Tensorized Neural Networks for Faster Training and Better Co-design

Chunxing Yin, Richard Vuduc
School of Computational Science and Engineering

Georgia Tech Center for Research into Novel Computing Hierarchies

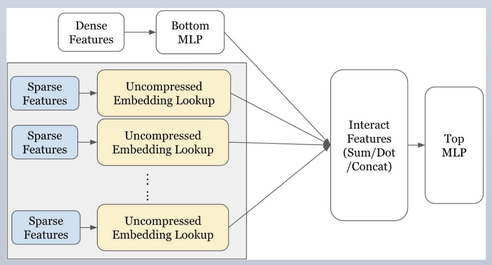
Motivation

- Deep neural networks (DNNs) are witnessing an unprecedented growth in data, model complexity, and the cost of infrastructure required for model training.
- The number of the SODA language models have over 175 billion parameters; similarly, the embedding tables in recommendation models are growing exponentially into the TB scale.
- Tensorization replaces layers of a neural network with an approximate and structured low-rank form.
- We design a **Tensor-Train** compression technique for **Deep Learning Recommendation Models** (DLRMs) and deep **convolutional neural networks** (CNNs).

Network Architecture

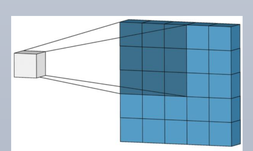
DLRM

- The Multi Layer Perceptron (MLP) layers are used to process continuous features, such as user age
- The Embedding Tables (EMBs) are used to process categorical features by encoding sparse, high-dimensional inputs into dense vector space
  - Encoding vector length ranges from 16 to 128
  - #of rows in EMBs ranges from hundreds to tens of millions



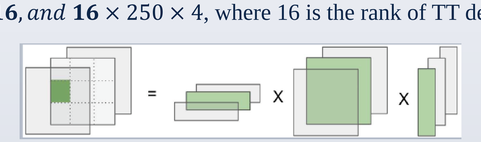
CNN

- The input tensor X with spatial dimension $H \times W$ and C feature maps is convolved with a 4-dimensional kernel tensor of size $D \times D \times C \times K$

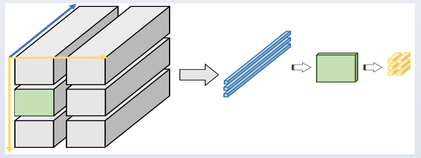


Tensor-Train Decomposition

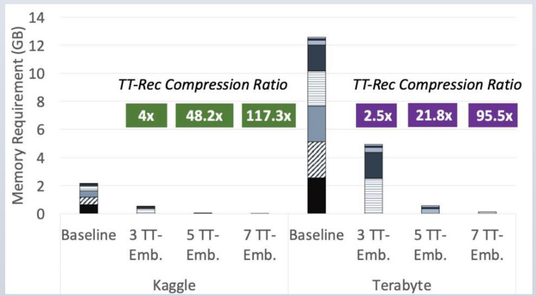
- Tensor Train (TT) decomposition is a technique to tensor decomposition by factorizing multidimensional data into product of small tensors
- A d-dimensional tensor A can be decomposed as $A(i_1, i_2, \dots, i_d) = G_1(:, i_1, :)G_2(:, i_2, :) \dots G_d(:, i_d, :)$, where G_k is a $3-d$ tensor called TT core.
- For example, to decompose a matrix of size $1M \times 64$, we choose TT cores of size $200 \times 4 \times \mathbf{16}, \mathbf{16} \times 200 \times 4 \times \mathbf{16}, and\ \mathbf{16} \times 250 \times 4$, where 16 is the rank of TT decomposition



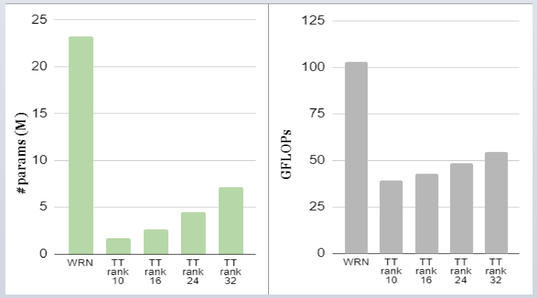
TT decomposition of a 3-d tensor



Decompose a large conv kernel into 3 small conv



Memory reduction of TT-DLRM

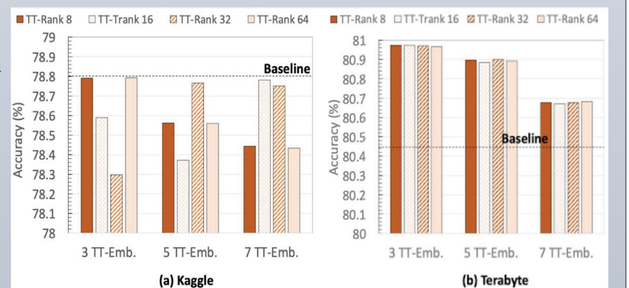


Memory and computation reduction of TT-Wide Resnet 28

Model Accuracy

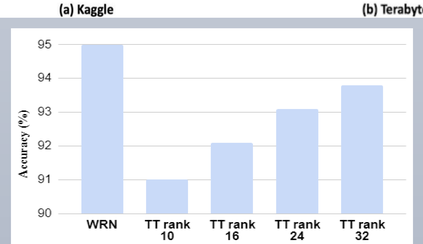
DLRM

- When compressing the largest 3,5, 7 embedding tables in TT format, the optimal TT-rank to achieve a nearly accuracy-neural results varies, with optimal rank of 8, 32 and 64 respectively
- Using larger TT-ranks produces more accurate models at the expense of lower compression ratio
- Compressing every 2 more embedding tables with TT format results in 0.3% loss of accuracy



Wide Resnet

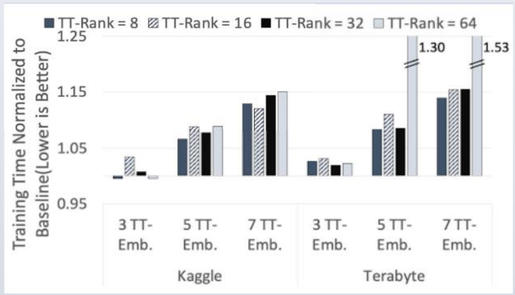
- When compressing with TT-rank 32, on CIFAR 10 the model achieves a compression ratio of 3.2 times, with a loss of accuracy of 1.2%
- Decreasing TT rank by 8 every time results in < 1% loss of accuracy



Kernel Efficiency

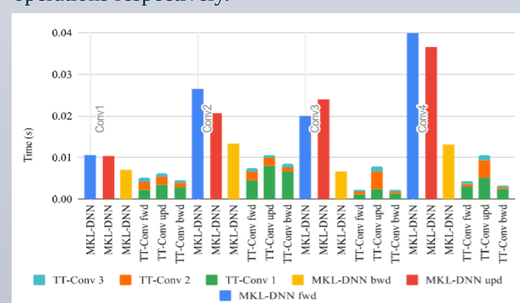
DLRM

- Our implementation achieves a memory footprint reduction of roughly 10,000x compared to Pytorch EmbeddingBag
- The training time using the optimal TT-rank increases by 12.5% and 11.8% for Kaggle and Terabyte respectively.



CNN

- Our implementation achieves an average speedup of 2.7x, 1.8x and 1.5x for forward, backward, and update kernels respectively for the smaller 2 conv blocks.
- On larger convolution blocks, our implementation shows a average of 9.4x, 3.45x, and 4.03x speedup on the three operations respectively.



Future work

- Improve kernel efficiency for different network config.
- Explore performance model of TT compressed networks
- Design algorithm for auto-selection TT related hyperparamters