CoMERA: **Co**mputing- and **M**emory-**E**fficient Training via **R**ank-**A**daptive Tensor Optimization

Zi Yang[1], Ziyue Liu[2], Samridhi Choudhary[3], Xinfeng Xie[4], Cao Gao[4], Siegfried Kunzmann[3], Zheng Zhang[2]

1. University at Albany, SUNY; 2. UC Santa Barbara; 3. Amazon Alexa AI; 4. Meta.

The cost of training large AI models is exploding!

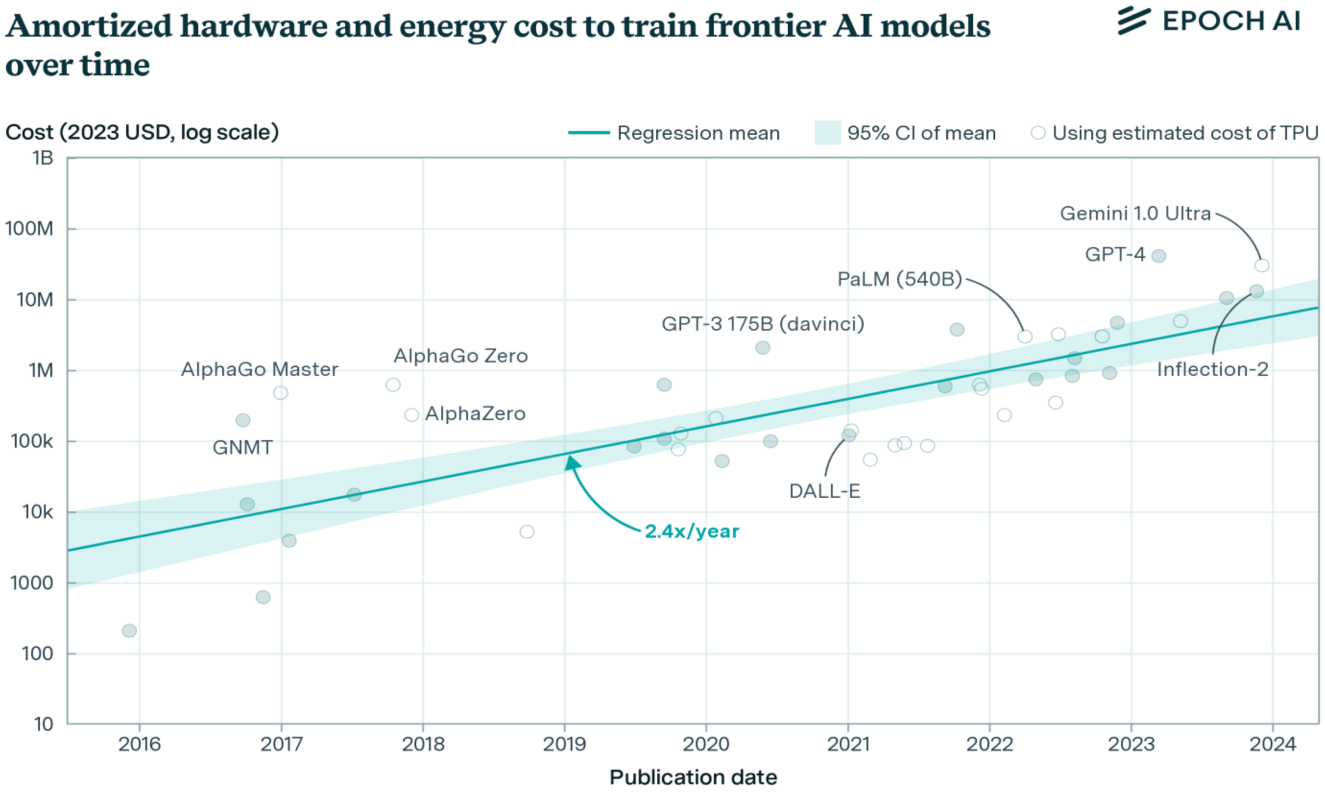


Figure from: https://epochai.org/blog/how-much-does-it-cost-to-train-frontier-ai-models

Contributions

We propose CoMERA for **Efficient Training** via **Tensor Compression**.

- **Multi-Objective Optimization for Rank-Adaptive Tensor-Training**:
  - Balance compression ratio and model performance
  - Customize model for a specific resource requirement
  - Partially capable of automatic architecture search
- **Performance Optimization of Tensor-Compressed Training**:
  - Optimize tensor-network contractions of tensorized linear layers and lookup of tensorized embedding tables
  - Eliminate the GPU backend overhead via CUDA Graph

CoMERA Training Framework

- **Multi-Objective Training Model**
  - Modified TT representation to use diagonal matrices to control TT ranks.

$$\mathcal{W} = \mathcal{G}_1 \times_{3,1} D_1 \times_{2,1} \mathcal{G}_2 \times_{3,1} \cdots \times_{3,1} D_{2d-1} \times \mathcal{G}_{2d},$$

  - Formulated a multi-objective problem to minimize loss and model size simultaneously

$$\min_{\mathcal{G},D} (L(\mathcal{G}, D), S(D))$$

- **Two-Stage Training Method:**
  - **Early-stage**: We start the training with the following linear scalarization formulation to gradually prune tensor ranks.

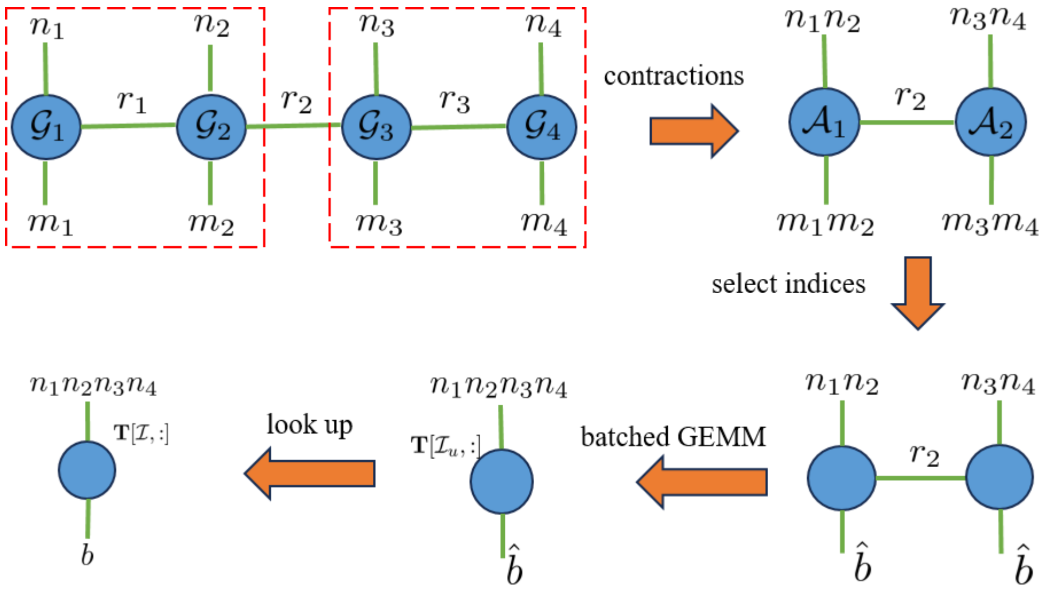
$$\min_{\mathcal{G},D} L(\mathcal{G}, D) + \gamma\, \hat{S}(D) + \beta \left| |\mathcal{G}| \right|^2,$$

  - **Late-stage (optional)**: Given target loss L_0 and target size S_0, we solve the following achievement scalarization for a Pareto point close to (L_0, S_0).

$$\min_{\mathcal{G},D} \max\{\omega\_1(L(\mathcal{G}, D) - L_0), \omega_2(S(D) - S_0)\} + \rho(L(\mathcal{G}, D) + \hat{S}(D)).$$

Performance Optimization of TTM Embedding Tables

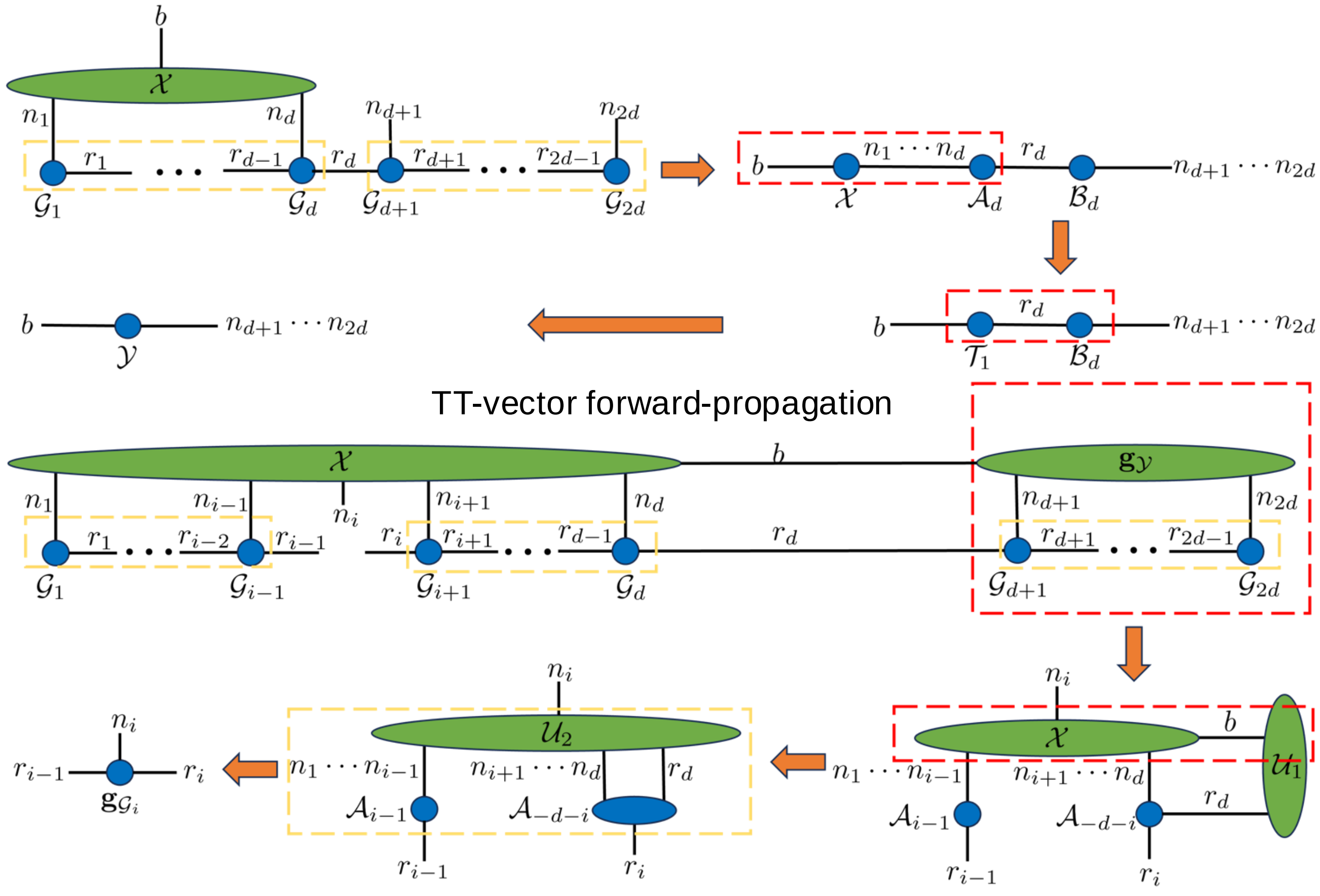
- The embedding table is represented in the Tensor-Train-Matrix format,
  $\mathcal{T} = \mathcal{G}_1 \times_{4,1} \mathcal{G}_2 \times_{4,1} \mathcal{G}_3 \times_{4,1} \mathcal{G}_4$
- A **fast** and **memory-efficient optimzied** TTM lookup by considering uniqueness on token level and tensor index level.



Optimized TTM embedding lookup

Optimized TT-vector multiplication for training

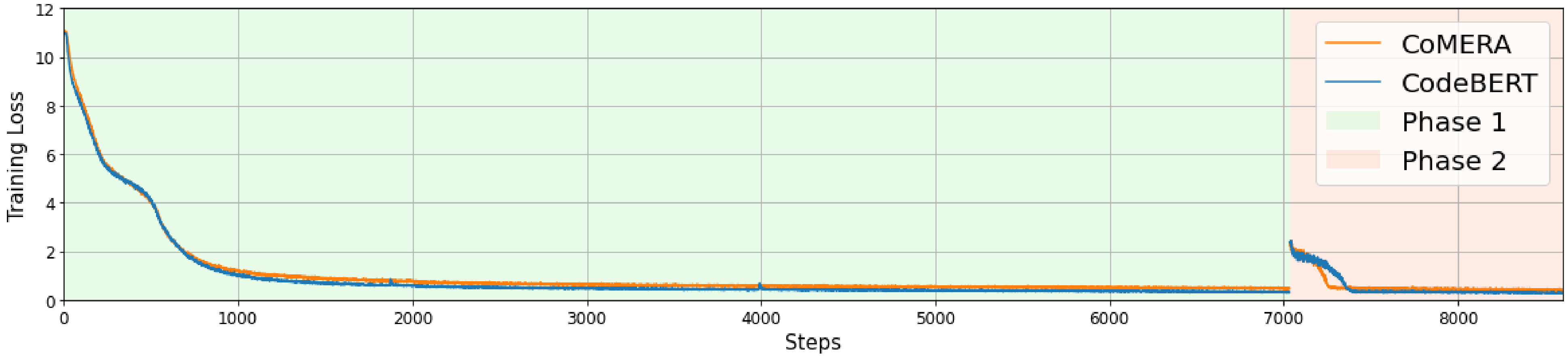
- We designed optimized forward- and back-propagation contraction paths for TT-vector multiplication.



TT-vector back-propagation: compute $g_{\mathcal{G}_i}, 1 \le i \le d$

CodeBERT$_{\text{LARGE}}$ of 357M parameters

- We train CodeBERT$_{\text{LARGE}}$ from scratch on CodeSearchNet dataset, a collection of 2M pairs and 6M pure code sequences and reach
  - A similar training loss curve
  - **4.23×** model compression ratio
  - **1.9-2.3×** training speedup in different training phases.



Pre-training loss curves of CodeBERT and CoMERA.

Six-encoder Transformer

- We test CoMERA to train a six-encoder Transformer on MNLI dataset. By rank-adaptive training, CoMERA achieves
  - **80×** compression ratio
  - **2-3×** speedup