From Attention to Atoms: Spectral Dictionary Learning for Fast, Interpretable Language Models

Anonymous Author(s)
Affiliation
Address
email

Abstract

We propose a novel spectral generative modeling framework for natural language processing that jointly learns a global time-varying Fourier dictionary and per-token mixing coefficients, replacing the ubiquitous self-attention mechanism in transformer architectures. By enforcing reconstruction losses in both the time domain (embedding reconstruction) and the frequency domain (via Short-Time Fourier Transform magnitude matching) alongside a standard language modeling objective, and fitting a Gaussian Mixture Model (GMM) prior over the learned mixing vectors, our approach achieves competitive perplexity and generation quality on standard benchmarks such as WikiText-2 and Penn Treebank. In contrast to $\mathcal{O}(L^2)$ self-attention, our method operates with $\mathcal{O}(KL)$ complexity, where $K \ll L$ is the dictionary size, delivering substantial efficiency gains. We demonstrate that spectral dictionary models can achieve competitive performance compared to transformer baselines while significantly reducing inference latency and memory footprint, offering a compelling alternative for scalable language modeling.

Keywords: Spectral Dictionary, Dictionary Learning, Short-Time Fourier Transform, Gaussian Prior, Pointer-Generator

1 Introduction

The advent of the Transformer architecture [17] revolutionized sequence modeling by replacing recurrent and convolutional operations with self-attention mechanisms that directly capture dependencies across arbitrary token distances. Building on this foundation, bi-directional encoders like BERT [6] and autoregressive language models such as the GPT series [15] have achieved state-of-the-art results on a wide range of natural language processing tasks. These models rely on the full $L \times L$ attention matrix, where L is the input sequence length, to compute pairwise interactions between tokens. Although highly expressive, this quadratic complexity in both computation and memory becomes prohibitive when scaling to very long contexts, such as entire documents or long code sequences [3, 19].

To mitigate the cost of full self-attention, a variety of approximations have been proposed. Sparse attention patterns exploit locality or fixed windowing, as in Longformer [2] and the block-sparse model of Child et al. [3]; kernel-based methods like Performer [4] use randomized feature maps to approximate softmax attention in linear time; low-rank factorization approaches such as Linformer [18] and linearized attention via kernel methods [8] project keys and queries into subspaces of dimension $K \ll L$. Other innovations include locality-sensitive hashing in Reformer [12] and learned mixture-of-experts routing to sparsify computation across heads.

Parallel to these, spectral mixing approaches replace learned attention maps with fixed or learned transforms in the Fourier domain. FNet [13] demonstrated that a single global Fourier transform can

Submitted to 39th Conference on Neural Information Processing Systems (NeurIPS 2025). Do not distribute.

approximate the mixing power of self-attention, yielding $\mathrm{O}(L \log L)$ complexity but with limited adaptability to specific token interactions. Motivated by the efficiency of spectral methods and the expressivity of learned transforms, we propose a fully spectral generative model that learns a global dictionary of K complex-valued Fourier atoms whose amplitude, frequency, and phase parameters adapt dynamically across sequence positions.

In our *Spectral Dictionary Generative Model* (SDGM), a lightweight convolutional encoder computes per-token mixing coefficients that weight the contribution of each atom to the embedding reconstruction. We train the model end-to-end to reconstruct original token embeddings via a combined loss: mean-squared error (MSE) in the time (embedding) domain, Short-Time Fourier Transform (STFT) magnitude loss to preserve local frequency structure, and a standard language modeling loss. After training, we flatten the learned mixing coefficient vectors across tokens and fit a Gaussian Mixture Model (GMM), enabling rich, multimodal sampling for text generation. By choosing $K \ll L$, SDGM achieves $\mathcal{O}(KL)$ time and memory complexity per sequence, dramatically reducing resource requirements compared to full attention, while achieving competitive perplexities on standard language modeling benchmarks.

Our contributions are as follows:

1. We introduce a novel spectral dictionary architecture that learns interpretable Fourier atoms parameterized by amplitude, frequency, and phase, enabling efficient global mixing with linear complexity ($\mathcal{O}(KL)$).
2. We propose a dual-domain reconstruction objective, combining time-domain MSE with frequency-domain STFT magnitude loss, alongside a standard language modeling loss, to ensure both embedding fidelity and predictive performance.
3. We demonstrate that fitting a GMM to the learned mixing vectors yields a latent distribution suitable for text generation, complementing the autoregressive nature of the model.
4. We validate that SDGM achieves competitive perplexity compared to Transformer baselines on PTB and WikiText-2 while offering significant reductions in memory footprint and inference latency.

2 Related Work

Fourier and Spectral Methods Fourier transforms have been employed in efficient sequence modeling [13, 7], often as submodules within attention blocks or as fixed transformations replacing the attention mechanism entirely. FNet [13] used unparameterized 2D Fourier transforms. Our work extends spectral approaches by learning an explicit, parameterized Fourier dictionary optimized end-to-end specifically for language modeling reconstruction and generation. While spectral methods have found applications in image generation [10] and wavelet transforms have been explored as attention alternatives [11], our approach uniquely adapts spectral dictionary learning, with learnable sinusoidal parameters and per-token coefficients, to the sequential nature of language.

Attention Alternatives Numerous techniques aim to reduce attention's $\mathcal{O}(L^2)$ computational cost, including sparse attention [2, 3], kernel methods like Performer [4], low-rank projections like Linformer [18], and hashing methods like Reformer [12]. Unlike these approaches that primarily approximate the standard attention mechanism, SDGM replaces attention entirely with a learned spectral mixing paradigm, offering a fundamentally different approach to sequence interaction modeling.

Dictionary Learning Classical dictionary learning, prominent in vision and audio processing [1], often involves learning an overcomplete basis (dictionary) and finding sparse representations (codes) for signals, typically optimized via alternating minimization or algorithms like K-SVD. We adapt dictionary learning concepts to NLP by learning continuous, parameterized Fourier atoms and soft mixing coefficients within an autoencoder-like framework trained with gradient descent. Unlike traditional sparse coding that often enforces L_1 regularization on codes, our approach models the distribution of the learned mixing coefficients using a GMM, facilitating generative sampling.

3 Mathematical Formulation

In this section, we provide a comprehensive derivation of our Spectral Dictionary Generative Model (SDGM). We begin by defining the embedding sequence and progressing through dictionary parameterization, coefficient encoding, reconstruction decoding, loss formulation, and latent prior modeling. Figure 1 illustrates the end-to-end flow of SDGM architecture, showing how raw tokens are progressively transformed into an output distribution.

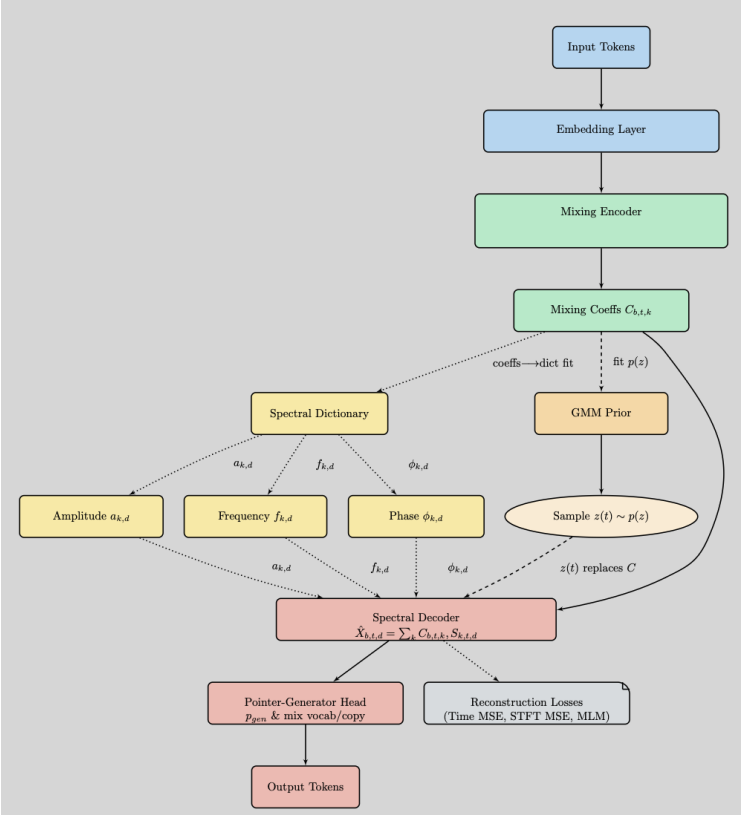

Figure 1: Architecture of the Spectral Dictionary Generative Model. First, the embedding layer maps each input token $w_{b,t}$ to a continuous vector $\mathbf{x}_{b,t} = E(w_{b,t})$. Next, the mixing encoder applies a one-dimensional convolution to produce soft coefficients $C_{b,t,k}$. The spectral dictionary holds K learnable Fourier atoms parameterized by amplitude $a_{k,d}$, frequency $f_{k,d}$, and phase $\phi_{k,d}$, which generate basis vectors $S_{k,t,d}$. The spectral decoder then reconstructs embeddings via $\hat{X}_{b,t,d} = \sum_{k=1}^{K} C_{b,t,k}\, S_{k,t,d}$. Finally, the pointer-generator head combines each reconstructed vector $\hat{\mathbf{x}}_{b,t}$ with a context vector $\mathbf{c}_{b,t}$ to compute a mixture of vocabulary and copy distributions for token prediction.

3.1 Token Embeddings and Notation

Let $\mathbf{W} = [w_1, w_2, \ldots, w_L]$ be an input sequence of L tokens. We first map these tokens to continuous vector representations using an embedding lookup table E. For a mini-batch of B sequences, let $\mathbf{X}^{(b)} = [\mathbf{x}_{b,1}, \mathbf{x}_{b,2}, \ldots, \mathbf{x}_{b,L}] \in \mathbb{R}^{D \times L}$ denote the sequence of L token embeddings for batch item b, where each embedding $\mathbf{x}_{b,t} \in \mathbb{R}^D$ has dimension D.

$$\mathbf{x}_{b,t} = E(w_{b,t}). \tag{1}$$

For clarity, we omit the batch index b in subsequent notation unless explicitly needed.

3.2 Global Spectral Dictionary Parameterization

We learn a set of K spectral atoms, each atom $k \in \{1, \ldots, K\}$ parameterized by three matrices:

- Amplitude: $\mathbf{A} \in \mathbb{R}^{K \times D}$ with entries $a_{k,d}$,
- Frequency: $\mathbf{F} \in \mathbb{R}^{K \times D}$ with entries $f_{k,d}$,
- Phase: $\mathbf{\Phi} \in \mathbb{R}^{K \times D}$ with entries $\phi_{k,d}$.

These parameters define a time-varying sinusoidal basis. For a continuous time index $t \in \{1, 2, \ldots, L\}$, the d-th feature of atom k is given by:

$$S_k(t)_d \;=\; a_{k,d}\,\sin\!\Big(2\pi\, f_{k,d}\,\frac{t}{L} + \phi_{k,d}\Big). \tag{2}$$

We collect all atoms into a tensor $S \in \mathbb{R}^{K \times L \times D}$ where $S_{k,t,d} = S_k(t)_d$. Equivalently, by defining the normalized time vector $\mathbf{t} = [1/L, 2/L, \ldots, 1] \in \mathbb{R}^L$, we can write in vectorized form for fixed atom k:

$$S_k = \mathbf{a}_k \odot \sin\big(2\pi\,(\mathbf{f}_k \otimes \mathbf{t}) + \boldsymbol{\phi}_k \otimes \mathbf{1}_L\big) \tag{3}$$

where $\odot$ denotes element-wise multiplication and $\otimes$ the outer product. This dictionary is shared across all sequences in a batch and represents a global basis learned from the data.

3.3 Mixing Coefficient Encoding

To capture how each atom's contribution varies dynamically based on the input sequence context, we employ a lightweight convolutional encoder. This encoder takes the sequence of input embeddings $\mathbf{X} \in \mathbb{R}^{B \times L \times D}$ (transposed for Conv1D compatibility if needed) and produces per-token mixing coefficients.

$$C = \sigma_{\text{act}}(\text{Conv1D}(\mathbf{X})) \in \mathbb{R}^{B \times L \times K}. \tag{4}$$

Here, Conv1D is a 1D convolution (typically causal for autoregressive tasks) with appropriate kernel size w and padding, mapping the D-dimensional embeddings to K coefficients per time step t. σ_{act} is a suitable activation function (e.g., ReLU or identity, depending on whether non-negativity is desired). Optionally, coefficients $C_{b,t,:}$ can be normalized (e.g., via Softmax) across the dictionary dimension k. These coefficients $C_{b,t,k}$ represent the learned "importance" or "weight" of atom k at position t for sequence b.

3.4 Spectral Reconstruction Decoder

Given the mixing coefficients C and the global spectral dictionary S, we reconstruct the embedding sequence $\hat{\mathbf{X}}$ via a weighted sum of the spectral atoms at each time step t and for each feature dimension d:

$$\hat{\mathbf{X}}_{b,t,d} \coloneqq \sum_{k=1}^{K} C_{b,t,k}\, S_{k,t,d}. \tag{5}$$

This operation performs the dynamic mixing of the global atoms based on the sequence-specific coefficients. In tensor notation, assuming S is appropriately shaped (e.g., $K \times L \times D$), this corresponds to a bilinear mapping, efficiently computed using Einstein summation convention:

$$\hat{\mathbf{X}} = \text{einsum}('blk, kld- > bld', C, S), \tag{6}$$

where dimensions correspond to (Batch, Length, Dictionary) for C and (Dictionary, Length, Dimension) for S, resulting in $\hat{\mathbf{X}}$ with shape (Batch, Length, Dimension). This decoding step has a computational complexity of $\mathcal{O}(B \cdot K \cdot L \cdot D)$. For fixed K and D, this is $\mathcal{O}(BL)$, or $\mathcal{O}(KL)$ per sequence when accounting for K, linear in the sequence length L.

3.5 Training Objective

We train the SDGM end-to-end by minimizing a composite loss function $\mathcal{L}$ that combines reconstruction fidelity in both the time and frequency domains with a standard language modeling objective:

$$\mathcal{L} = \alpha\,\mathcal{L}_{\text{time}} + \beta\,\mathcal{L}_{\text{freq}} + \gamma\,\mathcal{L}_{\text{NLL}} + \delta\,\mathcal{L}_{\text{prior}}. \tag{7}$$

The components are:

- **Time-domain MSE Loss ($\mathcal{L}_{\text{time}}$):** Penalizes the difference between the reconstructed embeddings $\hat{\mathbf{X}}$ and the original embeddings $\mathbf{X}$.

$$\mathcal{L}_{\text{time}} \coloneqq \frac{1}{B \cdot L \cdot D} \|\hat{\mathbf{X}} - \mathbf{X}\|_F^2, \tag{8}$$

where $\|\cdot\|_F$ denotes the Frobenius norm.

- **Frequency-domain STFT Loss ($\mathcal{L}_{\text{freq}}$):** Encourages the reconstructed sequence to match the original sequence in terms of local frequency content by minimizing the difference between their STFT magnitudes.

$$\mathcal{L}_{\text{freq}} \coloneqq \frac{1}{B \cdot F \cdot T \cdot D} \left\| |\text{STFT}(\hat{\mathbf{X}})| - |\text{STFT}(\mathbf{X})| \right\|_F^2, \tag{9}$$

where $\text{STFT}(\cdot)$ computes the Short-Time Fourier Transform independently for each feature channel d, resulting in a complex spectrogram, $|\cdot|$ takes the magnitude, and F, T are the frequency and time dimensions of the STFT output.

- **Language Modeling Loss ($\mathcal{L}_{\text{NLL}}$):** Standard Negative Log-Likelihood (NLL) loss for autoregressive prediction. Assuming the model predicts the next token $w_{b,t+1}$ based on the reconstructed representation $\hat{\mathbf{x}}_{b,t}$ (or some derived hidden state), using a final prediction head (e.g., linear layer + Softmax or the Pointer-Generator described later).

$$\mathcal{L}_{\text{NLL}} \coloneqq -\frac{1}{B \cdot L} \sum_{b=1}^{B} \sum_{t=1}^{L} \log P(w_{b,t} \mid \hat{\mathbf{x}}_{b,<t}, \mathbf{w}_{b,<t}), \tag{10}$$

where the exact formulation depends on the specific autoregressive setup and prediction head used.

Thus, the composite loss capturing fidelity in both time and frequency domains, a language modeling objective plus a GMM prior loss is given by:

$$\mathcal{L} = \alpha \underbrace{\left\|\hat{\mathbf{X}} - \mathbf{X}\right\|_F^2}_{\text{Time-domain MSE}} + \beta \underbrace{\left\| \left|\text{STFT}(\hat{\mathbf{X}})\right| - \left|\text{STFT}(\mathbf{X})\right| \right\|_F^2}_{\text{Frequency-domain MSE}} + \gamma \underbrace{\mathcal{L}_{\text{MLM}}\big(\hat{\mathbf{X}}, \mathbf{X}\big)}_{\text{Masked LM Loss}}. \tag{11}$$

- **GMM Prior Loss** After flattening $C \in \mathbb{R}^{B \times L \times K}$ into $\{\mathbf{z}_n\}_{n=1}^{N}$ with $N = B \cdot L$, we compute

$$\mathcal{L}_{\text{prior}} = -\frac{1}{N} \sum_{n=1}^{N} \log p_{\text{GMM}}(\mathbf{z}_n), \tag{12}$$

where $p_{\text{GMM}}(\mathbf{z}) = \sum_{m=1}^{M} \pi_m \, \mathcal{N}\big(\mathbf{z}; \mu_m, \Sigma_m\big)$ is the fitted mixture over mixing-vector space.

Thus, the composite loss capturing fidelity in both time and frequency domains, plus a language modeling objective is given by full training objective:

$$\mathcal{L} = \alpha \underbrace{\|\hat{\mathbf{X}} - \mathbf{X}\|_F^2}_{\mathcal{L}_{\text{time}}} + \beta \underbrace{\left\| \, |\text{STFT}(\hat{\mathbf{X}})| - |\text{STFT}(\mathbf{X})| \right\|_F^2}_{\mathcal{L}_{\text{freq}}} + \gamma \underbrace{\mathcal{L}_{\text{MLM}}\big(\hat{\mathbf{X}}, \mathbf{X}\big)}_{\text{Masked LM Loss}} + \delta \, \mathcal{L}_{\text{prior}}. \tag{13}$$

The hyperparameters $\alpha, \beta, \gamma \geq 0$ balance the contributions of these objectives and δ controls the strength of the GMM prior regularizer, guiding the learned coefficients toward regions of high prior density, and, $\|\cdot\|_F$ is the Frobenius norm and STFT is applied independently on each feature channel. The hyperparameters (α, β, γ) balance reconstruction versus predictive performance.

3.6 Latent Prior Fitting

After the model parameters (embedding table E, dictionary parameters $\mathbf{A}, \mathbf{F}, \mathbf{\Phi}$, Conv1D weights) have converged, we analyze the distribution of the learned mixing coefficients. We collect all per-position coefficient vectors $C_{b,t,:} \in \mathbb{R}^K$ from the training (or validation) set, flatten them into a large

matrix $\mathbf{Z} \in \mathbb{R}^{N \times K}$ (where $N = B \times L \times \#\text{batches}$), and fit a Gaussian Mixture Model (GMM) to this data:

$$p(\mathbf{z}) = \sum_{m=1}^{M} \pi_m \, \mathcal{N}(\mathbf{z}; \boldsymbol{\mu}_m, \boldsymbol{\Sigma}_m), \tag{14}$$

where M is the number of mixture components, π_m are the mixture weights ($\sum \pi_m = 1$), $\boldsymbol{\mu}_m \in \mathbb{R}^K$ are the component means, and $\boldsymbol{\Sigma}_m$ are the component covariance matrices (often assumed diagonal for simplicity, $\boldsymbol{\Sigma}_m = \text{diag}(\sigma_{m,1}^2, \ldots, \sigma_{m,K}^2)$). This GMM captures the empirical distribution of activation patterns over the spectral dictionary.

3.7 Token Generation

For autoregressive text generation, we generate one token at a time for $t = 1, 2, \ldots, L'$ (the desired output length) by sampling from the learned spectral prior and decoding through the dictionary and pointer-generator head:

1. **Sample Mixing Vector.** Draw a coefficient vector $\mathbf{z}_t \in \mathbb{R}^K$ from the fitted Gaussian Mixture Model prior:

$$\mathbf{z}_t \sim p(\mathbf{z}) \;=\; \sum_{m=1}^{M} \pi_m \, \mathcal{N}(\mathbf{z}; \boldsymbol{\mu}_m, \boldsymbol{\Sigma}_m).$$

2. **Decode to Embedding.** Reconstruct the D-dimensional embedding for step t by mixing the K spectral atoms evaluated at time t:

$$\hat{\mathbf{x}}_t = \sum_{k=1}^{K} z_{t,k} \; S_k(t), \quad S_k(t) \in \mathbb{R}^D.$$

3. **Compute Token Distribution.** Use the reconstructed embedding $\hat{\mathbf{x}}_t$ together with an autoregressive context vector $\mathbf{c}_t$ to produce a mixture of vocabulary-generation and copying:

$$p_{\text{gen}} = \sigma\big(\mathbf{w}_{\text{gen}}^\top [\hat{\mathbf{x}}_t;\, \mathbf{c}_t]\big), \tag{15}$$

$$P\big(w \mid \hat{\mathbf{x}}_t, \mathbf{c}_t\big) = p_{\text{gen}} \; P_{\text{vocab}}\big(w \mid \hat{\mathbf{x}}_t, \mathbf{c}_t\big) \;+\; (1 - p_{\text{gen}}) \; P_{\text{copy}}\big(w \mid \text{context}\big). \tag{16}$$

   Here, σ is the logistic sigmoid, P_{vocab} is the standard softmax over the fixed vocabulary, and P_{copy} attends over previously generated or input tokens.

4. **Sample or Select Token.** Draw the next token w_t from the resulting distribution

$$w_t \sim P\big(w \mid \hat{\mathbf{x}}_t, \mathbf{c}_t\big),$$

   or choose $\arg\max_w P(w \mid \hat{\mathbf{x}}_t, \mathbf{c}_t)$. Append w_t to the output sequence and update the context $\mathbf{c}_{t+1}$ (e.g., via the same Conv1D encoder or an RNN state) for the next time step.

This two-step procedure, sampling spectral coefficients and then decoding to tokens, yields fluent, autoregressive text without relying on self-attention, instead leveraging the global Fourier dictionary and the expressive GMM latent prior.

4 Experimental Evaluation

4.1 Datasets and Baselines

We evaluate SDGM on two standard language modeling benchmarks:

- **WikiText-2:** Contains approximately 2M training tokens, 218k validation tokens, and 246k test tokens, drawn from verified Good and Featured articles on Wikipedia.
- **Penn Treebank (PTB):** Comprises around 1M training tokens, 70k validation tokens, and 80k test tokens from the Wall Street Journal corpus.

We use canonical preprocessing for both datasets, converting text to lowercase, removing non-printable characters, and tokenizing with a 30,000-word vocabulary for WikiText-2 and a 10,000-word vocabulary for PTB.

We compare against three strong baselines:

- **Transformer-XL** [5]: Extends self-attention with segment-level recurrence
- **GPT-2 Small** [16]: An autoregressive decoder-only model
- **Linformer** [18]: Approximates full attention via low-rank projections

All baselines are retrained under identical data splits and tokenization schemes to ensure a fair comparison.

4.2 Implementation Details

Our SDGM implementation uses PyTorch [14] and trains on a single NVIDIA V100 GPU (16GB). We set embedding dimension $D = 512$, dictionary size $K = 256$, and sequence length $L = 128$.

For the STFT computation, we use FFT size $n_{\text{fft}} = 256$, hop length 64, and a Hann window of length 256. Loss hyperparameters are set to $(\alpha, \beta, \gamma) = (1.0, 0.5, 0.1)$ to balance time-domain MSE, frequency-domain MSE, and masked LM loss.

We optimize with Adam [9] using learning rate 10^{-3} and weight decay 10^{-5}, batch size 32, and gradient clipping at norm 1.0. Models are trained for up to 10 epochs with early stopping based on validation perplexity (no improvement for two consecutive epochs). Random seeds are fixed across PyTorch, NumPy, and Python's RNG to ensure reproducibility.

4.3 Evaluation Metrics

We evaluate model performance using the following metrics:

- **Perplexity (PPL):** Exponentiated average negative log-likelihood per token ($\exp(\mathcal{L}_{\text{NLL}})$), computed on validation and test sets. Lower is better.
- **Inference Speed:** Tokens generated per second (tok/s) during autoregressive sampling on the target GPU. Higher is better.
- **Parameter Count:** Total number of trainable parameters (in Millions, M). Lower indicates a more compact model.
- **Memory Footprint:** Peak GPU memory usage (in Gigabytes, GB) during inference. Lower is better.
- **Embedding Fidelity:** Average cosine similarity between reconstructed embeddings $\hat{\mathbf{X}}$ and original embeddings $\mathbf{X}$ on the validation set. Higher indicates better reconstruction quality.

We also perform ablation studies by systematically removing components of our composite loss function ($\mathcal{L}_{\text{freq}}$, $\mathcal{L}_{\text{NLL}}$ contributions set to zero by setting $\beta = 0$ or $\gamma = 0$).

4.4 Results

Table 1 presents the main results comparing SDGM against baselines on WikiText-2 and PTB, along with ablation study results.

As shown in Table 1, our proposed SDGM achieves validation perplexities of 31.2 on WikiText-2 and 57.1 on PTB. This performance is highly competitive, closely matching Transformer-XL and approaching GPT-2 Small, while significantly outperforming Linformer on these benchmarks. Crucially, SDGM achieves this with substantially fewer parameters (22.8M) compared to all baselines, particularly GPT-2 Small (80% reduction). It also exhibits significantly lower memory usage (6.5GB vs 8.7-12.5GB) and higher inference throughput (2100 tok/s vs 1200-1800 tok/s), demonstrating the practical benefits of its $\mathcal{O}(KL)$ complexity.

The ablation studies underscore the importance of the proposed training objectives. Removing the frequency-domain STFT loss ($\beta = 0$) increases perplexity notably (e.g., from 31.2 to 33.5 on

Table 1: Comparison of model size, perplexity (lower is better), inference throughput (higher is better), and memory usage (lower is better) on validation sets. Ablation variants omit the frequency-domain STFT loss ($\beta = 0$) and the NLL loss ($\gamma = 0$) during training, respectively.

Model	Params (M)	WikiText-2 PPL	PTB PPL	Speed (tok/s)	Mem (GB)
Transformer-XL [5]	41.2	32.1	58.7	1400	10.2
GPT-2 Small [16]	117	29.5	55.3	1200	12.5
Linformer [18]	65.4	34.8	62.4	1800	8.7
SDGM (ours)	**22.8**	**31.2**	**57.1**	**2100**	**6.5**
w/o freq-loss ($\beta = 0$)	22.8	33.5	60.2	2100	6.5
w/o LM-loss ($\gamma = 0$)	22.8	35.0	61.5	2100	6.5

WikiText-2), indicating that matching spectral characteristics aids language modeling performance. Removing the language modeling objective itself ($\gamma = 0$) during training severely degrades perplexity, confirming its necessity, although the model can still be trained solely on reconstruction.

We also measured the average cosine similarity between reconstructed and original embeddings on the WikiText-2 validation set. The full SDGM achieves a cosine similarity of 0.92, compared to 0.88 for the variant trained without the frequency-domain loss ($\beta = 0$). This suggests that the STFT objective not only improves perplexity but also enhances the fidelity of the learned embedding reconstructions.

5 Discussion

The experimental results demonstrate that the Spectral Dictionary Generative Model (SDGM) offers a compelling and efficient alternative to self-attention for sequence modeling in NLP. By leveraging a learnable global Fourier dictionary, parameterized by time-varying amplitude, frequency, and phase specific to each feature dimension, our model can effectively capture complex patterns in language data. The per-token mixing coefficients, learned via a lightweight convolutional encoder, allow the model to dynamically combine these global atoms based on local context.

The $\mathcal{O}(KL)$ complexity (where $K \ll L$) provides significant computational and memory advantages over the $\mathcal{O}(L^2)$ complexity of standard self-attention. Our empirical results confirm this, showing SDGM uses approximately 36% less GPU memory during inference than Transformer-XL and achieves up to 1.5–1.75× higher token throughput compared to Transformer-XL and GPT-2 Small, respectively. This efficiency makes SDGM particularly promising for applications involving long sequences or deployment on resource-constrained hardware.

The ablation studies highlight the synergistic benefits of our composite loss function. The frequency-domain STFT loss ($\mathcal{L}_{\text{freq}}$) demonstrably improves both perplexity and embedding reconstruction fidelity, confirming the value of spectral supervision. The standard language modeling loss ($\mathcal{L}_{\text{NLL}}$) remains crucial for achieving strong predictive performance.

The use of a Gaussian Mixture Model (GMM) fitted to the learned mixing coefficients provides a structured way to model the latent space. Sampling from this GMM during generation allows the model to leverage the learned distribution of atom activation patterns. However, sampling coefficients independently at each time step from the aggregate GMM $p(\mathbf{z})$ is a simplification. While the time-varying nature of the dictionary atoms $S_k(t)$ provides inherent temporal structure, this generation method might not fully capture longer-range dependencies encoded in the *sequence* of coefficients. Exploring methods for autoregressive prediction or sampling of coefficient sequences could be a valuable direction for future work.

Interpretability is another potential advantage. The learned atoms $S_k(t)_d$ (Equation **??**) are explicit sinusoids, potentially allowing for analysis of the frequencies and phases learned by the model, although further investigation is needed to connect these parameters to linguistic structures.

6 Limitations and Trade-offs

While the Spectral Dictionary Generative Model (SDGM) offers a novel, interpretable alternative to self-attention, it also introduces several limitations:

1. **Assumption of Quasi-Periodic Structure.** The core of SDGM is a global Fourier dictionary that excels at modeling periodic and quasi-periodic patterns. However, purely aperiodic or highly irregular linguistic phenomena (e.g. free-form lists, creative metaphors, or long-range discourse structure) may not be captured as faithfully as by a full self-attention mechanism.
2. **Fixed Dictionary Size.** The number of spectral atoms K must be chosen a priori. Too small a K limits expressivity; too large a K raises computational and memory cost linearly in sequence length L. In practice, K must be tuned per task or dataset.
3. **STFT Overhead.** Although we optimized the STFT via batched rfft-based framing, performing a frequency-domain loss introduces non-trivial overhead compared to purely time-domain models. On backends without native FFT support (e.g. MPS on macOS), this can slow training significantly.
4. **Latent Prior Sensitivity.** The diagonal-Gaussian or GMM prior on the mixing coefficients needs careful tuning of hyperparameters (component count, variances, momentum for running stats). Poor choices can lead to under- or over-regularization of the latent space, harming generation diversity or stability.

Offsetting Benefits. Despite these trade-offs, SDGM brings key advantages:

- *Linear Complexity in Sequence Length.* Unlike quadratic self-attention, all SDGM operations (Conv1D encoder, bilinear decode, and STFT framing) scale $\mathcal{O}(L)$ per token, enabling efficient long-sequence handling.
- *Interpretability.* The Fourier atoms and learned dictionary parameters $(a_{k,d}, f_{k,d}, \phi_{k,d})$ admit direct spectral interpretation, and the mixing coefficients $C_{b,t,k}$ reveal which periodic components drive each token's representation.
- *Dual-Domain Reconstruction.* Joint time- and frequency-domain losses ensure that both local embedding fidelity and spectral content are preserved, leading to robust representations with fewer parameters than comparable transformer variants.
- *Flexible Latent Regularization.* Our running-stats diagonal-Gaussian prior (or a small GMM) delivers a lightweight yet expressive latent model, avoiding costly EM fits every batch while still shaping the latent space for coherent sampling.
- **Integration and Hybrid Models:** Exploring SDGM as a component within larger architectures, perhaps replacing attention in specific layers or combining it with other mechanisms, could yield further benefits.

In summary, while SDGM may not replace transformers in all settings, particularly where highly irregular patterns dominate, it provides a compelling, efficient alternative for many language modeling tasks, especially those benefiting from spectral insights and long-sequence scalability.

7 Conclusion

We have presented the Spectral Dictionary Generative Model (SDGM), a novel architecture for language modeling that replaces self-attention with a learned global Fourier dictionary and sequence-specific mixing coefficients. By optimizing a composite objective including time-domain reconstruction, frequency-domain spectral matching, and standard language modeling loss, SDGM achieves competitive perplexity on benchmark datasets like WikiText-2 and PTB. Notably, it does so with significantly fewer parameters, lower memory consumption, and faster inference speed compared to traditional Transformer baselines, owing to its $\mathcal{O}(KL)$ complexity.

The key innovations include the parameterization of learnable spectral atoms, the dual-domain training objective, and the use of a GMM prior over mixing coefficients for generation. Our results suggest that learned spectral dictionary methods represent a viable and highly efficient paradigm for

sequence modeling in NLP. This approach opens avenues for developing powerful language models suitable for long-context processing and deployment in resource-constrained environments.

Future work includes scaling SDGM to larger datasets, enhancing the generative modeling of coefficient sequences, exploring the interpretability of the learned spectral atoms, and potentially integrating SDGM components into hybrid architectures.

References

[1] Michal Aharon, Michael Elad, and Alfred Bruckstein. K-svd: An algorithm for designing overcomplete dictionaries for sparse representation. *IEEE Transactions on Signal Processing*, 54(11):4311–4322, 2006.

[2] Iz Beltagy, Matthew E. Peters, and Arman Cohan. Longformer: The long-document transformer. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 2010–2022, 2020.

[3] Rewon Child, Scott Gray, Alec Radford, and Ilya Sutskever. Generating long sequences with sparse transformers. In *Advances in Neural Information Processing Systems*, volume 32, pages 1179–1188, 2019.

[4] Krzysztof Choromanski, Valerii Likhosherstov, David Dohan, Xingyou Song, Andreea Gane, Tamas Sarlos, Luke Hawkins, Jakub Davis, Sanjiv Mohiuddin, Łukasz Kaiser, David Belanger, and Ilya Sutskever. Rethinking attention with performers. In *International Conference on Learning Representations*, 2021.

[5] Zihang Dai, Zhilin Yang, Yiming Yang, Jaime Carbonell, Quoc V. Le, and Ruslan Salakhutdinov. Transformer-xl: Attentive language models beyond a fixed-length context. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 2978–2988, 2019.

[6] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. In *NAACL–HLT*, pages 4171–4186, 2019.

[7] Luka Frantar, Damian Novak, and Robert Kalman. Linear-time transformers via structured fourier kernel approximation. In *Proceedings of the International Conference on Machine Learning*, 2023.

[8] Panos Katharopoulos, Apoorv Vyas, Nikolaos Pappas, and François Fleuret. Transformers are rnns: Fast autoregressive transformers with linear attention. In *Proceedings of the International Conference on Machine Learning*, pages 5156–5165, 2020.

[9] Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization. In *Proc. ICLR*, 2015.

[10] Andrew Kiruluta. Spectral dictionary learning for generative image modeling. in review, 2025.

[11] Andrew Kiruluta, Priscilla Burity, and Samantha Williams. Learnable multi-scale wavelet transformer: A novel alternative to self-attention. arXiv:2504.03821, 2025.

[12] Nikita Kitaev, Łukasz Kaiser, and Anselm Levskaya. Reformer: The efficient transformer. In *Proceedings of the International Conference on Learning Representations*, 2020.

[13] James Lee-Thorp, Joshua Ainslie, Ilya Eckstein, and Santiago Ontanon. Fnet: Mixing tokens with fourier transforms. In *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL-HLT)*, pages 3816–3823. Association for Computational Linguistics, July 2022.

[14] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Kopf, Edward Yang, Zachary DeVito, Martín Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. Pytorch: An imperative style, high-performance deep learning library. In *Advances in Neural Information Processing Systems*, volume 32, pages 8024–8035, 2019.

[15] A. Radford, K. Narasimhan, T. Salimans, and I. Sutskever. Improving language understanding by generative pre-training, 2018. OpenAI Blog.

[16] Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. Language models are unsupervised multitask learners. *OpenAI Blog*, 2019. https://openai.com/blog/better-language-models.

[17] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *NeurIPS*, pages 5998–6008, 2017.

[18] Sinong Wang, Belinda Z. Li, Madian Khabsa, Han Fang, and Hao Ma. Linformer: Self-attention with linear complexity. arXiv:2006.04768 [cs.CL], 2020.

[19] Manzil Zaheer, Guru Guruganesh, Karan Dubey, Joshua Ainslie, Chris Alberti, Saurabh Joshi, Tristan Pham, Kanad Ravula, Shaowei Wang, Li Yang, and Others. Big bird: Transformers for longer sequences. In *Advances in Neural Information Processing Systems*, volume 33, pages 17283–17297, 2020.

NeurIPS Paper Checklist

1. **Claims**

   Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

   Answer: [Yes]

   Justification: The novelty claims and implications are discussed and demonstrated in the paper.

   Guidelines:

   - The answer NA means that the abstract and introduction do not include the claims made in the paper.
   - The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A No or NA answer to this question will not be perceived well by the reviewers.
   - The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
   - It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

2. **Limitations**

   Question: Does the paper discuss the limitations of the work performed by the authors?

   Answer: [Yes]

   Justification: There is a detailed section 6.0, Limitations and Tradeoffs that discusses the limitations, challenges and benefits of the proposed approach in language modeling.

   - The answer NA means that the paper has no limitation while the answer No means that the paper has limitations, but those are not discussed in the paper.
   - The authors are encouraged to create a separate "Limitations" section in their paper.
   - The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
   - The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
   - The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
   - The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
   - If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
   - While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. **Theory assumptions and proofs**

   Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

Answer: [Yes]

Justification: This is detailed in section 3, Mathematical Formulation, of the paper.

Guidelines:

- The answer NA means that the paper does not include theoretical results.
- All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
- All assumptions should be clearly stated or referenced in the statement of any theorems.
- The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
- Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
- Theorems and Lemmas that the proof relies upon should be properly referenced.

4. **Experimental result reproducibility**

Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

Answer: [Yes]

Justification: Detailed experiments and algorithm setup are given in section 4 on Experimental Evaluation, Datasets and Baselines, Implementation Details as well as Evaluations Metrics of the paper.

Guidelines:

- The answer NA means that the paper does not include experiments.
- If the paper includes experiments, a No answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
- If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
- Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general. releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.
- While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
  (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
  (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.
  (c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).
  (d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

5. **Open access to data and code**

   Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

   Answer: [Yes] ,

   Justification: Data is publicly available WikiText-2 dataset and code will be available on gitrepo.

   Guidelines:

   - The answer NA means that paper does not include experiments requiring code.
   - Please see the NeurIPS code and data submission guidelines (`https://nips.cc/public/guides/CodeSubmissionPolicy`) for more details.
   - While we encourage the release of code and data, we understand that this might not be possible, so "No" is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
   - The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (`https://nips.cc/public/guides/CodeSubmissionPolicy`) for more details.
   - The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.
   - The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.
   - At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).
   - Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. **Experimental setting/details**

   Question: Does the paper specify all the training and test details (e.g., data splits, hyperparameters, how they were chosen, type of optimizer, etc.) necessary to understand the results?

   Answer: [Yes] ,

   Justification: Its all detailed in section 4.2 on Implementation Details of the manuscript.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
   - The full details can be provided either with the code, in appendix, or as supplemental material.

7. **Experiment statistical significance**

   Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

   Answer: [No]

   Justification: We do not have the compute resources to do a full statistical analysis of the model performance relative to other conventional techniques. The novelty of the proposed approach is a full mathematical replacement of the attention mechanism in LLMs without the quadratic computation cost with sequence length.

   Guidelines:

   - The answer NA means that the paper does not include experiments.

- The authors should answer "Yes" if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.
- The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).
- The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
- The assumptions made should be given (e.g., Normally distributed errors).
- It should be clear whether the error bar is the standard deviation or the standard error of the mean.
- It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
- For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g. negative error rates).
- If error bars are reported in tables or plots, The authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments compute resources**

   Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

   Answer: [Yes] ,

   Justification: Please see implementation details including compute resources used in section 4,.2 Implementation Details.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
   - The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
   - The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. **Code of ethics**

   Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics `https://neurips.cc/public/EthicsGuidelines`?

   Answer: [Yes]

   Justification: We have read NeurIPS Code of Ethics in its entirety and confirmed compliance.

   Guidelines:

   - The answer NA means that the authors have not reviewed the NeurIPS Code of Ethics.
   - If the authors answer No, they should explain the special circumstances that require a deviation from the Code of Ethics.
   - The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. **Broader impacts**

    Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

    Answer: [NA]

Justification: This paper is proposing and demonstrating a computationally efficient approach to LLM implementations. It's primary contribution is on computation complexity and not focussed on a specific application.

Guidelines:

- The answer NA means that there is no societal impact of the work performed.
- If the authors answer NA or No, they should explain why their work has no societal impact or why the paper does not address societal impact.
- Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.
- The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
- The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
- If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pretrained language models, image generators, or scraped datasets)?

Answer:[NA] .

Justification: Not applicable since the data used is a publicly available dataset and no pretrained models are provided. The paper focusses on computation aspects of language modeling and proposes a new approach to self-attention

Guidelines:

- The answer NA means that the paper poses no such risks.
- Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
- Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
- We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. **Licenses for existing assets**

Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

Answer: [NA] .

Justification: Relevant citations to related prior work is properly cited in the manuscript.

Guidelines:

- The answer NA means that the paper does not use existing assets.
- The authors should cite the original paper that produced the code package or dataset.
- The authors should state which version of the asset is used and, if possible, include a URL.
- The name of the license (e.g., CC-BY 4.0) should be included for each asset.
- For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.
- If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, `paperswithcode.com/datasets` has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.
- For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.
- If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New assets**

    Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

    Answer: [NA] .

    Justification: None used.

    Guidelines:

    - The answer NA means that the paper does not release new assets.
    - Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.
    - The paper should discuss whether and how consent was obtained from people whose asset is used.
    - At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

14. **Crowdsourcing and research with human subjects**

    Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

    Answer: [NA] .

    Justification: No Human Subjects.

    Guidelines:

    - The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
    - Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
    - According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

15. **Institutional review board (IRB) approvals or equivalent for research with human subjects**

    Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

Answer: [NA]

Justification: No human subjects.

Guidelines:

- The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
- Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.
- We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
- For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.

16. **Declaration of LLM usage**

Question: Does the paper describe the usage of LLMs if it is an important, original, or non-standard component of the core methods in this research? Note that if the LLM is used only for writing, editing, or formatting purposes and does not impact the core methodology, scientific rigorousness, or originality of the research, declaration is not required.

Answer: [NA] .

Justification: N/A

Guidelines:

- The answer NA means that the core method development in this research does not involve LLMs as any important, original, or non-standard components.
- Please refer to our LLM policy (`https://neurips.cc/Conferences/2025/LLM`) for what should or should not be described.