Ritwik Takkar
11/6/22

Paper Report:
Check-N-Run: a Checkpointing System for Training Deep Learning Recommendation Models by Assaf Eisenman et al.

ritwiktakkar.com

1 Summary

Checkpoints, commonly used within industry, take snapshots of an ML model to store in non-volatile memory to maintain speedy access to the model for training and preventing irrecoverable loss from failures. However, checkpoint frequency is bottlenecked by storage write bandwidth and capacity along with network bandwidth. *Check-N-Run* is a scalable checkpointing system developed at Facebook for training large ML models that aims to reduce the required write bandwidth and capacity, thereby improving checkpoint capabilities while reducing the total cost of ownership.

2 Strengths of the paper

It was nice to see this paper on arXiv as a pre-print unlike most others that are published as journal articles or conference proceedings. But part of it probably has to do with how, unlike seminal papers in CS of the past, today's ML field is so fast-paced that researchers can't afford wait to share their work until their reviewer's comments or publication date.

As someone not too familiar with ML, I was grateful for a step-by-step guide as to the several key criteria checkpoints must meet and why. These included accuracy (to avoid training accuracy degradation), frequency (maximize frequency to minimize re-training time), write bandwidth (minimize required bandwidth to support frequent checkpoints otherwise bottlenecked by network/storage capaccities), and storage capacity (reduce checkpoint size corresponding to model(s) due to hardware limits).

When describing a novel method aimed to improve any sort of complexity, whether space or time, I always appreciate visuals like graphs to convey improvements and diagrams to portray system design. The paper contains ample visuals, all of which are relevant according to me, to help the reader visualize what's elucidated in the text.

3 Weakness of the paper

The authors stated one of the four criteria for checkpoints as **storage capacity**. Specifically, they note standard compression algorithms such as Zstandard not being "useful enough for deep recommendation workloads." This sort of leads to a *chicken versus the egg* problem, suppose one came out with an amazing new compression algorithm to support the sort of workloads at FB, would that then render the work done to reduce storage capacity by Check-N-Run trivial? I wish a bit more discussion was present on this topic. Also, I don't fully understand what constitutes the *total cost of ownership*.

4 Future work opportunities

The authors discuss an approach (*intermittent incremental checkpoint*) in order to reduce prevent endless checkpoint size growth when optimizing in the form of incremental checkpointing. This approach stands to benefit from more accurate predicton models.