# CompactifAI: Extreme Compression of Large Language Models using Quantum-Inspired Tensor Networks

Andrei Tomut[1,2], Saeed S. Jahromi[3,1], Abhijoy Sarkar[1], Uygar Kurt[1], Sukhbinder Singh[4], Faysal Ishtiaq[1], César Muñoz[1], Prabdeep Singh Bajaj[1], Ali Elborady[1], Gianni del Bimbo[1], Mehrazin Alizadeh[4], David Montero[1], Pablo Martín-Ramiro[1], Muhammad Ibrahim[1], Oussama Tahiri Alaoui[1], John Malcolm[4], Borja Aizpurua[1], Samuel Mugel[4], and Román Orús[1,3,5]

1- Multiverse Computing, Parque Cientifico y Tecnológico de Gipuzkoa, Paseo de Miramón, 170 2° Planta, 20014 Donostia / San Sebastián, Spain

2- Catalan Institute of Nanoscience and Nanotechnology (ICN2), CSIC and The Barcelona Institute of Science and Technology, Campus UAB, 08193 Bellaterra, Catalonia, Spain

3- Donostia International Physics Center, Paseo Manuel de Lardizabal 4, E-20018 San Sebastián, Spain

4- Multiverse Computing, Centre for Social Innovation, 192 Spadina Avenue Suite 509, Toronto, ON M5T 2C2, Canada

5- Ikerbasque Foundation for Science, Maria Diaz de Haro 3, E-48013 Bilbao, Spain

**Abstract**. Large Language Models (LLMs) like ChatGPT and LlaMA offer immense opportunities but face challenges due to their vast size, leading to high training/inference costs and energy demands. Traditional compression methods focus on reducing neurons or precision. We introduce *CompactifAI*, a novel LLM compression using quantum-inspired Tensor Networks to compress the model's correlation space. Testing on LlaMA-2 7B showed a 93% memory and 70% parameter reduction, with minimal accuracy loss (2-3%) and significant training (50%) and inference (25%) speedups.

## 1 Introduction

Large Language Models (LLMs) like OpenAI's ChatGPT [1], Meta's LlaMA [2], and Google's BERT [3], based on the transformer architecture [4], have revolutionized artificial intelligence, enabling unprecedented human-machine interactions across sectors and attracting substantial investments.

However, LLMs face significant challenges due to their vast size and enormous energy consumption for training. Training models like ChatGPT-3 reportedly cost $100 million in electricity [5], with expenses predicted to double every ten months. This unsustainable energy demand necessitates greener, more efficient solutions. Compression techniques like quantization [6], distillation [7], pruning [8], and low-rank approximations [9] have been proposed but often act as brute-force methods, making it challenging to control compression errors and leading to mixed results.

In this paper, we introduce *CompactifAI* [10], a novel LLM compression technique based on quantum-inspired Tensor Networks (TNs) [11]. Our method "tensorizes" the self-attention and multi-layer perceptron (MLP) layers using specific TNs, truncating model correlations. By controlling the TN bond dimension, we achieve significant reductions in memory size and the number of parameters while maintaining accuracy. This compression reduces energy and memory requirements, making training, retraining, and inference more efficient. Notably, tensorization drastically reduces GPU-CPU transfer times in distributed training, cutting training and inference times by 50% and 25%, respectively. A brief retraining allows the compressed model's accuracy to approach that of the original uncompressed version.

## 2 Method

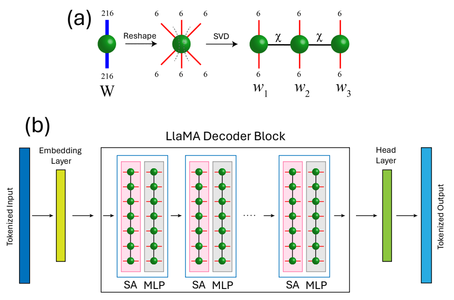
We compress Large Language Models (LLMs) by efficiently decomposing neural network weight matrices into Tensor Networks (TNs), specifically Matrix Product Operators (MPOs). While TNs have been applied in deep learning architectures before [12, 13, 14], this is, to our knowledge, their first use for LLM compression. Our approach begins with layer sensitivity profiling to identify layers suitable for compression. We then replace the trainable weights in these layers with MPOs characterized by a bond dimension $\chi$.

In the case of the LlaMA-2 chat model, as illustrated in Fig.1.1, we substitute the weight matrices in the Self-Attention (SA) and Multi-Layer Perceptron (MLP) layers with MPOs. These MPOs are constructed by performing sequential Singular Value Decompositions (SVDs) on the original weight matrices and retaining the top $\chi$ singular values at each step. This process effectively truncates correlations within each layer, preserving only the most relevant information while discarding less significant data.

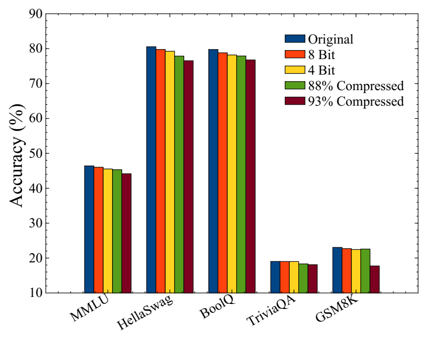
This method significantly reduces memory requirements, as storing the truncated MPOs-whose memory cost scales polynomially with the number of tensors-is far more efficient than storing the original weight matrices, whose size scales exponentially with the system size (i.e., the number of tensors), not with the matrix dimensions. The bond dimension $\chi$ controls the level of compression: smaller values of $\chi$ result in greater compression but may lead to reduced accuracy. Additionally, the choice of TN architecture and the number of decomposed tensors per layer serve as extra hyperparameters for the compressed model.

To maintain high accuracy, we include a rapid retraining phase, termed "healing," after truncating the MPOs. This step is crucial because layer-by-layer truncations might not be globally optimal, as they don't account for interactions with other layers. Retraining is more efficient than training the original model due to the reduced number of parameters, which decreases CPU-GPU transfer times in distributed training setups. As we demonstrate later, just a few retraining epochs enable the compressed model's accuracy to closely approach that of the original uncompressed version, but at a fraction of the computational cost.

Additionally, we optimize the MPO decomposition by tuning compression



1. (a) Diagram of a weight matrix $W$ decomposed into a MPO with $\chi$ bond dimension for truncation. (b) Schematic of LLM tensorization in the LlaMA model. The Self Attention and MLP layers' weight matrices are decomposed using MPOs with bond dimension $\chi$.



2. Accuracies of the original and compressed models for various tasks. The compressed models only deviate by 2%-3% compared to the original LlaMA-2 7B.

Fig. 1: [Color online] Illustration of the tensor network method (1) and its impact on model accuracy (2).

parameters for different layers, ensuring that each layer's compression is adjusted dynamically to maintain overall model accuracy (leveraging layer profiling tools). While this work focuses on compressing pre-trained models, training a model from scratch with MPO-replaced weight matrices could lead to faster training due to the smaller number of parameters, although it remains to be seen whether such models would achieve the same accuracy as those obtained via post-training compression.

## 3 Benchmark

We evaluated our method by compressing the LlaMA-2 7B model, a large language model with 7 billion parameters, pre-trained on over 2 trillion tokens, offering a context length of 4096, and fine-tuned with more than 1 million human annotations.

We created several compressed versions of the LlaMA-2 7B model using a combination of tensor network compression and quantization (see Table 1). The 8-bit and 4-bit quantized models were produced using the `bitsandbytes` quantization library. The 88% and 93% compressed models (in memory size) were obtained by applying *CompactifAI* to the float-16 quantized version of the original model. While both tensor network compression and quantization reduce model size, the tensorized models significantly reduce the number of parameters, allowing for greater size reduction. Unlike quantization, this parameter reduction is the key feature that enables significant speedups in both training and inference.

After compression, we retrained ("healed") the tensorized models to recover

| Model | Size | Parameters | Quantization |
|---|---|---|---|
| Original | 27.1 GB | 7B | float-32 |
| 8-bit | 6.8 GB | 7B | int-8 |
| 4-bit | 3.4 GB | 7B | int-4 |
| 88% | 4.1 GB | 2.1B | float-16 |
| 93% | 2.1 GB | 2.1B | mixed |

Table 1: Details of the models used in the benchmarks. The quantization in the 93% compressed model is a mix of float-16 for the tensorized layers and int-4 quantization for the not-tensorized layers.

any accuracy loss due to parameter reduction. We used generic chat datasets such as *Ultrachat*, *Alpaca*, and *OpenHermess* for retraining, implemented on a single AWS EC2 instance with 8 NVIDIA A10g GPUs using distributed training. Healing was performed for less than one epoch on these datasets. The 93% compressed model was obtained by further applying 4-bit quantization to the non-tensorized layers of the 88% compressed and healed model.

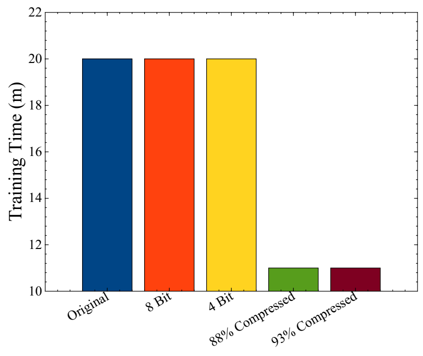
We benchmarked all models on tasks related to language understanding (MMLU), commonsense reasoning (HellaSwag), reading comprehension (BoolQ), world knowledge (TriviaQA), and math (GSM8K), using the `LLM Evaluation Harness` [15].

The compressed models maintained accuracies very close to the original LlaMA-2 7B model, deviating only by 2-3%, even with 70% fewer parameters (just 2.1 billion) (see Fig. 1.2 and Table2). This suggests that a substantial portion of the parameters in LLMs are redundant, and that large language models are heavily overparametrized. Notably, the accuracies of the tensorized models were achieved after training for only one epoch during healing; further fine-tuning could yield even better results, sometimes surpassing the original model's performance, as observed in tests with smaller models. Before the healing phase, the accuracy loss varies depending on the compression level. However, ongoing work explores methods such as context-aware SVD, which significantly mitigates accuracy drop by preserving task-relevant information during tensor decomposition. This would allow task-specific compression strategies that prioritize preserving accuracy for specific applications.

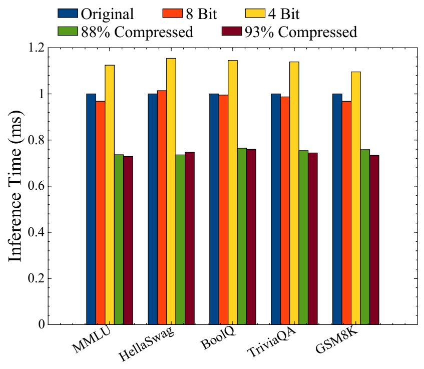
| Task/Model | Original | 8-bit | 4-bit | 88% | 93% |
|---|---|---|---|---|---|
| MMLU | 46.41 | 46.03 | 45.53 | 45.32 | 44.16 |
| HellaSwag | 80.55 | 79.77 | 79.25 | 77.87 | 76.54 |
| BoolQ | 79.76 | 78.81 | 78.19 | 77.90 | 76.77 |
| TriviaQA | 19.03 | 19.01 | 19.00 | 18.33 | 18.10 |
| GSM8K | 23.05 | 22.71 | 22.44 | 22.58 | 17.74 |

Table 2: Accuracies of the models in Table 1 for the MMLU, HellaSwag, BoolQ, TriviaQA, and GSM8K tasks.

Additionally, tensorized models demonstrated remarkable speedups in both training and inference. When training the models on the same data, tensorized models exhibited a 50% acceleration (i.e., twice as fast) compared to the original and purely quantized models (see Fig. 2.1). This significant speedup is attributed to the substantially smaller number of parameters, which reduces the data transfer time between GPUs and CPUs during distributed training. In inference, tensorized models achieved over 25% speedup compared to the original model (see Fig. 2.2). Conversely, the 4-bit quantized model slowed down inference by 13%, possibly due to inefficiencies in processing certain quantized operations on conventional GPUs.



1. Training time (in minutes) of the different models on the same amount of MMLU data used for healing the tensorized models. The tensorized models show 2× speedup with distributed training on eight A10g NVIDIA GPUs compared to both the original and purely quantized models.



2. Inference time (in milliseconds) of the different models for measuring the accuracies on various tasks. The tensorized models are 25% faster with distributed inference compared to the original model. Note that inference time for some quantized models is even higher than that of the original.

Fig. 2: [Color online] Training and inference times of the different models.

These results highlight the effectiveness of our tensor network compression method, *CompactifAI*, in reducing model size and computational costs while maintaining high accuracy, demonstrating its potential for making LLMs more efficient and sustainable.

## 4 Conclusions

We introduced and benchmarked *CompactifAI*, a novel compression method for Large Language Models (LLMs) based on quantum-inspired Tensor Networks (TNs). Our approach decomposes weight matrices in the Self-Attention and Multi-Layer Perceptron layers into Matrix Product Operators with a controllable bond dimension, effectively truncating correlations within the model. The compression rate is adjustable via the bond dimension, and model accuracy can be restored with a brief retraining ("healing") process.

By combining *CompactifAI* with quantization, we achieved a 93% reduction in memory size for the LlaMA-2 7B model, reduced the number of parameters by 70%, and accelerated training and inference times by 50% and 25%, respectively, all with only a small accuracy drop of 2%-3%. This surpasses what is achievable with other compression techniques and indicates that standard LLMs are heavily overparameterized.

Our method offers a more refined, controllable, and explainable compression compared to alternatives like pruning, distillation, quantization, and low-rank approximations, and it is compatible with these techniques. This work paves the way for the democratization of LLMs, enabling energy-efficient models that can be deployed on-premises without reliance on external servers, opening new possibilities for personalized AI. We believe *CompactifAI* and TN methods will play a fundamental role in the development of next-generation AI technology.

## References

[1] Samantha Lock. What is AI chatbot phenomenon ChatGPT and could it replace humans? *The Guardian*, (5), December 2022.

[2] Hugo et al. Touvron. LlaMA: Open and Efficient Foundation Language Models. 2023.

[3] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. 2018.

[4] Ashish et al. Vaswani. Attention is all you need. In I. Guyon, U. Von Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc., 2017.

[5] The bigger-is-better approach to AI is running out of road. *The Economist*, (21), June 2023.

[6] Benoit Jacob et al. Quantization and training of neural networks for efficient integer-arithmetic-only inference. *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 2704–2713, 2017.

[7] Geoffrey E. Hinton, Oriol Vinyals, and Jeffrey Dean. Distilling the knowledge in a neural network. *ArXiv*, abs/1503.02531, 2015.

[8] Song Han, Jeff Pool, John Tran, and William J. Dally. Learning both weights and connections for efficient neural network. In *Neural Information Processing Systems*, 2015.

[9] Max Jaderberg, Andrea Vedaldi, and Andrew Zisserman. Speeding up convolutional neural networks with low rank expansions. *ArXiv*, abs/1405.3866, 2014.

[10] See also. *Multiverse Computing CompactifAI*, 2023.

[11] Román Orús. A practical introduction to tensor networks: Matrix product states and projected entangled pair states. *Annals of Physics*, 349:117–158, oct 2014.

[12] Alexander Novikov, Dmitry Podoprikhin, Anton Osokin, and Dmitry Vetrov. Tensorizing neural networks, 2015.

[13] Saeed S Jahromi and Román Orús. Variational tensor neural networks for deep learning. *arXiv preprint arXiv:2211.14657*, 2022.

[14] Maolin Wang, Yu Pan, Zenglin Xu, Xiangli Yang, Guangxi Li, and Andrzej Cichocki. Tensor networks meet neural networks: A survey and future perspectives, 2023.

[15] Leo et al. Gao. A framework for few-shot language model evaluation, 12 2023.