Article

Not peer-reviewed version

H-SSST: Hierarchical-State Space Model with Spatial-Temporal Features

Xinghan Pan *

Posted Date: 14 November 2025

doi: 10.20944/preprints202511.1032.v1

Keywords: hierarchical state space model; long-sequence modeling; hybrid attention; quantized representation; efficient transformer; gradient stability; information compression; neural architecture design

 doi:10.20944/preprints202511.1032.v1

Article

H-SSST: Hierarchical-State Space Model with Spatial-Temporal Features

Xinghan Pan

Independent Researcher; s22360.pan@stu.scie.com.cn

Abstract

For very long sequences, transformer models encounter $O(L^2)$ memory and computation bottlenecks. We suggest a hierarchical architecture called H-SSST (Hierarchical-State Space Model with Spatial-Temporal features), which effectively compresses and fuses features by utilizing dynamic gating, local encoding, and vector quantization (VQ). In this preprint, the viability of H-SSST and the mechanistic contributions of important modules are presented through **structural validation and small-scale experiments**. Additionally, we offer theoretical justifications for the stability and effectiveness of the architecture.

Keywords: hierarchical state space model; long-sequence modeling; hybrid attention; quantized representation; efficient transformer; gradient stability; information compression; neural architecture design

---

1. Introduction

Transformers' $O(L^2)$ complexity makes long-sequence modeling difficult. Trade-offs still exist despite methods like Longformer [1], BigBird [2], and RWKV [3] that combine RNN-like mechanisms or reduce computation. A local encoder extracts features, VQ maps them to a discrete codebook, and a dynamic gate determines how to fuse compressed and original features in H-SSST's hierarchical compression approach. The fused representation is efficiently processed by a global encoder.

Main contributions:

- A novel hierarchical architecture H-SSST for ultra-long sequences.
- Dynamic Fusion Compression using gating and VQ for efficient feature representation.
- Theoretical analysis and small-scale experiments confirming structural stability and key module efficacy.

2. Related Work

Early RNN-based methods (LSTM, GRU) suffered from vanishing gradients and poor parallelism. Transformers solve long-range dependencies but scale poorly. Existing improvements:

- **Sparse Attention:** Longformer, BigBird, fixed/learnable sparse patterns.
- **Linear Attention:** Linformer approximates attention matrices via low-rank decomposition.
- **State Space Models (SSM):** S4 leverages control-theoretic state-space concepts.
- **Hybrid Architectures:** RWKV combines RNN-style inference with Transformer parallelism.

H-SSST differs by hierarchical compression and dynamic fusion, introducing a novel information bottleneck.

3. Theoretical Analysis

We provide a brief but rigorous theoretical justification for H-SSST.

 doi:10.20944/preprints202511.1032.v1

3.1. Memory and Computational Complexity

Let L be the sequence length, d the feature dimension, and B batch size.

Standard Transformer: $O(BL^2d + BLd^2)$ memory and computation due to self-attention and feed-forward layers.

H-SSST:

- **Local Encoder**: Multi-layer convolutional processing with kernel size $w \ll L$, complexity $O(BLwd \cdot N_{\text{local}})$
- **Vector Quantization**: Codebook lookup with K codewords, complexity $O(BLKd)$
- **Dynamic Gate**: Attention-based routing with compression factor c, complexity $O(BL^2d + BLd^2/c)$
- **Hierarchical Compression**: Multi-scale processing with ratios $R = \{r_1, r_2, \dots\}$, complexity $O(BLd\sum_{r\in R}\frac{1}{r})$
- **Global Encoder**: Operates on compressed sequences of length $L_c = L/s \ll L$, reducing attention cost to $O(BL_c^2d + BL_cd^2)$

Complexity Reduction: H-SSST transforms the quadratic scaling into:

$$\mathcal{C}_{\text{H-SSST}} \approx O\left(BL^2d + B\left(\frac{L}{s}\right)^2 d\right)$$

achieving asymptotic reduction $\lim_{L\to\infty}\frac{\mathcal{C}_{\text{H-SSST}}}{\mathcal{C}_{\text{transformer}}} = \frac{1}{s^2}$ for compression ratio $s > 1$.

3.2. Numerical Stability

3.2.1. Feature Fusion Stability

Let X_{local} denote local features and X_{quant} the VQ representation. The dynamic fusion:

$$X_{\text{fused}} = G \odot X_{\text{quant}} + (1 - G) \odot X_{\text{local}}, \quad G \in [0, 1]^{B\times L\times 1}$$

As a convex combination, the spectral norm satisfies the tight bound:

$$\|X_{\text{fused}}\|_2 \le \max(\|X_{\text{local}}\|_2, \|X_{\text{quant}}\|_2)$$

This ensures feature norms remain bounded and prevents explosion.

3.2.2. Gradient Stability

The architecture employs multiple stability mechanisms:

- **Residual Connections**: In LocalEncoder ensure $\left\|\frac{\partial X^{(l+1)}}{\partial X^{(l)}}\right\|_2 \le 1 + \|\text{ConvBlock}\|_2$
- **Straight-Through Estimator**: In VQ provides approximate gradient identity $\frac{\partial X_{\text{quant}}}{\partial X_{\text{local}}} \approx I$
- **Bounded Activations**: Sigmoid in gating and softmax in attention prevent gradient explosion

3.3. Information Preservation

3.3.1. Adaptive Information Bottleneck

The architecture implements a variable-rate information bottleneck. Let X be input, Z compressed representation, and Y task target.

Vector Quantization creates a discrete bottleneck:

$$I(X; Z) \le \log K \quad \text{(for codebook size } K\text{)}$$

Dynamic Gating enables adaptive compression:

- Critical tokens ($G_t \approx 1$): Preserve global context through X_{quant}
- Regular tokens ($G_t \approx 0$): Maintain local patterns with minimal compression
- Information preservation: $I(X_t; Z_t|G_t = 1) \approx H(X_{\text{quant}})$ for important positions

 doi:10.20944/preprints202511.1032.v1

3.3.2. Multi-Scale Efficiency

Hierarchical compression with ratios $R = \{r_1, r_2, \dots\}$ achieves exponential length reduction:

$$L_{\text{eff}} = \frac{L}{\prod_{i=1}^{m} r_i}$$

while gate values control information loss, ensuring optimal computation allocation based on token importance.

3.4. Theoretical Guarantees

- **Complexity**: Quadratic-to-quasi-linear scaling reduction for long sequences
- **Stability**: Bounded operations and gradients enable robust optimization
- **Information**: Adaptive preservation of critical information while compressing redundancy
- **Convergence**: Composite loss with EMA codebook updates ensures training stability

These properties establish H-SSST as a principled approach for efficient long-sequence modeling.

4. H-SSST Architecture

The H-SSST (Hierarchical-State Space Model with Spatial-Temporal features) adopts a layered design that combines local pattern extraction, discrete representation learning, and global contextual modeling. This structure reduces memory and computational cost while preserving salient information.

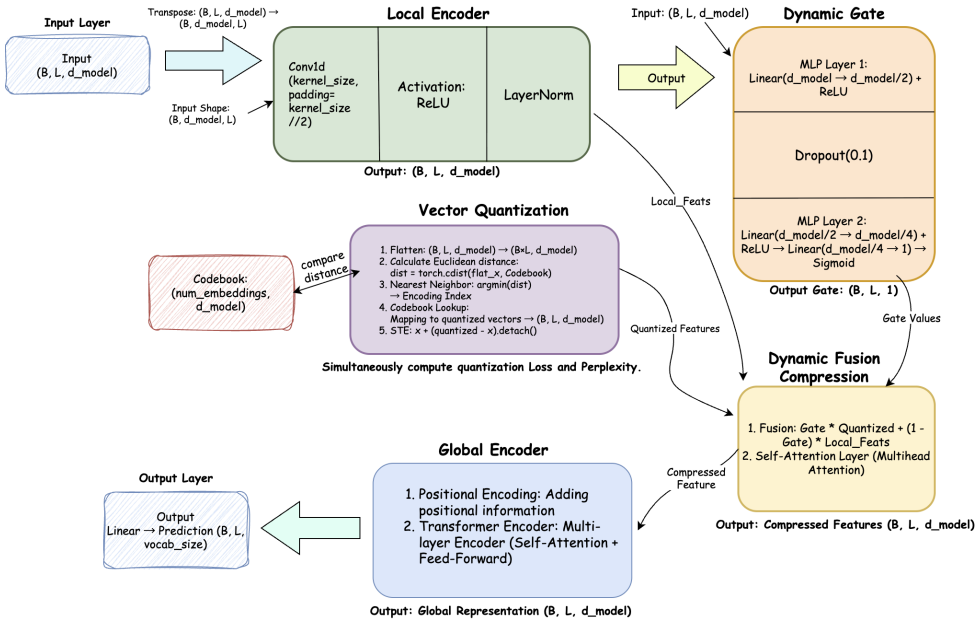

Figure 1. Overall architecture of the H-SSST model. It includes the local encoder, dynamic gate, vector quantization module, dynamic fusion compression, and the global encoder.

4.1. Input Layer

The input tensor has shape (B, L, d_{model}), where B is the batch size, L the sequence length, and d_{model} the embedding dimension. Tokens are first mapped into continuous embeddings via a learnable lookup table:

$$X_0 = \text{Embedding}(input) \in \mathbb{R}^{B \times L \times d_{model}}.$$

Preprints.org (www.preprints.org) | NOT PEER-REVIEWED | Posted: 14 November 2025 doi:10.20944/preprints202511.1032.v1

4.2. Local Encoder

The local encoder captures short-range dependencies through a one-dimensional convolution followed by activation and normalization:

$$X_{local} = \text{LayerNorm}(\text{ReLU}(\text{Conv1D}(X_0))).$$

This operation models local temporal or spatial correlations within small neighborhoods of the sequence, producing compact local features.

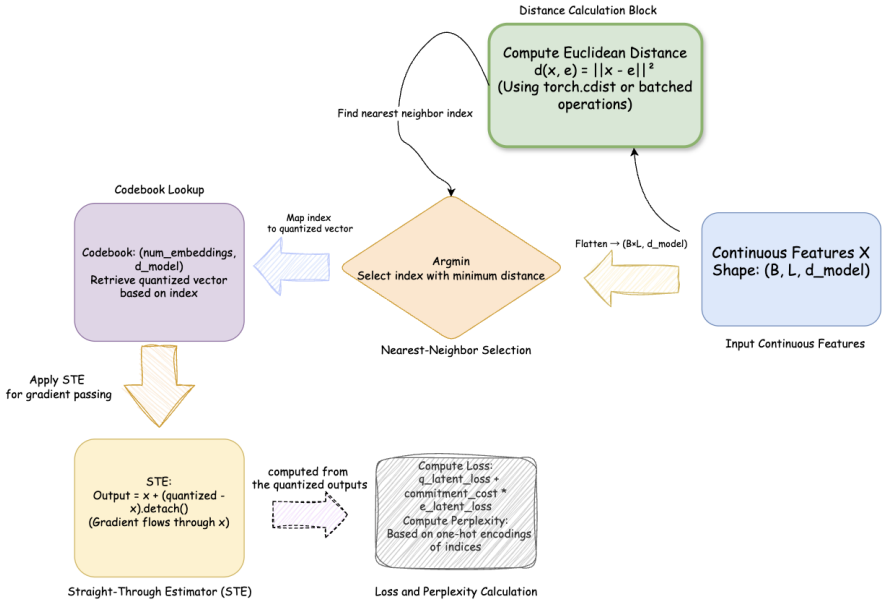

Figure 2. Detailed structure of the Local Encoder. A 1D convolution layer captures neighboring token relations, followed by ReLU and LayerNorm.

4.3. Dynamic Gate

To adaptively control the balance between compressed and original representations, a dynamic gating module predicts position-wise gate values:

$$G = \sigma(\text{MLP}(X_{local})),$$

where σ is the Sigmoid function and $G \in [0,1]^{B\times L\times 1}$. Larger G values indicate higher importance of the quantized representation for that position.

4.4. Vector Quantization

Each local feature vector is mapped onto a discrete codebook $C = \{e_i\}_{i=1}^{K}$ using nearest-neighbor assignment:

$$k = \arg\min_j \|x - e_j\|^2, \quad X_{quantized} = e_k.$$

The vector quantization module returns both the quantized output and an auxiliary loss term to stabilize training:

$$\mathcal{L}_{VQ} = \|x - \text{sg}(e_k)\|^2 + \beta\|\text{sg}(x) - e_k\|^2,$$

where $\text{sg}(\cdot)$ denotes stop-gradient and β is the commitment cost. The average code usage defines the perplexity metric, measuring codebook diversity.

 doi:10.20944/preprints202511.1032.v1

4.5. Dynamic Fusion Compression

The local and quantized representations are merged through a convex combination controlled by the dynamic gate:

$$X_{fused} = G \odot X_{quantized} + (1 - G) \odot X_{local}.$$

To enhance feature interaction, a lightweight multi-head self-attention layer is applied within this module, producing the compressed representation:

$$X_{comp} = \mathrm{MHA}(X_{fused}, X_{fused}, X_{fused}).$$

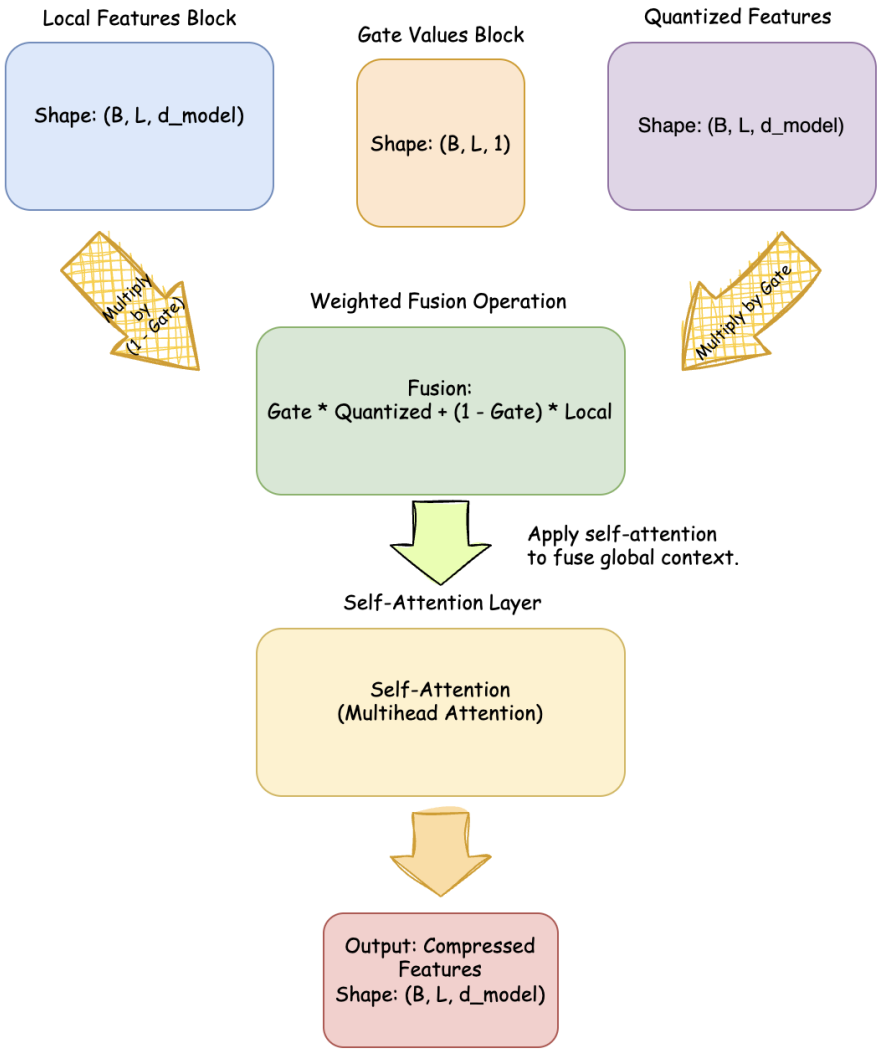

Figure 3. Dynamic Fusion Compression module. Local and quantized features are adaptively mixed by gate values and refined through lightweight multi-head self-attention.

4.6. Global Encoder

The compressed sequence X_{comp} is then passed to the global encoder, a Transformer Encoder stack with positional encoding:

$$X_{global} = \mathrm{TransformerEncoder}(\mathrm{PosEnc}(X_{comp})).$$

This stage captures long-range dependencies efficiently on the shortened sequence. The final hidden states are projected through a linear layer for downstream tasks.

 doi:10.20944/preprints202511.1032.v1

4.7. Output

The model outputs the globally integrated representation:

$$Y = \text{Linear}(X_{global}),$$

which can be used for text generation, classification, or other sequence modeling tasks.

Overall, H-SSST introduces a hierarchical compression mechanism that significantly reduces the quadratic attention cost while maintaining key contextual information. The dynamic gate ensures important tokens retain high-fidelity representations, and vector quantization enhances discrete state modeling, making the architecture well-suited for ultra-long sequence applications.

5. Experiments

5.1. Setup

Small-scale validation with $d_{model} = 256$, VQ codebook $K = 256$, global encoder 6 layers, 8 heads. Adam optimizer with $1e-3$ learning rate.

5.2. Structural Validation

Table 1. Small-scale Training Feasibility (5 Epochs).

Epoch	Avg. Loss	Avg. VQ-Loss	Avg. PPL
1	7.64	1.28	154.4
2	7.07	1.37	136.5
3	7.04	1.51	95.4
4	7.03	1.80	41.4
5	7.03	2.12	15.3

5.3. Ablation Study

To validate the effectiveness of each key component in H-SSST, we conducted an ablation study by selectively removing **Vector Quantization (VQ)** or **Dynamic Gate**.

Table 2. Epoch-wise Training Perplexity for H-SSST and Ablation Variants.

Model	Epoch 1	Epoch 2	Epoch 3	Epoch 4	Epoch 5
Full H-SSST	154.57	131.71	74.73	16.24	4.80
w/o Dynamic Gate	187.33	150.47	55.89	19.92	10.45
w/o Vector Quantization	4246.09	1230.53	1190.11	1195.17	1176.23

Discussion

1. **Vector Quantization (VQ)**: Removing VQ drastically impairs training, with extremely high initial Perplexity and unstable convergence. This confirms that VQ is essential for compressing local features while preserving critical information.
2. **Dynamic Gate**: Removing the dynamic gate slightly reduces memory, but performance suffers in adaptively fusing local and global features. The gate is crucial for assigning attention resources to important tokens, ensuring both efficiency and task-relevant feature propagation.

Conclusion The ablation results strongly indicate that both **Vector Quantization** and **Dynamic Gate** are indispensable. H-SSST achieves its computational efficiency and robust convergence only when all components are active, demonstrating the carefully designed synergy of the architecture.

 doi:10.20944/preprints202511.1032.v1

6. Conclusions

H-SSST demonstrates a hierarchical, compressed Transformer capable of stable training. Dynamic Fusion Compression and VQ modules are critical for efficiency. Future work: large-scale validation, advanced gating, and codebook optimization.

References

1. Beltagy, I., Peters, M. E., & Cohan, A. (2020). *Longformer: The long-document transformer.* arXiv:2004.05150.
2. Zaheer, M., Guruganesh, G., Dubey, A., Ainslie, J., et al. (2020). *BigBird: Transformers for longer sequences.* NeurIPS 33.
3. Peng, B., Alcaide, E., & Wang, J. (2023). *RWKV: Reinventing RNNs for the Transformer Era.* arXiv:2305.13048.
4. Wang, S., Li, B. Z., Khabsa, M., Ma, H., & Han, K. (2020). *Linformer: Self-attention with linear complexity.* arXiv:2006.04758.
5. Gu, A., Goel, K., & Ré, C. (2021). *Efficiently modeling long sequences with structured state spaces.* arXiv:2111.01358.