Tachyum

White Paper

Mainstreaming Large Language Models
With 2-bit TAI Weights

2-bit Weights LLM

Introduction

Generative Pre-trained Transformer (GPT) models set themselves apart through breakthrough performance across complex language modelling tasks, in text generation, few-shot learning, reasoning, protein sequence modeling, but also by their extremely high computational and storage costs. Specifically, due to their massive size, even inference for large, highly accurate GPT models may require multiple performant GPUs to execute, which limits the usability of such models.

The capacities of language models increase dramatically by more than 1,000 times within a few years, from BERT 340 million parameters to the Megatron Turing 530 billion dense parameters and to the sparse Switch Transformer 1.6 trillion sparse parameters and lower precision (bfloat16). Scaling up language models has been incredibly successful. It significantly improves a model's performance on language and vision tasks, and the models demonstrate amazing few shot capabilities similar to that of human beings.

Efficient deployment of large language models (LLMs) necessitates low-bit quantization to minimize model size and inference cost. While low-bit integer formats (e.g., INT8/INT4) have been the conventional choice, emerging low-bit floating-point formats (e.g., FP8/4bit TAI) offer a compelling alternative. Striking a balance between computational efficiency and maintaining model quality is a formidable challenge.

Post Training Quantization of Large Language Models

The research in LLMs quantization has primarily focused on employing low-bit integer (INT) formats for quantization. The optimal format is influenced by a combination of various factors, including the static/dynamic nature of tensors, outlier distribution, and quantization bit-width.

In the presence of outliers, uniform quantization like INT8 or INT4, fail to accurately represent the main body of the data as they become skewed towards the outlier. This issue stems from the inherent assumption in these techniques of a uniform data distribution, an assumption that might not correspond to the actual data points distribution.

Floating-point (FP) methods like FP8 or 4bit TAI arise as more compelling alternatives. Unlike the fixed range of integer types, exponential and floating-point methods allow for adjusting the decimal point position, enabling dynamic scaling across activation maps and preserving important features.

For weight tensors with static distribution, INT quantization outperforms FP quantization at 8-bit but this advantage diminishes at 4-bit.

Unlike static weight tensors during inference, activation tensors are dynamic and change with each input. Therefore, calibration is an essential step in the quantization process for activation tensors to determine the appropriate scaling factors.

For activation tensors with dynamic distribution and significant outliers, FP quantization surpasses INT quantization due to FP having the ability to represent large values with lower precision and small values with higher precision.

Post Training Quantization Methods

Several lightweight optimization-based methods, where the weight of the model is updated during quantization, have been proposed in the literature. Among these, we utilize the following quantization techniques **GPTQ, Hadamard, SpQR** and others.

GPTQ is a weight quantization method based on approximate second-order information, that is both highly accurate and highly efficient. At a high level, our method follows the structure of state-of-the-art post-training quantization methods, by performing quantization layer-by-layer, solving a corresponding reconstruction problem for each layer.

Blocks of consecutive columns are quantized at a given step, using the inverse Hessian information stored in the Cholesky decomposition, and the remaining weights (blue) are updated at the end of the step.

Hadamard quantizers leverage advances in the field of randomized numerical linear algebra. For inference of LLMs, the activation outliers are the main reason for accuracy degradation. A Hadamard quantizer quantizes a transformed version of the activation matrix to suppress outliers. The transformation is a block diagonal Hadamard matrix, which spreads the information carried in outliers to its nearby entries of the matrix and thus reduces the numerical range of the outliers.

SpQR works by identifying and isolating outlier weights, which cause large quantization errors, and storing them in higher precision, while quantizing other weights to 4 bits.

Bert PTQ	FP32	TAI	TAI	TAI
		Vanilla	GPTQ	Hadamard
exact match	80,91	74,97	78,77	77,98
f1 score	88,23	84,23	86,91	86,06

Table 1
Results for post-training quantization of BERT LLM to TAI format with different quantization techniques

LLama2 PTQ	FP16	TAI	TAI
		GPTQ	SPQR
wikitext	5,48	6,35	5,86
ptb	7,96	8,90	23,88
c4	7,27	8,31	7,39

Table 2
Results for post-training quantization of LLama2 LLM in different formats and quantization techniques

Post Training Pruning

Generally speaking, there are two main approaches for neural network pruning: magnitude-based and impact based. Magnitude-based methods use the magnitude of weight to determine its importance and whether it should be pruned. Impact-based pruning methods remove weights based on how much their removal would impact the loss function, often using second-order information on the loss function.

Tachyum's Prodigy processor provides hardware support for both dense and block sparse matrix multiplication using FP8 for activations and 4-bit TAI and 2-bit TAI2 for weights thus allowing to scale deep learning via joint quantization and pruning.

In our investigation and testing we build on the **SparseGPT** algorithm. SparseGPT is the first accurate one-shot pruning method which works efficiently at the scale of models with 10-100 billion parameters. SparseGPT works by reducing the pruning problem to an extremely large-scale instance of sparse regression. It is based on a new approximate sparse regression solver, used to solve a layer-wise compression problem.

Joint Sparsification & Quantization

A key aspect of the layer-wise pruning problem is that both the mask as well as the remaining weights are optimized jointly in the column-wise greedy framework of GPTQ. Exactly solving it for larger layers is unrealistic, leading all existing approaches to resort to approximations.

A particularly popular approach is to separate the problem into mask selection and weight reconstruction. Concretely, this means to first choose a pruning mask M according to some saliency criterion, like the weight magnitude, and then optimize the remaining unpruned weights while keeping the mask unchanged.

Another pruning algorithm we have used in our testing was pruning method Wanda (Pruning by Weights and activations). By exploiting the recent observation of emergent large magnitude features in LLMs, the Wanda approach prunes weights with the smallest magnitudes multiplied by the corresponding input activations, on a per-output basis.

Bert PTQ + pruning	FP16	TAI	TAI2
exact match	80,9	78,92	77,31
f1 score	88,23	86,19	85,74

Table 3
Results for post-training quantization in TAI format and pruning to 8:4 sparsity of BERT LLM

LLama 2 PTQ+Pruning	Baseline	SparseGPT TAI2	Sparse GPT TAI	Wanda TAI2	Wanda TAI
		TAI2A8	TAIA8	WTAI2A8	WTAIA8
wikitext perplexity	5,68	8,63	6,59	6,61	8,57

Table 4
Results for post-training quantization in TAI format and pruning to 8:4 sparsity of LLama2

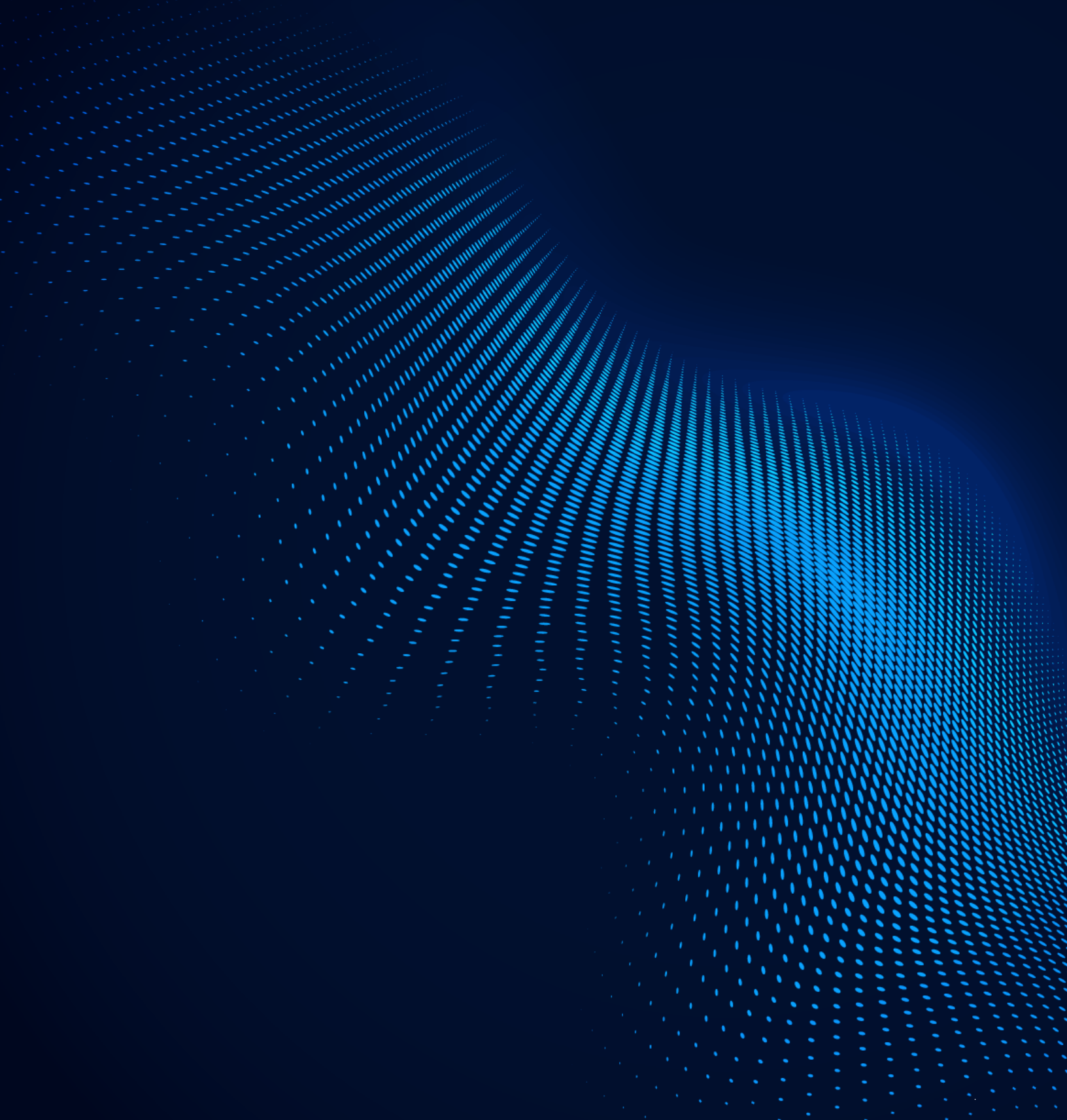
Conclusion

Tachyum addresses massive LLM models with capabilities that have dramatically increased by more than a thousand times over the past few years. Examples of these increases include ChatGPT-3.5 LLM with 175 billion parameters, the PALM LLM with 530 billion dense parameters and the Switch Transformer with 1.6 trillion sparse parameters.

For example, 1.6 trillion parameters Switch Transformer would require 52x NVIDIA H100 80GB GPUs at $41,789 each + 7 x $25,000 for Supermicro GPU servers = $2,348,028. In contrast, a $23,000 single Prodigy socket system with 2TB DDR5 DRAM could fit and run such big models and bring them into mainstream for generative AI applications.

AI systems built on Prodigy universal chips with 256PB DDR5 DRAM (Dynamic Random Access Memory) using FP8 (8-bit floating point) and 4-bit Tachyum AI (TAI) data formats enable fitting up to 100 quadrillion parameter models. It can serve more than 150,000x ChatGPT models or 610,000x PALM2 models and represents huge possibilities for using LLM as a mainstream technology in various industries from retail and e-commerce, marketing, finance, cyber security, military to healthcare including faster drug development or practical implementation of personalized medicine in hospitals.

By combining TAI 4-bit and effective 2-bit weights with FP8 per activation, we are capable of quantizing LLMs without much accuracy degradation. Combined with pruning we can avoid expensive multiplication while simultaneously reducing the size of the model by 4x to 8x, enabling generative AI models that can be applied in use cases from complex language modelling tasks, text generation, drug and chip design, few-shot learning and reasoning to protein sequence modelling. Completely new avenues of calculations are opened with Tachyum AI.



www.tachyum.com

Tachyum Inc., 8275 South Eastern Ave, Ste 233, Las Vegas, NV 89123, U.S.A.
Tachyum s.r.o., Karadžičova 14, CBC IV., 3rd floor, 821 08 Bratislava, Slovakia

Prodigy WP LLM v1.0_231114