Homework of ECE 594BB/194BB: Tensor Methods for Machine Learning and Big Data

Due Date: online submission due at 23:59pm 11/02/2021 Tuesday. No late submission

Submission requirement:

1. Please submit your zipped Matlab or python codes.
2. Please submit a PDF report to explain the key ideas of your code implementation and your results.
3. The report should be typed (either in word or in latex).

Guideline:

In this homework, you will be implementing at least two tensor factorization algorithms and use them to compress the model parameters of deep neural networks:

- You can choose one problem from Problems 1 & 2. Problem 3 is mandatary.
- You may get 5 bonus points if you implement and test all three tensor factorizations. Since you may also get 5 bonus points in Problem 1, you can get a maximum of 10 bonus points in total in this homework.
- While there are some tensor computing packages, **you are required to implement the codes independently and by yourself**, such that I can evaluate how much you have understood the technical details. You will lose credits if you use external tensor computation packages/codes.

Problem 1: CP compression of fully-connected (FC) layer [10 points]

In this problem, you will need to implement CP factorization and use it to compress the tensorized data array of a 784X512 FC layer, which is part of a classification model for the MNIST data set.

(a) **Implement a CP decomposition** with alternating least square. For a general d-way tensor with size I_1-by-I_2-by…-by-I_d and give a for a specified rank R, your algorithm should return the d factor matrices required in a CP format.

(b) **Test your algorithm with the dataset "cp_fc_layer.mat"**. We have already converted the FC weight matrix to a 4-way tensor of size [28,28,16,32]. Please increase the rank R from 20 to 50, and plot the Frobenius-norm error of your CP factorization as a function of R.

(c) [Not required, but you can get a maximum of 5 bonus points if you can solve this problem]. Implement a **stochastic gradient descent (SGD)** algorithm to perform CP factorization, and compare its performance with ALS (e.g., accuracy & CPU time) for given R. You are also encouraged to try different variance reduction methods or acceleration methods in SGD, and compare it with standard SGD and ALS methods.

Problem 2: Tucker compression of a convolution filter [10 points].

In this problem, you will need to implement Tucker factorization and use it to compress the tensorized format of a 9X64X64 convolution filter, which is part of a CNN classification model for the CIFAR-10 data set.

(a) Implement **Tucker** decomposition with high-order orthogonal iteration (HOOI) for a general d-way tensor with size I_1-by-I_2-by…-by-I_d for a specified multilinear rank (R_1, R_2, …, R_d)

(b) **Test your algorithm with the dataset "tucker_conv_tensor.mat"**. Please choose a set of multilinear ranks (R_1, R_2, …, R_d), and use a table to report the Frobenius-norm errors of your Tucker decomposition for each choice of multilinear rank.

(c) Choose one specific rank setting, and show how the error changes as the HOOI iteration index increases.

Problem 3: Tensor-train compression of an embedding table [10 points].

In this problem, you will need to implement the tensor-train factorization and use it to compress a 25,000X256 embedding table, which is the first layer of a natural language processing model. Embedding table is also widely used in the recommendation system of Facebook and Amazon.

(a) **Implement tensor-train decomposition for a general d-way tensor**. The function should input the original d-way tensor and a truncation error bound (see our lecture note), and output the resulting d TT cores.

(b) **Test your algorithm with the dataset "tt_embedding.mat"**. We have already folded the 2D embedding table to a 7-way tensor of size [5,8,25,25,4,8,8]. When you change the truncation error bound, you will get 7 TT cores with different size (and thus different number of variables in the TT cores). Then you can calculate both the actual Frobenious-norm error ε and a compression ratio S. Plot S as a function of ε (choose log or linear scale that can better illustrate your results).

References:

1) Lecture notes 3-6
2) T. Kolda and B. Bader, "Tensor decompositions and applications," SIAM Review, 2009
3) De Lathauwer, Lieven, Bart De Moor, and Joos Vandewalle. "A multilinear singular value decomposition." *SIAM J. Matrix Analysis and Applications* 21.4 (2000): 1253-1278.
4) De Lathauwer, Lieven, Bart De Moor, and Joos Vandewalle. "On the best rank-1 and rank-(r_1, r_2,..., r_n) approximation of higher-order tensors." SIAM journal on Matrix Analysis and Applications 21.4 (2000): 1324-1342
5) Oseledets, Ivan V. "Tensor-train decomposition." *SIAM J. Scientific Computing* 33.5 (2011): 2295-2317.