Contents

GLU Variants Improve Transformer

Paper Link: https://arxiv.org/abs/2002.05202

Authors: Noam Shazeer (Google Brain)

Publication: ArXiv 2020

Overview

This paper explores variants of Gated Linear Units (GLU) as alternatives to traditional activation functions in Transformer feed-forward networks. The research demonstrates that several GLU variants, particularly SwiGLU and GEGLU, outperform standard activations like ReLU and GELU, leading to their adoption in major language models like PaLM and LLaMA.

Key Problem Addressed

Traditional Transformer feed-forward networks use simple activation functions that may not be optimal:

1. **ReLU Limitations**: Simple but can cause gradient issues and limited expressiveness
2. **GELU Performance**: Better than ReLU but still room for improvement
3. **Activation Function Choice**: Limited exploration of gating mechanisms in FFNs
4. **Model Quality**: Need for better activation functions to improve model performance

Background: Gated Linear Units (GLU)

The original GLU (Dauphin et al., 2016) consists of:

```
GLU(x) = (xW + b)   (xV + c)
```

Where: - `x` is the input - `W` and `V` are weight matrices - `b` and `c` are bias vectors - is the sigmoid function - denotes element-wise multiplication

GLU Variants Explored

The paper explores different activation functions in place of sigmoid:

1. Original GLU

```
GLU(x) = (xW + b)   (xV + c)
```

Uses sigmoid gating function.

2. ReGLU

```
ReGLU(x) = (xW + b)  ReLU(xV + c)
```

Replaces sigmoid with ReLU.

3. GEGLU

```
GEGLU(x) = (xW + b)  GELU(xV + c)
```

Uses GELU as the gating function.

4. SwiGLU

```
SwiGLU(x) = (xW + b)  Swish(xV + c)
```

Where Swish(x) = x · (x).

5. Bilinear

```
Bilinear(x) = (xW + b)  (xV + c)
```

No nonlinear gating (linear gating).

Integration into Transformer FFN

Standard FFN

```
FFN(x) = ReLU(xW  + b )W  + b
```

GLU-based FFN

```
FFN_GLU(x) = GLU_variant(x)W  + b
```

Where GLU_variant produces the gated representation.

Experimental Results

Model Setup

- **Architecture**: Transformer sequence-to-sequence model
- **Evaluation**: Perplexity on language modeling tasks

- **Comparison**: Against ReLU and GELU baselines

Key Findings

1. Performance Ranking Based on perplexity results: 1. **SwiGLU**: Best overall performance 2. **GEGLU**: Close second-best 3. **ReGLU**: Moderate improvement 4. **GLU (original)**: Some improvement 5. **Bilinear**: Minimal improvement

2. Consistent Improvements

- GLU variants consistently outperform standard activations
- Improvements observed across different model sizes
- Benefits persist across different datasets

3. Computational Considerations

- GLU variants require more parameters (two linear projections vs. one)
- Computational cost is higher but manageable
- Quality improvements justify the additional cost

Why GLU Variants Work

1. Gating Mechanism

- **Selective Activation**: Gates control which information flows through
- **Dynamic Filtering**: Different gates for different inputs
- **Improved Expressiveness**: More complex activation patterns

2. Two-Path Design

- **Information Path**: One projection provides information
- **Gate Path**: Other projection controls information flow
- **Interaction**: Element-wise multiplication creates rich interactions

3. Smoother Gradients

- **Better Gradient Flow**: Gating can help with gradient problems
- **Reduced Saturation**: Less likely to saturate than simple activations
- **Improved Training**: Better optimization properties

Practical Impact

1. Model Adoption

- **PaLM**: Google's large language model uses SwiGLU
- **LLaMA**: Meta's models adopted SwiGLU
- **Industry Standard**: Becoming common in modern LLMs

2. **Performance Improvements**

- **Perplexity**: Consistent improvements in language modeling
- **Quality**: Better text generation quality
- **Efficiency**: Better parameter utilization

3. **Design Principles**

- **Activation Function Selection**: Systematic approach to choosing activations
- **Gating Benefits**: Demonstrated value of gating mechanisms
- **Empirical Validation**: Thorough experimental validation

Implementation Details

Parameter Count

GLU variants require more parameters: - **Standard FFN**: d_model × d_ff parameters - **GLU FFN**: 2 × d_model × d_ff parameters (for two projections)

Computational Cost

- **Forward Pass**: Approximately 2x computational cost
- **Memory**: Additional memory for intermediate activations
- **Training**: Slightly longer training time

Optimization

- **Learning Rate**: May need adjustment for GLU variants
- **Initialization**: Careful initialization of both projections
- **Regularization**: May benefit from appropriate regularization

Comparison with Other Activations

vs. ReLU

- **Performance**: Significant improvements over ReLU
- **Complexity**: Higher computational cost
- **Gradient Flow**: Better gradient properties

vs. GELU

- **Performance**: Moderate improvements over GELU
- **Smoothness**: Both provide smooth activations
- **Expressiveness**: GLU variants more expressive

vs. Swish

- **SwiGLU vs. Swish**: SwiGLU consistently better
- **Mechanism**: Gating provides additional benefits
- **Robustness**: More robust across different settings

Limitations and Considerations

1. Computational Overhead

- **Parameter Count**: Doubled parameter count in FFN
- **Memory Usage**: Higher memory requirements
- **Training Time**: Longer training times

2. Hyperparameter Sensitivity

- **Initialization**: Sensitive to initialization strategy
- **Learning Rate**: May require tuning
- **Regularization**: Need appropriate regularization

3. Architecture Dependence

- **Model Size**: Benefits may vary with model size
- **Task Dependence**: Performance may vary by task
- **Dataset Sensitivity**: Results may depend on dataset

Key Insights

1. Gating is Powerful

Gating mechanisms provide significant benefits in neural networks, not just in RNNs/LSTMs.

2. Systematic Exploration

Systematic exploration of activation functions can yield significant improvements.

3. Trade-offs Matter

Performance improvements often come with computational trade-offs that need to be considered.

4. Simple Changes, Big Impact

Small architectural changes can have large performance impacts.

Impact on the Field

1. Activation Function Research

- **Renewed Interest**: Sparked interest in activation function research
- **Systematic Approach**: Encouraged systematic exploration
- **Practical Focus**: Emphasis on practical improvements

2. Model Architecture

- **Standard Practice**: GLU variants becoming standard
- **Design Principles**: Influenced FFN design principles
- **Quality Improvements**: Contributed to overall model quality

3. Production Systems

- **Real-world Adoption**: Adopted in production systems
- **Scalability**: Proven to scale to large models
- **Reliability**: Demonstrated reliability in practice

Why Read This Paper

This paper is essential reading because:

1. **Practical Impact**: Directly improves model performance
2. **Simple Implementation**: Easy to implement and adopt
3. **Systematic Approach**: Demonstrates systematic exploration methodology
4. **Wide Adoption**: Understanding basis for modern LLM designs
5. **Design Principles**: Provides insights into effective activation design

Key Takeaways

1. **Gating Works**: Gating mechanisms are effective in feed-forward networks
2. **SwiGLU is Best**: SwiGLU consistently outperforms other variants
3. **Cost-Benefit**: Performance improvements justify computational costs
4. **Systematic Exploration**: Systematic exploration of design choices pays off
5. **Practical Adoption**: Simple architectural changes can have broad impact

GLU Variants represents a successful example of how systematic exploration of architectural components can lead to practical improvements that are widely adopted in the field, demonstrating the value of rigorous empirical evaluation in neural network design.