# GLU ATTENTION IMPROVE TRANSFORMER

**Wang Zehao**
Nanjing University
wangzehao_ai@163.com

## ABSTRACT

Gated Linear Units (GLU) have shown great potential in enhancing neural network performance. In this paper, I introduce a novel attention mechanism called GLU Attention, which introduces nonlinearity into the values of Attention. My experiments demonstrate that GLU Attention improves both model performance and convergence speed across text and vision modalities with zero additional parameters and negligible computational costs. GLU Attention is lightweight and can seamlessly integrate with other technologies, such as Flash Attention, Rotary Position Embedding (RoPE), and various Multi-Head Attention (MHA) variants such as Grouped-Query Attention (GQA). This project is open-sourced at github[1].

## 1 Introduction

Transformer[2] models have become the foundation of modern artificial intelligence. Transformer is a sequence-to-sequence model that uses Attention layer to capture relationships between tokens and Feed Forward Network (FFN) layer to perform transformations on each token. GLU FFN[3] outperforms the original FFN and has been adopted in popular open source Large Language Model (LLM) Llama 3[4]. My study shows GLU Attention outperforms original MHA. In MHA there is a softmax function introduce nonlinearity for querys and keys, but the values are projected by linear transformations. My study explores the integration of GLU nonlinearity into the values of MHA. Experiments show that adding GLU to MHA values can improve model performance and convergence speed. GLU Attention is a simple yet effective enhancement to the Transformer architecture, improving both training efficiency and model performance.

## 2 Backgrounds

### 2.1 Gated Linear Units

GLU were first introduced to improve performance by introducing nonlinearity and has been successfully applied in various architectures, including convolutional neural networks (CNN)[5] and transformer FFN layer[3].

GLU contains two inputs: a gate $g$ and a gated input $x$, along with one Rectified Linear Unit[6] such as $ReLU(x) = max(0, x)$ or $SiLU(x) = x * sigmoid(x)$. In this paper I use SiLU[7] as the Linear Unit, GLU is defined as:

$$GLU(x, g) = x * SiLU(g) \tag{1}$$

Or just split the last dimension of input $x$ into two parts, $x_1$ and $x_2$, and apply Rectified Linear Unit to $x_2$:

$$x_1, x_2 = split(x, dim = -1) \tag{2}$$

$$GLU(x) = x_1 * SiLU(x_2) \tag{3}$$

### 2.2 Multi-Head Attention

MHA is a key component of the Transformer architecture, enabling the model to focus on different parts of the input sequence simultaneously. The MHA layer has three inputs: queries $Q$, keys $K$, values $V$, and one output $O$. MHA applies three linear transformations $W_Q$, $W_K$, and $W_V$ to project the inputs into different subspaces for each attention head. A final linear transformation $W_O$ is used to project the output back to the original space. The MHA layer can be expressed as:

$$Q' = W_Q(Q) \tag{4}$$

$$K' = W_K(K) \tag{5}$$

$$V' = W_V(V) \tag{6}$$

$$O' = MHA(Q', K', V') \tag{7}$$

$$O = W_O(O') \tag{8}$$

In Multi-Head Self-Attention, the same input $X$ is used for queries, keys, and values. $Q = K = V = X$

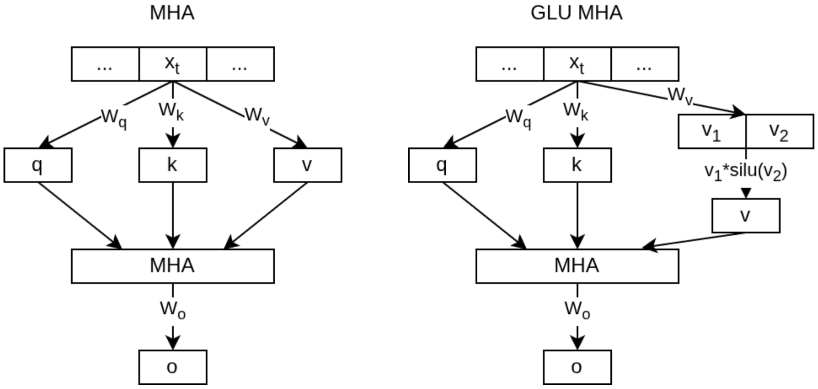

Figure 1: MHA and GLU MHA algorithm

## 3 Method

GLU Attention uses the projected value $V'$ as the intput of Equation (3). The GLU Attention can be expressed as:

$$V' = GLU(W_V(V)) \tag{9}$$

By replacing Equation (6) in MHA with Equation (9), while keeping other components unchanged, we obtain GLU Multi-Head Attention. To maintain the same number of parameters and computational costs, we use 4/3 of the original $W_V$ matrix output dimension and 2/3 of the original $W_O$ matrix input dimension.

## 4 Experiments

### 4.1 Models and Hyperparameters

I conducted experiments using two Transformer models: a baseline model with standard Multi-Head Attention (MHA), and a GLU Attention model with GLU Multi-Head Attention (GLU MHA). Both models consist of one embedding layer, one positional embedding layer, six transformer layers, and one linear classification layer. Each transformer layer contains a self-attention mechanism and a GLU feed-forward network (GLU FFN), with a model dimension of 384 and 8 attention heads.

To ensure a fair comparison, the projection and FFN linear layers are designed to match the number of parameters and FLOPs of classic transformers. In the GLU Attention model, the value projection layer has a shape of 384→512, and the output projection layer has a shape of 256→384, while other projection layers maintain a shape of 384→384. The GLU FFN consists of two linear layers: the first layer has a shape of 384→2048, and the second layer has a shape of 1024→384.

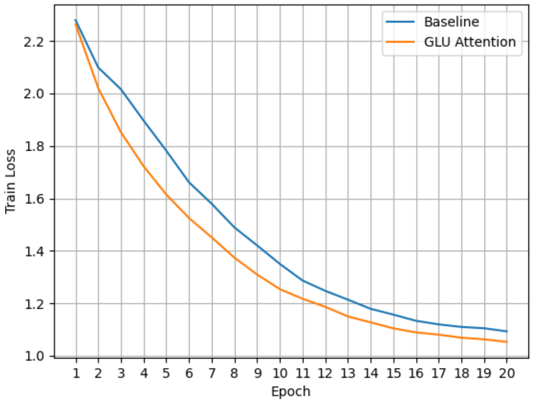

Figure 2: Cifar-10 training loss of each epoch. The lower the better.

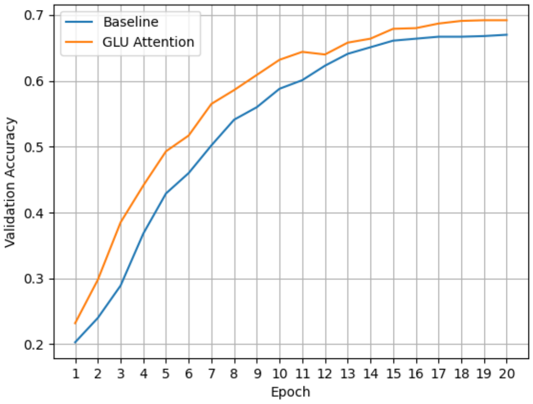

Figure 3: Cifar-10 validation accuracy of each epoch. The higher the better.

### 4.2 Cifar-10

I trained these models from scratch on the Cifar-10 dataset, a widely used benchmark for image classification. The training dataset consists of 60,000 32x32 color images across 10 classes, while the validation set consists of 10,000 images. I followed the standard ViT[8] procedure, dividing each 32x32x3 image into 64 patches of size 4x4x3. Training was conducted for 20 epochs with a batch size of 384. I used the AdamW optimizer with a learning rate of 1e-4 and a cosine annealing scheduler. The results are shown in Figure 2 and Figure 3. GLU Attention consistently outperformed the baseline model.

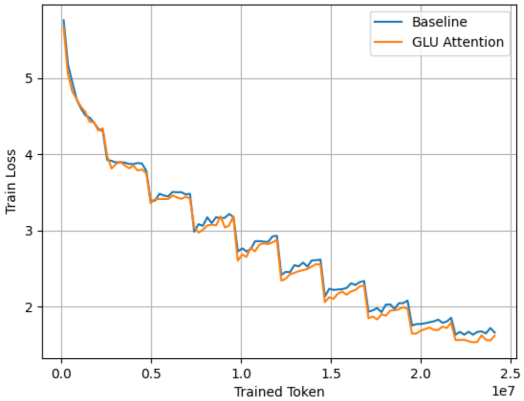

Figure 4: wikitext2 training loss for 10 epochs. The lower the better.

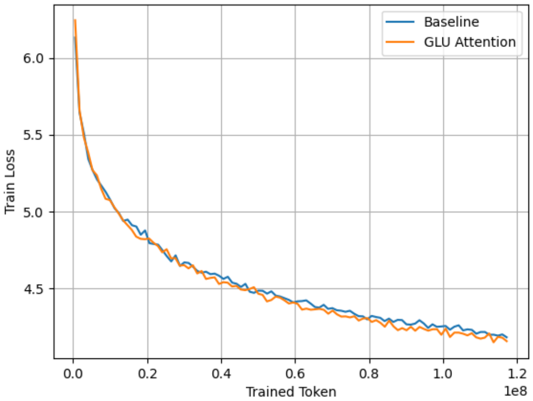

Figure 5: wikitext103 training loss for 1 epoch. The lower the better.

### 4.3 WikiText-2

I also trained these models from scratch on the WikiText-2 dataset which has 36,718 rows of token for language model pre-training, which is to predict the next token. I used the GPT-2 tokenizer to tokenize the text and applied the same training settings as in Cifar-10, except that the batch size was set to 1 and a causal mask was used to prevent the model from seeing future tokens. Training was conducted for 10 epochs. The results are shown in Figure 4. GLU Attention consistently outperformed the baseline model.

### 4.4 WikiText-103

Then I trained these models from scratch on the WikiText-103 dataset which has 1,801,350 rows of token using learning rate 1e-5 for 1 epoch. The results are shown in Figure 5. GLU Attention consistently outperformed the baseline model.

## 5 Conclusion

GLU Attention offers a straightforward yet impactful improvement to the Transformer architecture. By introducing nonlinearity into the values of MHA, it enhances model performance and convergence speed.

GLU Attention can be seamlessly integrated with other technologies, such as Flash Attention[9], RoPE[10], and various MHA variants like MQA and GQA[11], by simply adding a GLU function (Equation 3) after the value projection function and adjusting some parameters to accommodate the GLU function's property that output dimension is half of the input dimension.

## 6 Future Work

I highly recommend every researcher to test GLU Attention in your Transformers, as it is easy to adopt and provides a nearly cost-free performance boost. Future work could explore its application in different MHA variants with different FFN variants, on more datasets and tasks, as well as its scalability to larger models and datasets.

## References

[1] Wang Zehao. Glu attention github repository. `https://github.com/WangZehaoAI/GLU-Attention`, 2025.

[2] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention is all you need, 2023.

[3] Noam Shazeer. Glu variants improve transformer, 2020.

[4] Aaron Grattafiori et al. The llama 3 herd of models, 2024.

[5] Yann N. Dauphin, Angela Fan, Michael Auli, and David Grangier. Language modeling with gated convolutional networks, 2017.

[6] Kunihiko Fukushima. Cognitron: A self-organizing multilayered neural network, 1975.

[7] Stefan Elfwing, Eiji Uchibe, and Kenji Doya. Sigmoid-weighted linear units for neural network function approximation in reinforcement learning, 2017.

[8] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, Jakob Uszkoreit, and Neil Houlsby. An image is worth 16x16 words: Transformers for image recognition at scale, 2021.

[9] Tri Dao, Daniel Y. Fu, Stefano Ermon, Atri Rudra, and Christopher Ré. Flashattention: Fast and memory-efficient exact attention with io-awareness, 2022.

[10] Jianlin Su, Yu Lu, Shengfeng Pan, Ahmed Murtadha, Bo Wen, and Yunfeng Liu. Roformer: Enhanced transformer with rotary position embedding, 2023.

[11] Joshua Ainslie, James Lee-Thorp, Michiel de Jong, Yury Zemlyanskiy, Federico Lebrón, and Sumit Sanghai. Gqa: Training generalized multi-query transformer models from multi-head checkpoints, 2023.