T. HOEFLER

Twelve ways to fool the masses when reporting performance of deep learning workloads! (not to be taken too seriously)

IPAM workshop “HPC for Computationally and Data-Intensive Problems” at UCLA, November 2018

Los Angeles, CA, USA

Deep learning and HPC

- Deep learning is HPC
  - In fact, it’s probably (soon?) bigger than traditional HPC
    *Definitely more money...*

- Interest in the HPC community is tremendous
  - Number of learning papers at HPC conferences seems to be growing exponentially
    *Besides at SC18, whut!?*

- Risk of unrealism
  - HPC people know how to do HPC
  - And deep learning is HPC, right?
    *Not quite ... while it’s really similar (tensor contractions)*
    *But it’s also quite different!*
“Statistical performance” vs. “hardware performance”

- Tradeoffs between those two
  - Very weird for HPC people – we always operated in double precision
    Mostly out of fear of rounding issues

- Deep learning shows how little accuracy one can get away with
  - Well, examples are drawn randomly from some distribution we don’t know …
  - Usually, noise is quite high …
  - So the computation doesn’t need to be higher precision than that noise
    Pretty obvious! In fact, it’s similar in scientific computing but in tighter bounds and not as well known

- But we HPC folks like flop/s! Or maybe now just ops or even aiops? Whatever, fast compute!
  - A humorous guide to floptimization
  - Twelve rules to help present your (not so great?) results in a much better light
1) Ignore accuracy when scaling up!

- Too obvious for this audience
  - Was very popular in 2015!

- Surprisingly many (still) do this

HPC picking up!

Learning community’s self-correction (Y. LeCun)

Scalability without a good baseline? (D. Bailey)
2) Do not report test accuracy!

- Training accuracy is sufficient isn’t it?
3) Do not report all training runs needed to tune hyperparameters!

- Report the best run – SGD is a bit fragile, so don’t worry
  
  *At the end, the minutes for the final run matter most!*

---

**Observed model performance**

**flop/s!**

**Your model**

**Suggested Hyperparameters**
4) Compare outdated hardware with special-purpose hardware!

- **Tesla K20 in 2018!?**
  
  *Even though the older machines would win the beauty contest!*

VS.
5) Show only kernels/subsets when scaling!

- Run layers or communication kernels in isolation
  - Avoids issues with accuracy completely 😊

Doesn’t that look a bit like GoogLeNet?

VS.
6) Do not consider I/O!

- Reading the data? Nah, make sure it’s staged in memory when the benchmark starts!
7) Report highest ops numbers (whatever that means)!

- Yes, we’re talking ops today, 64-bit flops was so yesterday!
  - If we don’t achieve a target fast enough, let’s redefine it!
    
    And never talk about how many more of those ops one needs to find a solution, it’s all about the rate, op/s!

- Actually, my laptop achieves an “exaop”:
  - each of the 3e9 transistors switching a binary digit each at 2.4e9 Hz

VS.
8) Show performance when enabling option set A and show accuracy when enabling option set B!

- Pretty cool idea isn’t it? Hyperparameters sometimes conflict
  
  *So always tune the to show the best result, whatever the result shall be!*
9) Train on (unreasonably) large inputs!

- The pinnacle of floptimization! Very hard to catch!
  
  *But Dr. Catlock Holmes below can catch it.*

VS.

Low-resolution cat (244x244 – 1 Gflop/example)

High-resolution cat (8kx8x – 1 Tflop/example)
10) Run training just for the right time!

- Train for fixed wall-time when scaling processors
  - so when you use twice as many processors you get twice as many flop/s!
  
  *But who cares about application speedup?*
11) Minibatch sizing for fun and profit – weak vs. strong scaling.

- All DL is strong scaling – limited model and limited data
- So just redefine the terms relative to minibatches:
  - Weak scaling keeps MB size per process constant – overall grows (less iterations per epoch, duh!)
  - Strong scaling keeps overall MB size constant (better but harder)
- Microbatching is not a problem!
12) Select carefully how to compare to the state of the art!

- Compare either time to solution or accuracy if both together don’t look strong!

There used to be conventions but let’s redefine them.