The Inhibitor: ReLU and Addition-Based Attention for Efficient Transformers (Student Abstract)

Rickard Brännvall[1,2]

[1]Computer Science Department, RISE Research Institutes of Sweden
[2]Machine Learning Group, Luleå University of Technology, Sweden
rickard.brannvall@ri.se

Abstract

To enhance the computational efficiency of quantized Transformers, we replace the dot-product and Softmax-based attention with an alternative mechanism involving addition and ReLU activation only. This side-steps the expansion to double precision often required by matrix multiplication and avoids costly Softmax evaluations but maintains much of the core functionality of conventional dot-product attention. It can enable more efficient execution and support larger quantized Transformer models on resource-constrained hardware or alternative arithmetic systems like homomorphic encryption. Training experiments on four common benchmark tasks show test set prediction scores comparable to those of conventional Transformers with dot-product attention. Our scaling experiments also suggest significant computational savings, both in plaintext and under encryption.

The ReLU and addition-based attention mechanism introduced in this paper may enable privacy-preserving AI applications operating under homomorphic encryption by avoiding the costly multiplication of encrypted variables.

Introduction

Transformer models (Vaswani et al. 2017) have achieved great success in various ML tasks and are the foundation for modern large-scale language models. They use an attention mechanism to identify connections between different elements in a sequence, which conventionally takes the dot-product between the *query* and *key* matrices before passing the results through a Softmax function. Many alternative formulations have been proposed, such as the Fastformer (Wu et al. 2021) based on additive attention, or the ReLUformer (Shen et al. 2023) that uses ReLU activation in place of Softmax. These architectures share elements with what is proposed in this note; however, they still apply attention by a matrix multiplication with the *value* matrix. Extending our earlier work with ReLU and addition-based gated RNNs (Brännvall et al. 2023), we here propose the Inhibitor attention mechanism, which is designed to not rely on variable-to-variable multiplication and Softmax activation. These are particularly challenging operations under homomorphic encryption, an arithmetic system that permits calculation with encrypted variables without access to the secret key.

	Adding	MNIST	IMDB	IAMW
Dot-prod Attention	0.11%	98.2%	87.2%	17.9
Inhibitor Attention	0.12%	97.9%	87.3%	18.1

Table 1: The Inhibitor shows comparable performance to conventional attention for Transformers trained on four standard tasks (for mse, acc, acc, and edit distance, respectively).

Method

The Transformer architecture consists of repeated blocks of self-attention and feed-forward networks (FFNs). In the simple decoder application, each block takes as input a sequence embedding, $X \in \mathbb{R}^{n \times d}$, where n is the length and d is the dimension. The embedding is transformed into *query*, *key* and *value*, by multiplication with weight matrices according to $Q = XW_Q$, $K = XW_K$, and $V = XW_V$.

Attention scores are calculated by passing the scaled dot-product of *query* and *key* through the Softmax function,

$$S = \mathrm{Softmax}\left(QK^T/\sqrt{d}\right) \tag{1}$$

which is then used to form a weighted sum of *values* as output, $H = SV$, exploiting that each row of S is normalized to sum to one. The FFN has two fully connected layers with weight matrix multiplication and ReLU activation. Each Transformer block typically also has one or more layer-normalizations that stabilize the gradient flow.

We propose to alternatively calculate *attention scores* as

$$Z_{ij} = \sum_k \frac{1}{\gamma} |Q_{ik} - K_{jk}| \tag{2}$$

where mixing between *query* and *key* is based on their absolute difference instead of dot-product. We have thus replaced the cosine (dot-prod) distance of conventional attention with Manhattan distance. In place of softmax multiplication, we subtract *attention scores* from *values* inside a ReLu function,

$$H_{ik} = \sum_j \left(V_{jk} - Z'_{ij}\right)^+ \tag{3}$$

with $Z' = (Z - \alpha)^+$, a shifted score, such that entries of V for which Z' are large are zeroed out and do not contribute to the sum. We call this mechanism *inhibition* as it is reminiscent of subtractive inhibition in biological neurons. FFN and normalization are left unchanged.

Results

Benchmark comparisons. We carried out numerical experiments that trained Transformer models based on the Inhibitor and the conventional dot-product attention on four standard tasks. The aim was not to achieve SotA results but rather to examine if the Inhibitor mechanism would perform comparably on a set of familiar tasks, which is why we used simple set-ups without hyperparameter tuning. From Table 1, which reports the results, we note that for each task the two alternative attention mechanisms score very similarly. Indeed, none of the differences are significant at 95% confidence (over at least 20 repetitions for each experiment). For experiments with the Inhibitor, we used a shifted score with $\alpha = 0.5$ and scaling factor $\gamma = \sqrt{d}$.

The **adding** problem was introduced by Hochreiter and Schmidhuber (1997) to test long-term memory for RNNs. It takes two inputs: a sequence of random numbers and a two-hot sequence (which, in our experiments, each is 100 long). The ground truth is simply the dot-product of the two inputs as vectors, which, rather obviously, is not a challenging task for a conventional (dot-prod-based) Transformer. It is encouraging that the addition-based Inhibitor performs just as well on this task (which is hard for, e.g., RNNs to solve).

We also train one-layer Transformers for the **MNIST** handwritten digit recognition task (LeCun and Cortes 2010) and **IMDB** movie-review sentiment analysis task (Maas et al. 2011), which are simple go-to benchmark task for image classification and text analysis, respectively. Although far for SotA, we note that our simple models achieve decent accuracy for both attention mechanisms, where the differences in performance are not significant.

The final task, labeled **IAMW** in Table 1, uses the IAM Handwriting Database (Marti and Bunke 2002), which is a collection of handwritten words written by more than 700 writers. The model first applies two convolutional layers followed by a Transformer and uses Connectionist Temporal Classification (CTC) loss (Graves et al. 2006) as an end-point layer to predict the text and edit distance as used as the evaluation metric. Again, differences are small.

Scaling experiments. Next, we wanted to examine and compare the scaling properties for the proposed mechanism under two identified scenarios: i) quantization with integer arithmetics and ii) homomorphic encryption. Therefore, the two alternative attention mechanisms were implemented directly in low-level code rather than high-level ML libraries, where built-in optimizations for conventional models and design choices would bias a comparison.

For the plaintext experiment, we used integer 16-bit arithmetics implemented in the Rust programming language, which gives detailed low-level control over circuits and supports advanced time benchmarking through the Criterion package. The encrypted implementation uses the TFHE scheme as supported in the Concrete library for Python (Zama 2022). We here used considerably smaller networks where the embedding dimension was limited to size 2. The encrypted circuits were implemented for integers with up to 8-bit precision.

	32	64	128	256
Dot-prod Attention	98.6 µs	330 µs	1.2 ms	4.48 ms
Inhibitor Attention	63.1 µs	178 µs	577 µs	2.5 ms

Table 2: Estimated plaintext execution time on CPU for four different sequence lengths (with fixed size single head).

	2	4	8	16
Dot-prod Attention	2.68 s	22.4 s	107 s	828 s
Inhibitor Attention	0.749 s	8.56 s	23.8 s	127 s

Table 3: Estimated encrypted execution time on CPU for four different sequence lengths (with fixed size single head).

The results indicate that the proposed Inhibitor mechanism can have a significant advantage, with i) 30%–50% saving for the plaintext implementation on CPU as per Table 2, and ii) a factor 3–6 under encryption with TFHE as per table 3. Timing estimates are averaged over 20 repeated experiments and are significant at the 95% confidence level.

Conclusions. The proposed Inhibitor mechanism allows straightforward quantization as equations 2 and 3 naturally support integer implementation. While experiment results are promising on simple training tasks, *for future work*, it is necessary to examine performance under more challenging settings, for example, by pre-training a much larger Transformer model and testing on modern NLP benchmark tasks.

References

Brännvall, R.; Forsgren, H.; Sandin, F.; and Liwicki, M. 2023. ReLU and Addition-based Gated RNN. arXiv:2308.05629.

Graves, A.; Fernández, S.; Gomez, F.; and Schmidhuber, J. 2006. Connectionist Temporal Classification. In *Proc. ICML 2006*. New York, NY, USA: ACM.

Hochreiter, S.; and Schmidhuber, J. 1997. Long Short-Term Memory. *Neural Computation*, 9(8): 1735–1780.

LeCun, Y.; and Cortes, C. 2010. MNIST handwritten digit database. http://yann.lecun.com/exdb/mnist/. Accessed: 2023-08-15.

Maas, A. L.; et al. 2011. Learning Word Vectors for Sentiment Analysis. In *Proc. ACL 2023*. Portland, Oregon, USA.

Marti, U.-V.; and Bunke, H. 2002. The IAM-database: An English sentence database for offline handwriting recognition. *IJDAR*, 5: 39–46.

Shen, K.; Guo, J.; Tan, X.; Tang, S.; Wang, R.; and Bian, J. 2023. A Study on ReLU and Softmax in Transformer. arXiv:2302.06461.

Vaswani, A.; et al. 2017. Attention is All you Need. In *Advances in NeurIPS*, volume 30. Curran Associates, Inc.

Wu, C.; Wu, F.; Qi, T.; Huang, Y.; and Xie, X. 2021. Fastformer: Additive Attention Can Be All You Need. arXiv:2108.09084.

Zama. 2022. Concrete: TFHE Compiler for Python programs. https://github.com/zama-ai/concrete. Accessed: 2023-08-15.