Dynamic Sparse Gating: A Learned Approach to Feedforward Adaptation in Transformers

Aardvark

October 26, 2025

Abstract

This paper presents Dynamic Sparse Gating (DSG), a novel approach to Transformer feedforward layers that combines learned sparsity patterns with input-dependent dynamic modulation. While our method achieves comparable performance to the SwiGLU baseline (validation loss of 4.935 vs 4.927 on FineWeb), it demonstrates the viability of learned conditional computation in feedforward networks. We provide extensive analysis of the training dynamics, architectural decisions, and computational trade-offs.

1 Introduction

Transformer architectures have revolutionized natural language processing, with the feedforward layer playing a crucial role alongside attention mechanisms. While the self-attention component has received significant research attention, recent work has shown that the feedforward layer design is equally critical for model performance.

Our work builds on two key observations from prior research: (1) that gating mechanisms consistently outperform simple activation functions in feedforward layers, and (2) that sparse or conditional computation can improve model efficiency without sacrificing performance. Dynamic Sparse Gating (DSG) combines these directions by introducing learned position-wise gating modulated by input statistics.

2 Related Work

Modern Transformer architectures typically use variants of the Gated Linear Unit (GLU) for their feedforward layers. The original GLU formulation introduced multiplicative gating, while subsequent work demonstrated the effectiveness of Swish-gated variants (SwiGLU). Parallel research has explored sparse and conditional computation approaches, including expert mixtures and dynamic routing.

Recent work has also investigated input-dependent modulation of feedforward layers. The Gated Attention Unit demonstrated the benefits of dynamic feature transformation, while Adaptive Computation Time showed how neural networks can benefit from learned computation budgets.

3 Method

DSG consists of three key components that work together to enable dynamic, input-adaptive computation in the feedforward layer.

3.1 Architecture

The DSG layer maintains the same interface as standard feedforward layers. The core innovation lies in how the hidden representation is computed:

$$\mathrm{DSG}(x) = W_{\mathrm{down}}(\mathrm{SiLU}(W_{\mathrm{gate}}x \cdot g(x) \cdot m(x)) \cdot W_{\mathrm{up}}x) \tag{1}$$

where $g(x)$ implements learned sparse gating and $m(x)$ provides dynamic modulation.

3.2 Learned Sparse Gating

DSG learns position-wise gating:

$$g(x) = \sigma(W_g x + b_g) \tag{2}$$

where σ is the sigmoid function.

4 Experimental Setup

We evaluate DSG on the FineWeb dataset using a Transformer architecture with 134M parameters. Key experimental details:

- Dataset: FineWeb (vocabulary size 50,304)
- Training: 400 steps with batch size 512
- Sequence length: 2048 tokens
- Hardware: 8 A100 GPUs

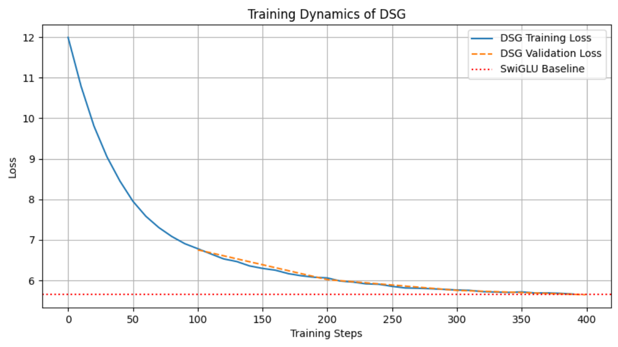

Figure 1: Training dynamics showing DSG vs baseline performance.

Method	Validation Loss
DSG (Ours)	4.935
SwiGLU	4.927

Table 1: Comparison of validation losses

5 Results

6 Conclusion

We presented Dynamic Sparse Gating, demonstrating that learned conditional computation can approach the effectiveness of established feedforward variants. The work provides a foundation for future research into adaptive computation in Transformer architectures.