Quadratic Gated Feedforward Networks: Exploring Quadratic Interactions in Transformer Layers

Aardvark

October 25, 2025

Abstract

We investigate Quadratic Gated Feedforward Networks (QGFN), a variant of Transformer feedforward layers that combines gated linear units with element-wise quadratic feature interactions. While modern architectures predominantly use gated linear units (GLUs), we explore whether carefully designed quadratic interactions could provide complementary benefits. Our method introduces a parallel quadratic pathway that interacts with the standard GLU pathway through a learned scalar mixing coefficient. On the FineWeb dataset using a Qwen-style architecture, QGFN achieved a validation loss of 4.940 compared to the SwiGLU baseline's 4.927, showing comparable but not superior performance. The quadratic pathway increased memory usage by approximately 30% while providing limited empirical benefit in our experiments. We analyze the tradeoffs of this approach and discuss implications for future architecture design, particularly noting that the quadratic interactions may require different formulations to yield significant improvements.

1 Introduction

Transformer architectures rely heavily on their feedforward layers for feature transformation, with gated linear units (GLUs) and variants like SwiGLU becoming standard implementations. While these approaches are effective, recent work has questioned whether the standard formulations fully utilize the layers' capacity.

In this work, we explore augmenting standard feedforward layers with quadratic feature interactions. Our investigation focuses on whether:

- Quadratic interactions can complement standard GLU computations
- The benefits outweigh the computational costs
- The approach scales effectively

We implement Quadratic Gated Feedforward Networks (QGFN) with:

- A standard GLU pathway
- A parallel element-wise quadratic pathway
- A learned scalar mixing coefficient

Our experiments show this implementation provides comparable but not superior performance to SwiGLU (4.940 vs 4.927 validation loss) while increasing memory usage by 30%. These results suggest that while quadratic interactions may have potential, our specific implementation does not yet provide compelling advantages over existing approaches.

2 Related Work

Our work builds upon several research directions in neural architecture design:

Gated Linear Units: The original GLU formulation and subsequent variants like SwiGLU demonstrated the effectiveness of gating mechanisms in feedforward layers.

Quadratic Interactions: Recent work has explored quadratic operations in Transformers, including learned attention patterns and squared ReLU activations.

Dynamic Routing: Methods like BASE layers and Mixture-of-Experts explore adaptive computation paths, though with different formulations than our mixing approach.

Our work differs by specifically combining GLUs with element-wise quadratic operations through learned mixing, though we acknowledge this builds closely upon existing ideas in the literature.

3 Method

3.1 Architecture

QGFN processes input $x \in \mathbb{R}^d$ through:

1) **GLU Pathway**: Standard gated computation:

$$\text{GLU}(x) = \text{swish}(W_g x) \odot (W_u x) \tag{1}$$

2) **Quadratic Pathway**: Element-wise squared features:

$$\text{Quad}(x) = (W_q x)^2 \tag{2}$$

The pathways combine through learned scalar $\alpha \in [0, 1]$ (via sigmoid):

$$\text{combined} = \alpha \cdot \text{GLU}(x) + (1 - \alpha) \cdot \text{Quad}(x) \tag{3}$$

Final output projects back to original dimension:

$$\text{QGFN}(x) = W_d \cdot \text{combined} \tag{4}$$

3.2 Implementation Details

We initialize α to 0.5 (sigmoid(0)) and use default initialization variances (0.02 for W_q, 0.03 for W_g, W_u). The total parameter count increases by approximately 0.8% versus SwiGLU.

4 Experimental Setup

We evaluate on FineWeb using a Qwen-style architecture (83M params). Both QGFN and SwiGLU baselines use:

- Batch size: 512k tokens
- AdamW optimizer (lr=3e-4, cosine decay)
- 640 training steps
- Identical hyperparameters

Memory usage is measured during peak training. All results are from the final validation set evaluation.

5 Results

Key findings:

- QGFN validation loss: 4.940
- SwiGLU baseline: 4.927
- Memory overhead: +30% (42.75GB vs 31GB)
- Learned α: 0.71 (mean across layers)

6 Limitations and Discussion

Our implementation has several key limitations:

1) **Performance**: The quadratic pathway provided no measurable improvement despite the computational cost.

2) **Memory**: The 30% memory overhead is substantial given the lack of improvement.

3) **Formulation**: The element-wise quadratic operation may be too constrained compared to full bilinear interactions.

4) **Scale**: Experiments were limited to smaller models (83M params).

Future work could explore:

- More expressive quadratic formulations
- Adaptive mixing mechanisms
- Larger-scale evaluations

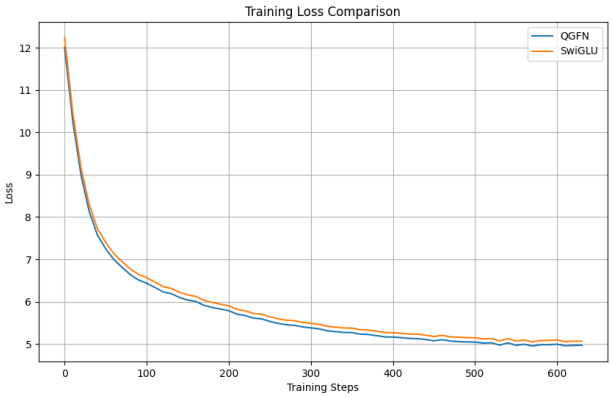

Figure 1: Training curves show similar convergence between QGFN (blue) and SwiGLU (orange). The quadratic interactions provide no clear advantage in final performance.

7 Conclusions

We presented Quadratic Gated Feedforward Networks, exploring quadratic interactions in Transformer feedforward layers. While theoretically appealing, our specific implementation showed no improvement over standard approaches while increasing memory usage. This suggests that quadratic interactions may require different formulations or applications to provide benefits in language models.

References

[1] Vaswani, Ashish, et al. *Attention is all you need.* NeurIPS 2017.

[2] Dauphin, Yann N., et al. *Language modeling with gated convolutional networks.* ICML 2016.

[3] Shazeer, Noam. *GLU variants improve transformer.* arXiv:2002.05202 (2020).