussed in Section 3.1, encoding additionally alters e_η and p_η in ways that cannot generally be absorbed by adjusting the neural representation alone.

This comparison also clarifies the distinction between encoding and embedding. Internal embeddings belong to the neural representation parameterized by $\boldsymbol{\theta}$ and do not define the target distribution. In DLMs, encoding and embedding both serve the representation role and leave P unchanged. In CLMs, by contrast, encoding additionally defines the continuous target p_η, while internal embeddings remain part of the model parameterization. Thus, reusing a DLM embedding as a CLM encoding gives it a target-defining role absent from its original discrete-model use.

Together, the theoretical and empirical analyses show that encoding design in CLMs requires separate consideration from encoding and embedding design in DLMs. However, as discussed in Section 2, some CLMs inherit one-hot vectors or token embeddings from DLMs. Since these representations were not designed to define continuous target distributions, it is unclear whether they provide suitable encodings for continuous modeling. This motivates the central question of Section 4: what properties should a good encoding space have for continuous language models?

4 What Makes a Good Encoding for CLMs?

Sections 3.1 and 3.2 establish, theoretically and empirically, that encoding poses a distinct modeling problem in CLMs. We therefore systematically examine how encoding affects CLM modeling from three complementary perspectives. Sections 4.1 and 4.2 study the geometry and dimension of fixed encodings, while Section 4.3 studies encoding–model compatibility when the encoding is learned jointly with the neural parameterization. Although the gap terms in Equation 3 are generally difficult to evaluate explicitly, the decomposition provides a conceptual guide throughout this section. Together, these analyses guide the real-vocabulary experiments in Section 5.

4.1 Encoding Geometry

We first return to the $K = 27$ Text8 setting. Section 3.2 shows that scalar encoding performs substantially worse than one-hot encoding for CLM. We hypothesize that its geometry contributes to this difference: scalar encoding places tokens asymmetrically along a line segment, whereas one-hot encoding places them symmetrically. To examine geometric effects without introducing a high-dimensional encoding, we construct uniform and clustered circular encodings.

In Circular, the 27 tokens are equally spaced on a unit circle in vocabulary order. As a local-spacing control, Clustered-circle divides the same ordered tokens into three groups of nine and contracts each around its center. Both encodings have unit-norm vectors and zero centroid, but differ in local spacing. Figure 3(c,d) shows their geometries. We also include Scalar-zero-pad, which maps each scalar s to $(s, 0)$, increasing the ambient dimension without changing the line geometry. Appendix B.1 provides detailed experimental settings.

Table 1(b) shows that Circular improves CLM SA from 56.15 with scalar encoding to 64.03, while zero-padding yields only 58.54. Clustering reduces SA to 57.90 despite preserving dimension, encoding-vector norms, and global second moments. By contrast, AR performs similarly across the four encodings. These results indicate that the CLM gains depend on geometry rather than merely increased input dimension, while these geometric differences have much less effect on AR.

To further examine how geometry affects model learning, we corrupt validation sequences at a fixed log-SNR of zero and evaluate the models' predictions of the original tokens. Figure 3(a,b) reports the resulting row-normalized confusion matrices for the uniform and clustered circles, respectively. Rows represent ground-truth tokens and columns represent predicted tokens.

The uniform circle produces a more pronounced diagonal, whereas clustering makes tokens within each group harder to distinguish and concentrates erroneous predictions on the six cluster endpoints. Adjacent endpoint columns appear together, producing four prominent vertical bands. We conjecture that this pattern arises because imbalanced local encoding geometry makes the encoding-dependent matching problems exposed by the gap terms in Equation 3 harder to learn, weakening token distinguishability and biasing predictions toward geometrically exceptional boundary points.

These results suggest that continuous encodings should avoid strong geometric imbalance. One-hot encoding also satisfies this principle because its symmetry gives no token a privileged position, but it requires dimension K. As we show in Section 4.2, this high dimensionality can become problematic as the vocabulary grows, motivating a separate examination of encoding dimension.

4.2 Encoding Dimension

The circular experiment highlights the importance of balanced geometry, but symmetry alone does not ensure a suitable continuous encoding. We therefore study encoding dimension across vocabularies of size $K \in \{27, 128, 1024\}$. For each K and dimension d, we construct an encoding by sampling one vector per token uniformly from the unit sphere in $\mathbb{R}^d$. Each encoding remains fixed during training. This extends the equal-norm circular design in Section 4.1 to higher dimensions without the endpoint asymmetry of scalar encoding.

Figure 3(e) shows a broadly inverted-U relationship between SA and the relative dimension d/K: intermediate dimensions perform best overall, and the best tested relative dimension decreases as K grows. At $K = 1024$, for example, a 32-dimensional sphere encoding performs well, whereas the 1024-dimensional one-hot baseline performs poorly. The stars mark separate one-hot baselines at

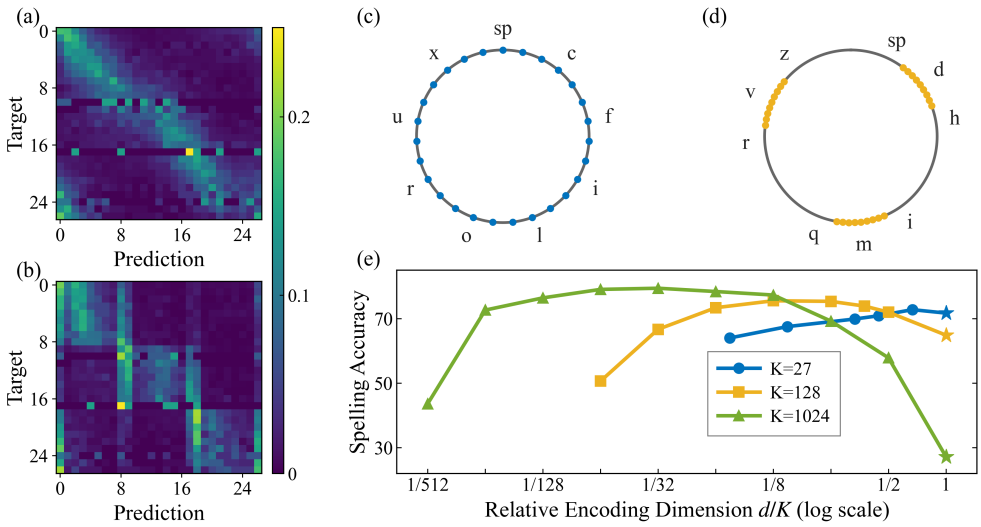

Figure 3: Effects of fixed-encoding geometry and dimension on Text8. (a,b) Row-normalized denoising confusion matrices for the uniform and clustered circular encodings, respectively, at log-SNR 0. (c,d) Geometric layouts of the fixed encodings used in (a) and (b), respectively. (e) SA versus relative encoding dimension d/K for sphere encodings with $K \in \{27, 128, 1024\}$. Stars mark separate one-hot baselines at $d = K$. Numerical results for (e) are reported in Appendix D.1.

$d = K$, highlighting their deterioration as vocabulary size grows. Appendix D.2 shows that random sphere encodings at $d = K = 1024$ perform better than one-hot but remain far below intermediate-dimensional encodings, preserving the inverted-U trend. Four-seed repeats and additional CLM formulations support the same qualitative trend, as reported in Appendices D.2 and D.3.

The poor performance of high-dimensional one-hot encodings further highlights the distinction between encoding and embedding. In DLMs, a K-dimensional one-hot encoding only identifies a token and can be projected to a lower-dimensional embedding without changing P. In CLMs, however, the encoding dimension determines the dimension of the encoding-defined distributions $e_{\boldsymbol{\eta}}$ and $p_{\boldsymbol{\eta}}$ that enter the encoding and path gaps in Equation 3. Learning a compact internal representation through $\boldsymbol{\theta}$ does not reduce the dimension of these targets; with one-hot encoding, the resulting continuous matching problem therefore remains high-dimensional and difficult to learn. In contrast, AR performance changes little across the same encoding families, consistent with this distinction.

These findings motivate the use of sphere encodings with dimensions far below K in the real-vocabulary experiments of Section 5. Geometry and dimension, however, specify only the fixed-encoding design. Section 4.3 therefore turns to learning the encoding jointly with the model and examines how the model's input and output parameterizations can support this mutual adaptation.

4.3 Encoding–Model Compatibility

Sections 4.1 and 4.2 identify useful geometric and dimensional properties of fixed encodings. Beyond these choices, Theorem 1 also motivates learning the encoding jointly with the model, since learning internal embeddings alone cannot alter $e_{\boldsymbol{\eta}}$ or $p_{\boldsymbol{\eta}}$ and therefore cannot take over the target-defining role. The discrete KL is our ultimate modeling objective, whereas the NELBO optimized during training differs from it by additional posterior gaps. Reducing these gaps therefore makes it a tighter surrogate, and jointly learning $\boldsymbol{\eta}$ and $\boldsymbol{\theta}$ provides additional freedom to achieve this reduction.

The explicit form of the encoding gap further clarifies what reducing this discrepancy entails: it directly compares the encoding-defined conditional target $e_{\boldsymbol{\eta}}(\boldsymbol{z}^0 \mid X)$ with the model posterior $q_{\boldsymbol{\theta},\boldsymbol{\eta}}(\boldsymbol{z}^0 \mid X)$. Joint training therefore involves mutual adaptation between the encoding-defined probabilistic structure and the neural parameterization used to model it. We examine this encoding–model compatibility at two interfaces connecting the target-defining and representation roles: how

encoding-induced noisy states are represented as neural inputs, and how neural outputs specify token probabilities and continuous predictions under the current encoding.

Input. We consider two input representations. The first is a posterior-mean representation inspired by Bayesian flow networks (BFNs) (Graves et al., 2023): it averages encoding vectors under the local token posterior defined by the Gaussian encoding channel and a uniform reference prior. The alternative directly rescales the noisy continuous state according to its noise level, following diffusion input preconditioning (Karras et al., 2022). The Transformer architecture is held fixed across the two variants, mapping either input representation to hidden states $\boldsymbol{h}$. For a given encoding and noise schedule, this changes only the neural input representation, not the forward noising process.

Output. For tokenwise encoding, let $\boldsymbol{E}_\eta \in \mathbb{R}^{K\times d}$ be the codebook matrix implementing the encoding map $\mathcal{E}$, with one token encoding vector per row. Let $\boldsymbol{m}$ and $\boldsymbol{\pi}$ denote the predicted continuous mean and token-probability row vector at each position, respectively. Omitting parameter subscripts on neural predictions, we compare three output parameterizations: mean-first ($\boldsymbol{h} \to \boldsymbol{m} \to \boldsymbol{\pi}$), separate heads ($\boldsymbol{h} \to (\boldsymbol{m}, \boldsymbol{\pi})$), and probability-first ($\boldsymbol{h} \to \boldsymbol{\pi} \to \boldsymbol{m} = \boldsymbol{\pi}\boldsymbol{E}_\eta$). We also evaluate adding the encoding channel's Gaussian log-likelihoods to the output logits, following Plaid (Gulrajani & Hashimoto, 2023) and RePlaID (Yang et al., 2026). Detailed input and output formulas are given in Appendix B.3, and Section 5 evaluates these choices in real-vocabulary language modeling.

5 Real-Vocabulary Language Modeling

We now apply the design considerations in Section 4 to progressively improve a CLM in a real-vocabulary setting. We conduct experiments on OpenWebText (Gokaslan & Cohen, 2019) using the GPT-2 BPE tokenizer (Radford et al., 2019), with vocabulary size $K = 50257$. We evaluate generation using generative perplexity (GenPPL), computed by GPT-2 Large, and token entropy as a diversity diagnostic. We also report validation perplexity (ValPPL), obtained by exponentiating the average per-token validation NELBO. Because the posterior gaps in Equation 3 can vary across encodings and model configurations, a lower ValPPL does not necessarily imply a smaller discrete KL; we therefore use it as a complementary likelihood-bound metric. Our model variants have approximately 130M parameters. Additional experimental details are provided in Appendix B.2.

We build on BFNs (Graves et al., 2023) for their strong small-vocabulary character-level language modeling results. Their original categorical formulation is tied to fixed one-hot encoding and does not directly accommodate general encodings. Consistent with our findings in Section 4.2, the one-hot baseline in the fixed-encoding block of Table 2(a) performs poorly at this vocabulary size. However, Xue et al. (2024) establish an equivalent SDE formulation of BFN dynamics through a modest sampling-rule modification. This equivalent SDE formulation enables our extension of BFN-based modeling to general encodings and underlies the following experiments.

We first examine fixed pretrained encodings. We use the GPT-2 Large input embedding table with $d = 1280$ and, motivated by Section 4.2, reduce it to $d = 128$ using principal component analysis (PCA). Both encodings remain fixed during training. As shown in the fixed-encoding block of Table 2(a), replacing one-hot encoding with the pretrained embedding reduces GenPPL from 117 to 92.3 and sharply lowers ValPPL; PCA reduction improves the model further, reaching a GenPPL of 74.3. This reinforces the distinction between encoding and embedding: using a pretrained embedding as a CLM encoding changes the continuous target, not merely the neural input representation. The benefit of PCA further shows that DLM representations need not be suitable continuous encodings in their original form, supporting encoding dimensions far below K.

We next make the encoding trainable. We initialize a $d = 64$ sphere encoding and constrain its token vectors to remain on the unit sphere during training. An ablation against RePlaID's $d = 16$ setting (Yang et al., 2026) in Appendix D.4 favors $d = 64$ for generation in our configuration, consistent with Section 4.2: overly low or high dimensions can hinder learning. The trainable-encoding block of Table 2(a) compares the mean-first, separate-head, and probability-first output parameterizations from Section 4.3. The probability-first parameterization, which sets $\boldsymbol{m} = \boldsymbol{\pi}\boldsymbol{E}_\eta$, performs best among the three. Adding the encoding channel's Gaussian log-likelihoods to the output logits further reduces GenPPL from 28.9 to 23.0.

Beyond the encoding and output interface, the noise schedule determines the noise levels encountered during training and sampling. Following learned-schedule approaches (Sohl-Dickstein et al.,

Table 2: Real-vocabulary language modeling on OpenWebText. (a) Progressive refinements of our model. (b) GenPPL comparison with representative diffusion LMs. Published results use their original sampling protocols and are therefore contextual rather than strictly matched.

(a) Progressive refinements

Configuration	GenPPL ↓	Ent. ↑	ValPPL ↓
Fixed encodings			
One-hot	117	6.14	17800
GPT-2 Large	92.3	6.79	394
GPT-2 Large + PCA	74.3	7.00	198
Trainable encodings			
Mean-first	41.6	7.06	72.4
Separate heads	40.7	6.88	59.6
Probability-first	28.9	7.07	48.7
+ Output evidence	23.0	7.08	47.6
+ Learnable schedule	20.1	7.11	35.5
+ Posterior-mean input	15.8	7.17	43.4

(b) Diffusion LMs

Model	GenPPL ↓
CLMs	
LangFlow (130M)[a]	36.5
AURORA-LM-S (130M)[b]	23.6
ELF-B (105M)[c]	19.7
RePlaID (130M)[⋆]	21.6
Discrete diffusion models	
SEDD (110M)[d]	52.0
MDLM (110M)[d]	46.8
BD3-LM (110M)[d]	25.7
Ours (130M)	**15.8**

[a–c]Published CLM results are from LangFlow (Chen et al., 2026), AURORA-LM-S (Liang et al., 2026), and ELF-B (Hu et al., 2026), respectively. [d]Matched OWT results for SEDD, MDLM, and BD3-LM are reported by Arriola et al. (2025); SEDD and MDLM were originally introduced by Lou et al. (2024) and Sahoo et al. (2024). [⋆]Our RePlaID reimplementation (Yang et al., 2026).

2015; Kingma et al., 2021), we learn its endpoint noise levels and interior shape rather than fixing them, allowing the schedule to adapt alongside the encoding and denoising model. Adding the learnable schedule further reduces GenPPL from 23.0 to 20.1 and ValPPL from 47.6 to 35.5.

Finally, we consider the input representation. The preceding trainable-encoding variants use noise-dependent input preconditioning, a common choice in continuous diffusion models (Karras et al., 2022; Gulrajani & Hashimoto, 2023), which rescales the continuous state before it enters the network. Motivated by the original BFN's local belief input, we instead use the posterior-mean representation from Section 4.3. This final refinement reduces GenPPL from 20.1 to 15.8, while entropy increases slightly from 7.11 to 7.17. Table 2(b) compares the resulting 130M model with representative recent diffusion language models; its GenPPL of 15.8 is lower than the comparable-scale baselines reported there. Generated samples are provided in Appendix D.5.

Together, these results translate the geometric, dimensional, and compatibility insights of Section 4 into practical language modeling. The substantial gains across successive refinements demonstrate the value of treating encoding as a target-defining component rather than merely a neural input representation, guiding effective choices of both encoding and model parameterization. The resulting model outperforms representative diffusion language models of comparable scale in GenPPL, motivating further investigation of encoding design for stronger CLMs.

6 Conclusion

This paper distinguishes encoding's representation role from its additional target-defining role in CLMs by decomposing discrete and continuous training objectives. Controlled experiments show that encoding affects the two model families in qualitatively different ways, supporting encoding design as a distinct problem for CLMs. Building on this distinction, we identify encoding geometry, encoding dimension, and encoding–model compatibility as key design considerations. These insights guide progressive refinements to encodings and neural parameterizations. The resulting model outperforms representative state-of-the-art diffusion language models of comparable scale in GenPPL. Our findings demonstrate the practical value of systematic encoding analysis and motivate further study of encodings tailored to continuous language modeling.

AI USE STATEMENT

In this work, we used generative AI tools to assist with theoretical analysis and proof drafting, hypothesis refinement, experimental design, method implementation and debugging, interpretation of experimental results, and translation. We also used these tools for literature search, manuscript editing, LaTeX preparation, and illustrative figures. The core research question and contributions originated with the authors, although discussions with AI helped refine our reasoning. We reviewed all AI-assisted content and checked the relevant text for potential plagiarism through a manual literature survey. The authors take responsibility for the final manuscript, including all AI-assisted text, claims, code, and figures.

ETHICS STATEMENT

This work studies modeling and representation choices for continuous language models and does not involve human-subject experiments or the collection of new user data. Our experiments use existing public datasets and pretrained model resources. As with other language-generation models trained on web text, the resulting models may reproduce biases, offensive content, or other undesirable patterns present in the training data. Our work is intended as a methodological study rather than a deployment-oriented system, and any downstream use should account for these general risks of language generation.

REPRODUCIBILITY STATEMENT

Appendix A specifies the training objectives and provides the proof of Theorem 1. Appendix B describes the data, encoding constructions, experimental protocols, and input and output parameterizations. Appendices C and D provide additional controls, dimension-sweep results, random-seed checks, and generated samples. Results from our RePlaID reimplementation are explicitly identified in Table 2.

REFERENCES

Josh Achiam, Steven Adler, Sandhini Agarwal, Lama Ahmad, Ilge Akkaya, Florencia Leoni Aleman, Diogo Almeida, Janko Altenschmidt, Sam Altman, Shyamal Anadkat, et al. Gpt-4 technical report. *arXiv preprint arXiv:2303.08774*, 2023.

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://openreview.net/forum?id=tyEyYT267x.

Tom B. Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel M. Ziegler, Jeffrey Wu, Clemens Winter, Christopher Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. Language models are few-shot learners, 2020. URL https://arxiv.org/abs/2005.14165.

Ting Chen, Ruixiang Zhang, and Geoffrey Hinton. Analog bits: Generating discrete data using diffusion models with self-conditioning. *arXiv preprint arXiv:2208.04202*, 2022.

Yuxin Chen, Chumeng Liang, Hangke Sui, Ruihan Guo, Chaoran Cheng, Jiaxuan You, and Ge Liu. Langflow: Continuous diffusion rivals discrete in language modeling, 2026. URL https://arxiv.org/abs/2604.11748.

Sander Dieleman, Laurent Sartran, Arman Roshannai, Nikolay Savinov, Yaroslav Ganin, Pierre H Richemond, Arnaud Doucet, Robin Strudel, Chris Dyer, Conor Durkan, et al. Continuous diffusion for categorical data. *arXiv preprint arXiv:2211.15089*, 2022.

Philip Gage. A new algorithm for data compression. *The C Users Journal archive*, 12:23–38, 1994. URL https://api.semanticscholar.org/CorpusID:59804030.

Aaron Gokaslan and Vanya Cohen. Openwebtext corpus. http://Skylion007.github.io/OpenWebTextCorpus, 2019.

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

Alex Graves, Rupesh Kumar Srivastava, Timothy Atkinson, and Faustino J. Gomez. Bayesian flow networks. *CoRR*, abs/2308.07037, 2023. doi: 10.48550/ARXIV.2308.07037. URL https://doi.org/10.48550/arXiv.2308.07037.

Ishaan Gulrajani and Tatsunori B Hashimoto. Likelihood-based diffusion language models. In A. Oh, T. Naumann, A. Globerson, K. Saenko, M. Hardt, and S. Levine (eds.), *Advances in Neural Information Processing Systems*, volume 36, pp. 16693–16715. Curran Associates, Inc., 2023. doi: 10.52202/075280-0730. URL https://proceedings.neurips.cc/paper_files/paper/2023/file/35b5c175e139bff5f22a5361270fce87-Paper-Conference.pdf.

Daya Guo, Dejian Yang, Haowei Zhang, Junxiao Song, Ruoyu Zhang, Runxin Xu, Qihao Zhu, Shirong Ma, Peiyi Wang, Xiao Bi, et al. Deepseek-r1: Incentivizing reasoning capability in llms via reinforcement learning. *arXiv preprint arXiv:2501.12948*, 2025.

Martin Heusel, Hubert Ramsauer, Thomas Unterthiner, Bernhard Nessler, and Sepp Hochreiter. Gans trained by a two time-scale update rule converge to a local nash equilibrium. In Isabelle Guyon, Ulrike von Luxburg, Samy Bengio, Hanna M. Wallach, Rob Fergus, S. V. N. Vishwanathan, and Roman Garnett (eds.), *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, December 4-9, 2017, Long Beach, CA, USA*, pp. 6626–6637, 2017. URL https://proceedings.neurips.cc/paper/2017/hash/8a1d694707eb0fefe65871369074926d-Abstract.html.

Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. *Advances in neural information processing systems*, 33:6840–6851, 2020.

Keya Hu, Linlu Qiu, Yiyang Lu, Hanhong Zhao, Tianhong Li, Yoon Kim, Jacob Andreas, and Kaiming He. Elf: Embedded language flows, 2026. URL https://arxiv.org/abs/2605.10938.

Tero Karras, Miika Aittala, Timo Aila, and Samuli Laine. Elucidating the design space of diffusion-based generative models. In *Proc. NeurIPS*, 2022.

Diederik Kingma, Tim Salimans, Ben Poole, and Jonathan Ho. Variational diffusion models. *Advances in neural information processing systems*, 34:21696–21707, 2021.

Alex Krizhevsky. Learning multiple layers of features from tiny images. 2009. URL https://api.semanticscholar.org/CorpusID:18268744.

Xiang Li, John Thickstun, Ishaan Gulrajani, Percy S Liang, and Tatsunori B Hashimoto. Diffusion-lm improves controllable text generation. *Advances in Neural Information Processing Systems*, 35:4328–4343, 2022.

Jiajun Liang, Yucheng Liao, Yukang Cao, Jiazhe Wei, Ken Li, Wende Tan, Jiankun Zhang, ZY Cui, Jingkang Yang, Liucheng Guo, Shiqi Yang, B. Yang, Caifeng Shan, Ziwei Liu, and Chenyang Si. AURORA-LM: Autoencoding unified representation for continuous-latent diffusion language modeling, 2026. URL https://arxiv.org/abs/2608.02602.

Xingchao Liu, Chengyue Gong, and qiang liu. Flow straight and fast: Learning to generate and transfer data with rectified flow. In *The Eleventh International Conference on Learning Representations*, 2023. URL https://openreview.net/forum?id=XVjTT1nw5z.

Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization, 2019. URL https://arxiv.org/abs/1711.05101.

Aaron Lou, Chenlin Meng, and Stefano Ermon. Discrete diffusion modeling by estimating the ratios of the data distribution. In *Forty-first International Conference on Machine Learning, ICML 2024, Vienna, Austria, July 21-27, 2024*. OpenReview.net, 2024. URL https://openreview.net/forum?id=CNicRIVIPA.

M Mahoney. Large text compression benchmark. 2011.

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, 2025. URL https://arxiv.org/abs/2502.09992.

Jingyang Ou, Shen Nie, Kaiwen Xue, Fengqi Zhu, Jiacheng Sun, Zhenguo Li, and Chongxuan Li. Your absorbing discrete diffusion secretly models the conditional distributions of clean data. In *ICLR*, 2025. URL https://openreview.net/forum?id=sMyXP8Tanm.

William Peebles and Saining Xie. Scalable diffusion models with transformers. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 4195–4205, 2023.

Alec Radford, Jeff Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. Language models are unsupervised multitask learners. 2019.

Pierre H. Richemond, Sander Dieleman, and Arnaud Doucet. Categorical sdes with simplex diffusion, 2022.

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, 2024. URL https://arxiv.org/abs/2406.07524.

Rico Sennrich, Barry Haddow, and Alexandra Birch. Neural machine translation of rare words with subword units, 2016. URL https://arxiv.org/abs/1508.07909.

Jascha Sohl-Dickstein, Eric Weiss, Niru Maheswaranathan, and Surya Ganguli. Deep unsupervised learning using nonequilibrium thermodynamics. In *International conference on machine learning*, pp. 2256–2265. PMLR, 2015.

Yang Song, Jascha Sohl-Dickstein, Diederik P Kingma, Abhishek Kumar, Stefano Ermon, and Ben Poole. Score-based generative modeling through stochastic differential equations. *arXiv preprint arXiv:2011.13456*, 2020.

Robin Strudel, Corentin Tallec, Florent Altché, Yilun Du, Yaroslav Ganin, Arthur Mensch, Will Grathwohl, Nikolay Savinov, Sander Dieleman, Laurent Sifre, et al. Self-conditioned embedding diffusion for text generation. *arXiv preprint arXiv:2211.04236*, 2022.

Tong Wu, Zhihao Fan, Xiao Liu, Yeyun Gong, Yelong Shen, Jian Jiao, Hai-Tao Zheng, Juntao Li, Zhongyu Wei, Jian Guo, Nan Duan, and Weizhu Chen. Ar-diffusion: Auto-regressive diffusion model for text generation, 2023.

Kaiwen Xue, Yuhao Zhou, Shen Nie, Xu Min, Xiaolu Zhang, Jun Zhou, and Chongxuan Li. Unifying bayesian flow networks and diffusion models through stochastic differential equations. In *Forty-first International Conference on Machine Learning, ICML 2024, Vienna, Austria, July 21-27, 2024*. OpenReview.net, 2024. URL https://openreview.net/forum?id=1jHiq640y1.

Zhihan Yang, Wei Guo, Shuibai Zhang, Subham Sekhar Sahoo, Yongxin Chen, Arash Vahdat, Morteza Mardani, and John Thickstun. Continuous diffusion scales competitively with discrete diffusion for language, 2026. URL https://arxiv.org/abs/2605.18530.

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

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

A Training Objectives and Proof of Theorem 1

This appendix specifies the objectives introduced in Section 2, derives their standard training forms, and proves Theorem 1. All quantities are defined for a full sequence; per-token quantities are obtained by dividing the entire objective by n. All distributions are normalized, and the expected log-density ratios and KL divergences below are assumed to be well-defined and finite.

A.1 Explicit Training Objectives

We omit parameter subscripts in this subsection. The forward processes R, r are fixed, while e, Q, q, and D may depend on the corresponding encoding or model parameters.

AR models. Writing $X_{<i} = (X_1, \dots, X_{i-1})$, the autoregressive model factorizes as

$$Q^{\mathrm{AR}}(X) = \prod_{i=1}^{n} Q^{\mathrm{AR}}(X_i \mid X_{<i}).$$

Its expected negative log-likelihood is

$$\mathcal{L}_{\mathrm{AR}} = -\mathbb{E}_{X\sim P} \log Q^{\mathrm{AR}}(X) = -\mathbb{E}_{X\sim P} \sum_{i=1}^{n} \log Q^{\mathrm{AR}}(X_i \mid X_{<i}). \tag{4}$$

MDMs. Let $Z^0 = X$, with forward masking distribution $R(Z^{1:T} \mid X)$ and normalized generative joint $Q^{\mathrm{MDM}}(X, Z^{1:T})$. The discrete marginal is obtained by summing over the masking path. The expected NELBO is

$$\mathcal{L}_{\mathrm{MDM}} = \mathbb{E}_{X\sim P}\mathbb{E}_{Z^{1:T}\sim R(\cdot|X)} \left[\log \frac{R(Z^{1:T} \mid X)}{Q^{\mathrm{MDM}}(X, Z^{1:T})} \right]. \tag{5}$$

CLMs: prior, denoising, and reconstruction. For Markov forward and reverse processes, the conditional forward path density and continuous model admit the standard diffusion factorizations (Sohl-Dickstein et al., 2015; Ho et al., 2020; Li et al., 2022)

$$p(\boldsymbol{z}^{0:T} \mid X) := e(\boldsymbol{z}^0 \mid X) r(\boldsymbol{z}^{1:T} \mid \boldsymbol{z}^0), \tag{6}$$

and the generative joint is $q(\boldsymbol{z}^{0:T}) D(X \mid \boldsymbol{z}^0)$. Its discrete marginal is Q^{CLM} in Equation 1. The complete expected NELBO is

$$\mathcal{L}_{\mathrm{CLM}} = \mathbb{E}_{X\sim P}\mathbb{E}_{\boldsymbol{z}^{0:T}\sim p(\cdot|X)} \left[\log \frac{p(\boldsymbol{z}^{0:T} \mid X)}{q(\boldsymbol{z}^{0:T}) D(X \mid \boldsymbol{z}^0)} \right]. \tag{7}$$

The numerator includes both the encoding density e and the forward path density r, as specified in Equation 6.

A.2 Standard Training Forms

Parameter subscripts are omitted throughout this subsection.

MDMs: masked-token cross-entropy. For absorbing diffusion with time-independent clean-token predictors, we use a masking-event representation with $T = n$ (Ou et al., 2025). Starting from $Z^0 = X$, the forward process masks one uniformly selected unmasked position at each event, so Z^k contains exactly k mask tokens $\mathrm{M} \notin \Omega$. The reverse chain starts from the fully masked sequence, selects a masked position uniformly, and predicts its token using $Q_i^{\mathrm{MDM}}(\cdot \mid Z^k)$.

Let I_k denote the position masked between Z^{k-1} and Z^k. On every valid path,

$$\begin{aligned} R(Z^{1:n} \mid X) &= \frac{1}{n!}, \\ Q^{\mathrm{MDM}}(X, Z^{1:n}) &= \frac{1}{n!} \prod_{k=1}^{n} Q_{I_k}^{\mathrm{MDM}}(X_{I_k} \mid Z^k). \end{aligned} \tag{8}$$

The factors $1/n!$ cancel in Equation 5. Under the forward process, I_k is uniform over the k masked positions conditional on X, Z^k. Therefore,

$$\mathcal{L}_{\mathrm{MDM}} = -\mathbb{E}_{X\sim P}\mathbb{E}_{Z^{1:n}\sim R(\cdot|X)}\left[\sum_{k=1}^{n}\frac{1}{k}\sum_{i:Z_i^k=\mathrm{M}}\log Q_i^{\mathrm{MDM}}(X_i \mid Z^k)\right]. \tag{9}$$

An unbiased estimator samples k uniformly from $\{1,\dots,n\}$, masks a uniformly selected subset of k positions, and weights the sum of masked-token cross-entropies by n/k.

An equivalent form samples a masking probability $\tau \sim \mathcal{U}(0,1)$. Let $R_\tau(\widetilde{Z} \mid X)$ independently mask each position with probability τ. Then

$$\mathcal{L}_{\mathrm{MDM}} = -\mathbb{E}_{X\sim P}\mathbb{E}_{\tau\sim\mathcal{U}(0,1)}\mathbb{E}_{\widetilde{Z}\sim R_\tau(\cdot|X)}\left[\frac{1}{\tau}\sum_{i:\widetilde{Z}_i=\mathrm{M}}\log Q_i^{\mathrm{MDM}}(X_i \mid \widetilde{Z})\right]. \tag{10}$$

To verify the equivalence, a fixed masked subset of size $k \geq 1$ receives weight

$$\int_0^1 \tau^{k-1}(1-\tau)^{n-k}\,d\tau = \frac{(k-1)!(n-k)!}{n!} = \frac{1}{k\binom{n}{k}},$$

which is exactly its weight in Equation 9. Equation 10 is the weighted masked-token cross-entropy form used by LLaDA (Nie et al., 2025).

CLMs: prior, denoising, and reconstruction. For Markov forward and reverse processes, the conditional forward path density and continuous model factorize as

$$\begin{aligned} p(\boldsymbol{z}^{0:T} \mid X) &= e(\boldsymbol{z}^0 \mid X)\prod_{t=1}^{T} r(\boldsymbol{z}^t \mid \boldsymbol{z}^{t-1}), \\ q(\boldsymbol{z}^{0:T}) &= q(\boldsymbol{z}^T)\prod_{t=1}^{T} q(\boldsymbol{z}^{t-1} \mid \boldsymbol{z}^t). \end{aligned} \tag{11}$$

The conditional forward chain also admits the reverse factorization

$$p(\boldsymbol{z}^{0:T} \mid X) = p(\boldsymbol{z}^T \mid X)\prod_{t=1}^{T} p(\boldsymbol{z}^{t-1} \mid \boldsymbol{z}^t, X).$$

Substituting these factorizations into Equation 7 gives the standard training decomposition:

$$\begin{aligned} \mathcal{L}_{\mathrm{CLM}} &= L_{\mathrm{prior}}^{\mathrm{CLM}} + L_{\mathrm{path}}^{\mathrm{CLM}} + L_{\mathrm{recon}}^{\mathrm{CLM}}, \\ L_{\mathrm{prior}}^{\mathrm{CLM}} &= \mathbb{E}_{X\sim P}\left[D_{\mathrm{KL}}\big(p(\boldsymbol{z}^T \mid X)\,\big\|\,q(\boldsymbol{z}^T)\big)\right], \\ L_{\mathrm{path}}^{\mathrm{CLM}} &= \sum_{t=1}^{T}\mathbb{E}_{X\sim P}\mathbb{E}_{\boldsymbol{z}^t\sim p(\cdot|X)}\left[D_{\mathrm{KL}}\big(p(\boldsymbol{z}^{t-1} \mid \boldsymbol{z}^t, X)\,\big\|\,q(\boldsymbol{z}^{t-1} \mid \boldsymbol{z}^t)\big)\right], \\ L_{\mathrm{recon}}^{\mathrm{CLM}} &= -\mathbb{E}_{X\sim P}\mathbb{E}_{\boldsymbol{z}^0\sim e(\cdot|X)}\log D(X \mid \boldsymbol{z}^0). \end{aligned} \tag{12}$$

The marginals $p(\boldsymbol{z}^t \mid X)$ and bridges $p(\boldsymbol{z}^{t-1} \mid \boldsymbol{z}^t, X)$ are induced by the conditional forward chain in Equation 11. The encoding log-density is already included in this factorization, so no additional encoding-entropy term is required.

These training expressions and Equation 3 describe the same objectives but group their terms differently. The denoising term compares individual reverse transitions with forward bridges, whereas the path gap compares complete forward paths with model path posteriors. Likewise, the reconstruction loss is distinct from the encoding gap.

A.3 Proof of Theorem 1

Proof. For any discrete model distribution Q with positive probability on the support of P,

$$-\mathbb{E}_{X\sim P}\log Q(X) = H(P) + D_{\mathrm{KL}}(P\|Q). \tag{13}$$

AR models. Equation 4 is the exact expected negative log-likelihood. Applying Equation 13 with $Q = Q^{\mathrm{AR}}_{\boldsymbol{\theta},\eta}$ gives the AR identity.

MDMs. The model joint factorizes into its discrete marginal and path posterior:

$$Q^{\mathrm{MDM}}_{\boldsymbol{\theta},\eta}(X, Z^{1:T}) = Q^{\mathrm{MDM}}_{\boldsymbol{\theta},\eta}(X) Q^{\mathrm{MDM}}_{\boldsymbol{\theta},\eta}(Z^{1:T} \mid X).$$

Substituting this factorization into Equation 5 yields

$$\begin{aligned}\mathcal{L}_{\mathrm{MDM}} = &-\mathbb{E}_{X\sim P} \log Q^{\mathrm{MDM}}_{\boldsymbol{\theta},\eta}(X) \\ &+ \mathbb{E}_{X\sim P}\left[D_{\mathrm{KL}}\left(R(Z^{1:T} \mid X) \,\middle\|\, Q^{\mathrm{MDM}}_{\boldsymbol{\theta},\eta}(Z^{1:T} \mid X)\right)\right].\end{aligned} \tag{14}$$

Applying Equation 13 to the first term gives the MDM decomposition.

CLMs. The generative joint $q_{\boldsymbol{\theta},\eta}(\boldsymbol{z}^{0:T}) D_{\boldsymbol{\theta},\eta}(X \mid \boldsymbol{z}^0)$ induces the endpoint posterior

$$q_{\boldsymbol{\theta},\eta}(\boldsymbol{z}^0 \mid X) = \frac{q_{\boldsymbol{\theta},\eta}(\boldsymbol{z}^0) D_{\boldsymbol{\theta},\eta}(X \mid \boldsymbol{z}^0)}{Q^{\mathrm{CLM}}_{\boldsymbol{\theta},\eta}(X)}. \tag{15}$$

Because D depends only on $\boldsymbol{z}^0$, the complete model posterior factorizes as

$$q_{\boldsymbol{\theta},\eta}(\boldsymbol{z}^{0:T} \mid X) = q_{\boldsymbol{\theta},\eta}(\boldsymbol{z}^0 \mid X) q_{\boldsymbol{\theta},\eta}(\boldsymbol{z}^{1:T} \mid \boldsymbol{z}^0). \tag{16}$$

Let $\ell_{\mathrm{CLM}}(X)$ denote the per-example NELBO in Equation 7. Factoring the generative joint into its discrete marginal and posterior gives

$$\begin{aligned}\ell_{\mathrm{CLM}}(X) = &-\log Q^{\mathrm{CLM}}_{\boldsymbol{\theta},\eta}(X) \\ &+ D_{\mathrm{KL}}\left(e_{\boldsymbol{\eta}}(\boldsymbol{z}^0 \mid X) r(\boldsymbol{z}^{1:T} \mid \boldsymbol{z}^0) \,\middle\|\, q_{\boldsymbol{\theta},\eta}(\boldsymbol{z}^{0:T} \mid X)\right).\end{aligned} \tag{17}$$

Using Equation 16, the KL chain rule gives

$$\begin{aligned}&D_{\mathrm{KL}}\left(e_{\boldsymbol{\eta}}(\boldsymbol{z}^0 \mid X) r(\boldsymbol{z}^{1:T} \mid \boldsymbol{z}^0) \,\middle\|\, q_{\boldsymbol{\theta},\eta}(\boldsymbol{z}^{0:T} \mid X)\right) \\ &\quad = D_{\mathrm{KL}}\left(e_{\boldsymbol{\eta}}(\boldsymbol{z}^0 \mid X) \,\middle\|\, q_{\boldsymbol{\theta},\eta}(\boldsymbol{z}^0 \mid X)\right) \\ &\qquad + \mathbb{E}_{\boldsymbol{z}^0\sim e_{\boldsymbol{\eta}}(\cdot\mid X)}\left[D_{\mathrm{KL}}\left(r(\boldsymbol{z}^{1:T} \mid \boldsymbol{z}^0) \,\middle\|\, q_{\boldsymbol{\theta},\eta}(\boldsymbol{z}^{1:T} \mid \boldsymbol{z}^0)\right)\right].\end{aligned} \tag{18}$$

Averaging Equation 17 over $X \sim P$, the negative log-likelihood becomes $H(P) + D_{\mathrm{KL}}(P \| Q^{\mathrm{CLM}}_{\boldsymbol{\theta},\eta})$. The first term on the right-hand side of Equation 18 gives the encoding gap. For the remaining term, Equation 2 gives

$$\sum_X P(X) e_{\boldsymbol{\eta}}(\boldsymbol{z}^0 \mid X) = p_{\boldsymbol{\eta}}(\boldsymbol{z}^0),$$

so its outer expectation becomes $\mathbb{E}_{\boldsymbol{z}^0\sim p_{\boldsymbol{\eta}}}$. This proves the CLM decomposition. All posterior gaps are nonnegative because they are expectations of KL divergences. □

B Experimental Details

B.1 Controlled Experiments in Sections 3–4.2

Data and models. For character-level Text8 (Mahoney, 2011), we use the space character and 26 English letters, giving $K = 27$. The corpus is divided into non-overlapping sequences of length $n = 256$, with 5000 sequences in each of the disjoint training, validation, and test sets. CIFAR-10 (Krizhevsky, 2009) images are converted to grayscale, downsampled to 16×16, and quantized into 27 intensity bins, again using 5000 examples per split. For larger-vocabulary Text8 experiments, BPE (Gage, 1994; Sennrich et al., 2016) tokenizers with $K \in \{128, 1024\}$ are trained on the full corpus. The models use Transformer backbones with hidden size 768, 24 layers, and 12 attention heads, with approximately 170M parameters. The continuous and masked diffusion implementations follow DiT (Peebles & Xie, 2023) and RADD (Ou et al., 2025), respectively. Each small-scale model is trained on one GPU.

Encoding constructions. For the $K = 27$ constructions, we use implementation IDs $k = 0, \ldots, 26$, with space first for Text8. ID k corresponds to token $k + 1$ under the convention in Section 2. Scalar encoding assigns equally spaced values in alphabetical order for text and intensity order for images, centered at zero and rescaled to have unit mean squared magnitude. Denoting these values by s_k, SCALAR-ZERO-PAD maps s_k to $(s_k, 0)$. For the circular encodings, write $k = 9r + j$, with $r \in \{0, 1, 2\}$ and $j \in \{0, \ldots, 8\}$, and define

$$\begin{aligned} \phi_{r,j}(\rho) &= \frac{2\pi r}{3} + \rho(j-4)\frac{2\pi}{27} + \frac{8\pi}{27}, \\ \boldsymbol{e}_{9r+j}(\rho) &= \big(\sin \phi_{r,j}(\rho), \cos \phi_{r,j}(\rho)\big). \end{aligned} \tag{19}$$

We use $\rho = 1$ for CIRCULAR and $\rho = 0.3$ for CLUSTERED-CIRCLE. Both encodings have unit row norms, zero centroid, and $\boldsymbol{E}^\top \boldsymbol{E}/K = \boldsymbol{I}_2/2$. Random sphere encodings are constructed by normalizing independently sampled standard Gaussian vectors and remain fixed during training.

Geometry diagnostics. Figure 3(a,b) uses the raw checkpoints at step 170,800 for the uniform and clustered circular encodings. We corrupt 64 validation sequences at log-SNR 0, predict their continuous means, and decode each prediction to its nearest encoding vector. The confusion matrices are row-normalized by the ground-truth token counts and share the same color scale.

B.2 REAL-VOCABULARY EXPERIMENTS IN SECTION 5

Data and architecture. We use the OpenWebText (Gokaslan & Cohen, 2019) preprocessing and training/validation split from our RePlaID reimplementation, with the GPT-2 BPE tokenizer (Radford et al., 2019) and $K = 50257$. Sequences have length 1024. Our models use a Transformer with 12 layers, hidden size 768, 12 attention heads, and dropout 0.1, with approximately 130M parameters depending on the encoding and output interface.

Encodings. The fixed-encoding block of Table 2(a) compares one-hot encoding with two encodings derived from the GPT-2 Large input embedding table. The original embedding has $d = 1280$ and is normalized to unit norm per row. For the PCA variant, we center the original embedding table by subtracting its column mean, project onto its leading 128 principal components without whitening, and normalize each projected row. The trainable-encoding block uses $d = 64$ sphere encodings with shared random initialization and unit row norms throughout training. The RePlaID reimplementation and the lower-dimensional ablations use $d = 16$.

Training. We use AdamW (Loshchilov & Hutter, 2019) with $(\beta_1, \beta_2) = (0.9, 0.999)$, a global batch size of 512, and a 2500-step learning-rate warmup followed by constant learning rates. The backbone and input projection use 3×10^{-4}; trainable encodings, noise-schedule endpoints, and the schedule-shape network use 10^{-2}. Weight decay is 0.1 for endpoint parameters and zero otherwise. Evaluation uses an exponential moving average of the parameters with decay 0.9999. All configurations in Table 2(a) are trained for 70,000 optimization steps on two GPUs, while the RePlaID reimplementation uses four GPUs. Self-conditioning (Chen et al., 2022) is applied at the sequence level with probability 0.25, with stop-gradient through the encoding parameters on the self-conditioning path.

Evaluation. Using an exponential moving average (EMA) of the parameters, we evaluate on 131072 validation tokens and generate 256 sequences of length 1024 with 1024 ancestral sampling steps and terminal cutoff 10^{-3}. Generation uses terminal tokenwise maximum a posteriori (MAP) decoding, and both GenPPL and entropy are computed from the resulting token sequences. GPT-2 Large scores these token IDs directly. GenPPL is the arithmetic mean of per-sequence perplexities, while entropy is the unigram entropy of the pooled generated tokens, measured in nats. ValPPL is the exponential of the complete per-token validation NELBO, including prior, denoising, and reconstruction terms.

B.3 INPUT AND OUTPUT FORMULAS

Gaussian channel and local evidence. Fix a token position i, and suppress the position index on continuous row vectors. Let $\boldsymbol{e}_k$ denote the encoding vector for token $k \in \Omega$, with encoding-

parameter dependence implicit on individual vectors. Consider the isotropic Gaussian channel

$$\boldsymbol{z}^t = \alpha_t \boldsymbol{e}_{X_i} + \sigma_t \boldsymbol{\epsilon}, \qquad \boldsymbol{\epsilon} \sim \mathcal{N}(0, \boldsymbol{I}_d), \tag{20}$$

where $\alpha_t > 0$ and $\sigma_t > 0$ are the signal and noise scales. No constraint such as $\alpha_t^2 + \sigma_t^2 = 1$ is required. Expanding the Gaussian log-likelihood and dropping terms independent of k gives

$$\ell_{\boldsymbol{\eta},k}(\boldsymbol{z}^t, t) = \frac{\alpha_t}{\sigma_t^2} \boldsymbol{z}^t \boldsymbol{e}_k^\top - \frac{\alpha_t^2}{2\sigma_t^2} \|\boldsymbol{e}_k\|^2. \tag{21}$$

Under a uniform reference prior, the local token posterior is $\boldsymbol{b}_{\boldsymbol{\eta}} = \mathrm{softmax}(\boldsymbol{\ell}_{\boldsymbol{\eta}})$, where $\boldsymbol{\ell}_{\boldsymbol{\eta}}$ collects the tokenwise log-likelihoods $\ell_{\boldsymbol{\eta},k}$. Equivalently, defining $\boldsymbol{y} = \alpha_t \boldsymbol{z}^t / \sigma_t^2$ and $s = \alpha_t^2 / \sigma_t^2$ gives $\ell_{\boldsymbol{\eta},k} = \boldsymbol{y}\boldsymbol{e}_k^\top - s\|\boldsymbol{e}_k\|^2/2$.

Input representations. For the unit-norm encodings used in our real-vocabulary experiments, we compare

$$\boldsymbol{u}_{\mathrm{state}} = \frac{\boldsymbol{z}^t}{\sqrt{\sigma_t^2 + \alpha_t^2/d}}, \qquad \boldsymbol{u}_{\mathrm{belief}} = \boldsymbol{b}_{\boldsymbol{\eta}} \boldsymbol{E}_{\boldsymbol{\eta}}. \tag{22}$$

The first rescales the continuous state, while the second represents it by the local posterior mean of the encoding vector, without further rescaling. The Transformer processes the chosen input together with noise-level and self-conditioning information to produce $\boldsymbol{h}$.

Output parameterizations. Let g_m and g_π denote neural output heads. Omitting their parameter dependence, we compare

$$\begin{aligned} \text{Mean-first:} \quad & \boldsymbol{m} = g_m(\boldsymbol{h}), \qquad \pi_k \propto \exp\left(-\frac{\|\boldsymbol{m} - \boldsymbol{e}_k\|^2}{2\nu^2}\right), \\ \text{Separate heads:} \quad & \boldsymbol{m} = g_m(\boldsymbol{h}), \qquad \boldsymbol{\pi} = \mathrm{softmax}(g_\pi(\boldsymbol{h})), \\ \text{Probability-first:} \quad & \boldsymbol{\pi} = \mathrm{softmax}(g_\pi(\boldsymbol{h})), \qquad \boldsymbol{m} = \boldsymbol{\pi}\boldsymbol{E}_{\boldsymbol{\eta}}. \end{aligned} \tag{23}$$

The mean-first probabilities are normalized over the vocabulary. We use $\nu^2 = 1/4$ for its denoising distance head.

Output evidence and noise schedule. For the probability-first interface with output evidence, we use

$$\boldsymbol{\pi} = \mathrm{softmax}\big(g_\pi(\boldsymbol{h}) + \lambda(u)\boldsymbol{\ell}_{\boldsymbol{\eta}}(\boldsymbol{z}^t, t)\big), \qquad \boldsymbol{m} = \boldsymbol{\pi}\boldsymbol{E}_{\boldsymbol{\eta}}, \tag{24}$$

following Plaid and RePlaID (Gulrajani & Hashimoto, 2023; Yang et al., 2026). When evidence is enabled, its weight is $\lambda(u) = \min(u/5000, 1)$ at optimizer step u during training and $\lambda = 1$ at evaluation; otherwise, $\lambda = 0$. The negative log-SNR $\gamma_t = \log(\sigma_t^2/\alpha_t^2)$ has initial clean and noisy endpoint values of -3 and 6. The schedule remains fixed through the + Output evidence configuration and becomes learnable for + Learnable schedule and + Posterior-mean input in Table 2(a).

C Additional Shuffle Experiments

In the grayscale experiment, SCALAR-SHUFFLE assigns the intensity bins listed below to scalar coordinates $0, \ldots, 26$, in order, before the same scalar normalization used by the unshuffled encoding:

$$13, 26, 0, 25, 1, 24, 2, 23, 3, 22, 4, 21, 5, 20,$$
$$6, 19, 7, 18, 8, 17, 9, 16, 10, 15, 11, 14, 12.$$

Thus, bin 13 is assigned coordinate 0, bin 26 coordinate 1, and bin 0 coordinate 2. This places visually distant intensities next to one another. By contrast, reversing the intensity order or cyclically shifting it preserves all or most neighboring intensity pairs. These controls have much smaller effects on CLM FID than scalar shuffling (Table 3).

D Additional Quantitative Results and Generated Samples

D.1 Encoding-Dimension Sweep

Table 4 gives the numerical results underlying Figure 3(e), with scalar and one-hot references included separately.

Table 3: Grayscale scalar-order controls for CLM. Scalar-shuffle FID is reported in Table 1(a).

Encoding order	FID (↓)
Original	21.64
Reverse	22.45
Cyclic-1	23.13
Cyclic-5	25.41
Cyclic-14	22.96

Table 4: SA in the encoding-dimension sweep. Dashes denote unreported configurations.

$K = 27$			$K = 128$			$K = 1024$		
Encoding	d	SA	Encoding	d	SA	Encoding	d	SA
Scalar	1	56.15	Scalar	1	35.73	Scalar	1	39.72
One-hot	27	71.73	One-hot	128	64.89	One-hot	1024	27.18
Sphere	2	63.58	Sphere	2	50.64	Sphere	2	43.64
Sphere	4	67.55	Sphere	4	66.71	Sphere	4	72.67
Sphere	9	69.90	Sphere	8	73.40	Sphere	8	76.45
Sphere	12	70.91	Sphere	16	75.56	Sphere	16	79.08
Sphere	18	72.80	Sphere	32	75.37	Sphere	32	79.44
–	–	–	Sphere	48	73.90	Sphere	64	78.42
–	–	–	Sphere	64	72.02	Sphere	128	77.33
–	–	–	–	–	–	Sphere	256	69.26
–	–	–	–	–	–	Sphere	512	57.87

D.2 Random Seeds and High-Dimensional Sphere Encodings

We repeat the $K = 1024$ sweep with four random seeds, using random sphere encodings at every dimension, including $d = K$. Table 5 reports the mean and standard deviation of SA. The high-dimensional decline persists without substituting one-hot encoding for the sphere encoding at the endpoint.

D.3 Encoding Dimension across Continuous Model Formulations

Table 6 reports additional $K = 1024$ Text8 experiments using BFNs (Graves et al., 2023), score-based diffusion (Song et al., 2020), and rectified flow (Liu et al., 2023). All three formulations prefer intermediate dimensions within the tested range. These are within-model dimension comparisons, not a matched-budget ranking of model families.

D.4 Encoding Dimension in Real-Vocabulary Models

We compare trainable unit-sphere encodings with $d \in \{16, 64\}$ under both the preconditioned-state and posterior-mean inputs. The output uses the probability-first interface with Gaussian evidence and a trainable noise schedule. Reporting both inputs distinguishes dimensional effects from their interaction with the input representation. Table 7 reports GenPPL, entropy, and ValPPL at matched evaluation and training budgets. Across both input parameterizations, increasing d from 16 to 64 improves GenPPL with a modest reduction in entropy. The posterior-mean input further improves both GenPPL and entropy at either dimension, although it yields higher ValPPL.

D.5 Generated Samples from Section 5

Figure 4 presents selected samples from the final posterior-mean-input configuration in Table 2(a). Their individual GenPPL values are illustrative and do not replace the aggregate evaluation in the main table.

Table 5: SA of random sphere encodings at $K = 1024$, reported as mean $\pm$ standard deviation over four seeds.

d	2	8	32	128	512	1024
SA(↑)	43.34 ± 0.91	76.15 ± 0.95	78.66 ± 0.96	77.16 ± 1.14	62.11 ± 2.93	40.66 ± 0.23

Table 6: SA across encoding dimensions for additional continuous model formulations on Text8 with $K = 1024$.

Model	$d = 2$	$d = 4$	$d = 8$	$d = 32$	$d = 128$	$d = 512$
BFN (Graves et al., 2023; Xue et al., 2024)	44.37	62.28	77.58	76.63	71.83	66.71
Score-based diffusion (Song et al., 2020)	42.84	50.43	64.12	50.68	48.73	48.91
Rectified flow (Liu et al., 2023)	42.82	65.80	73.82	73.50	50.30	48.36

Table 7: Encoding-dimension ablation in real-vocabulary language modeling under a common evaluation protocol and matched training budget.

Input	d	GenPPL (↓)	Entropy (↑)	ValPPL (↓)
Preconditioned state	16	23.0	7.17	32.7
Preconditioned state	64	20.1	7.11	35.5
Posterior mean	16	19.1	7.32	33.3
Posterior mean	64	15.8	7.17	43.4

The EU agreement marks the first step to bind the EU and Britain to build a country and then sell large number of mobile phones and Internet services in excess of one billion people. The agreement, which sets broad international trade guidelines for the EU and also between the UK and other European nations, includes a common agreement between European partners over the mutualisation of the barriers created by the task of maintaining global competitiveness and security. It is also a shift in the discussions between those countries and other partners. At the British level, continents have been enjoying stronger trade relations with Europe. In addition to the economic problems faced by the other European countries, the UK needs to be aware that trade and trading relations with those countries play an important role in ensuring global prosperity and security.Are your parents passionate about their next existence? In a new study from the Six Report Journal of Aging, hundreds of grandchildren answer dire questions about what happens next. Tens of thousands of them may want to find ways to build a happy relationship with each other's children in the future prior to their two sons died in hospital. "A lot of them would agree that living their relationships and being married has no idea what parents think about, and therefore doesn't reflect how close generations is in it," said Philip Chailar, director of Development Policy, in the Understanding of Family Support, at a research center at the University of Georgia in Austin, TX. "A lot of them would agree that living their relationships and being shared, is no

They also are voicing diplomatic concerns from several other Pacific countries. The Hamza family began dialogue on Friday after announcing their delegations to South Korea, worried that North Korea would boost its ability to develop its nuclear and ballistic weapons. Democratic vice presidential candidate Bernie Sanders said on Friday that the timing of the U.S. meeting with South Korean peacekeepers in Laos was in doubt. "I am disappointed that North Korea is not making great progress toward increasing its ability to develop its nuclear weapons and improving our U.S. ability to acquire its nuclear weapons," he said in an interview with Fox News. "I believe that the U.S. is on track to deliver its message to Japan, increasing its ability to use its nuclear and ballistic weapons, and our allies." A resolution approved on Thursday asked North Korean leaders to fulfill their commitments and the possibility of spreading their message to Japan and improving North Korea's economy. A U.S. meeting with South Korean officials in Laos and Japan's Prime Minister, Wang Yi, has been a cornerstone of the U.S. session. Olivia said late Friday that North Korea planned to hold its delegations with the U.S., addressing South Asian concerns. It will occur for the first time since the meeting is a dialogue with North Korea. "The Hamza family wants to boost this delegation with the U.S. to stop North Korea addressing South Asian concerns," Wang said in an interview with The Daily Beast. Olivia said Wang was on track to deliver his message to Japan, adding that South Korea and Japan were

Figure 4: Selected samples from the posterior-mean-input variant. Individual GenPPL values are 13.15 (top) and 19.26 (bottom).