Title: Optimizing Data Layout for Training Deep Neural Networks

JX-Ma

2023/10/26

1 ABSTRACT

1. Training DNNs is both computation-intensive and memory-intensive.
2. Select a appropriate data layout for input tensor is considered to the most efficient means to reduce training time.
3. The article propose a efficient framework of data layout, it can select a appropriate data layout by different DNNs.
4. The average training performance is improved by 14.3% and 3.1% compared to using the two popular data layouts evenly. (The popular data layouts are NCHW and NHWC).

2 Introduction

1. DNN has a wide spectrum of application in many cases, but the train processing of model is very time-consuming.
2. To reduce train time, it widely adopt pruning scheme, include the shape pattern-based shape pruning, filter pruning, channel pruning, and kernel pruning.
3. The performance of NHWC data layout is better on CPU because of vectorization, and The performance of NCHW data layout is better on GPU because it can reduce expensive computation in GPU.
4. After pruning tensors, the sectional data in tensor without using but be load in cache. the data may storage a same cache block with required data of computing, it can result in under-utilization of cache and cache thrashing.

   Cache Thrashing: it refer to use memory without cache——each data read from memory instead of cache.

1. The article contribution:

1. It investigates the training execution time under different data layouts and different pruning schemes.
2. It proposes a cache estimation model that is able to effectively predict the cache performance under different execution scenarios.
3. It implements an auto-arbitration framework based on the proposed cache model.
4. It evaluates the proposed work on 5 DNN models with 4 different pruning strategies

3 BACKGROUND AND MOTIVATION

3.1 pruning strategies

1. shape pruning: It prune the elements of bottom-right and top-left at first channel in each filter tensor.
2. filter pruning: It prune some filter tensors.
3. channel pruning: It prune some channel in each filter tensor.
4. kernel pruning: it prune each filter tensor differently.

3.2 motivation

The article mainly resolve a problem that select a appropriate data layout to reduce the probability of cache thrashing after pruning tensors. the article propose a framework to resolve this problem

4 OUR APPROACH

1. The framework proposed in this paper can select the appropriate data layout according to the tensor size, pruning strategy, and hardware configuration.
2. Total Access computational formula: the $\dot{H}$ represent the height of output tensor.

$$TotalAccess = N \times \dot{H} \times \dot{W} \times R \times S \times C$$

1. the cache capacity computational formula: the l is cache block size and the n is size of cache block.

$$CacheCap = l \times n.$$

1. in the NCHW ,If cCAP < C X R X W - C X l ,and l < W, then the miss rate in this conditions is:

$$Missrate = \frac{N \times C \times R \times W \times (H - R + 1)}{l \times TotalAccess}$$

1. in the NCHW ,if $Ccap \geq C \times R \times W - C \times l$,then the miss rate in this conditions is:

$$Missrate = \frac{N \times C \times H \times W}{l \times TotalAccess}$$

1. in the NHWC ,If cCAP < R X S X C- R X l ,and l < C, then the miss rate in this conditions is:

$$Missrate = \frac{N \times S \times R \times C \times (H - R + 1) \times (W - S + 1)}{l \times TotalAccess}$$

else:

$$Missrate = \frac{N \times \frac{R \times S \times C}{l} \times (H - R + 1) \times (W - R + 1)}{TotalAccess}$$

5 CONCLUSION

This article mainly propose a framework to resolve under-utilization of cache and cache thrashing after pruning. according to experiment, they found different date layout of tensors can resolve this problem, so they propose a framework that can select appropriate data layout by pruning strategy and other parameter.