Applause from Vikram Jha and 14 others

Yaroslav Bulatov Follow
Feb 9 · 5 min read

OTODSCINOs: The Root Cause of Slow Neural Net Training

I've been interested in the fundamentals of numerical optimization ever since dropping in Stephen Boyd's "Convex Optimization" class in Stanford years ago. Theory developed in this eld can be used to answer questions faced by today's practitioners, in particular, the question of—"Why does it take so long to train neural nets?"

Conv nets trained in the 80s on handwritten digits took about 1 week to train on hardware of the day. Today's research conferences want your methods demonstrated on ImageNet-sized datasets, which also take a week on hardware of the day (a single high end GPU).

One factor for stalling wall-clock times is psychological. Thirst for knowledge is unquenchable so researchers will keep increasing computational load until it takes too long. Some engineers at Google use 10 PFlop TPU pods to train exotic models on MNIST. Clearly, a creative person will nd a way to exhaust any amount of computation available to them.

However, there are also fundamental factors that are keeping training times high. Consider the following code to train a neural net:

```
for i in range([redacted]):
 params-=gradient(params)
 if objective(params)<0.01:
 break
```

Even though the number of available transistors has been growing exponentially, if you have to do 3 million iterations in a sequence, at some point extra transistors stop helping and you are limited by the single thread performance.

Single thread performance has not been doing great. Consider this chart:

Sign in to Follow Follow this author for updates

Sandra Sanders FOLLOW
Posted 1 hour ago

OTOSOMOn: The Root Cause of Slow Neural Net Training

Neural network training can be accelerated when parameters vary in a fixed way, a phenomenon we call Rhythm Training Optimization (RTO). We introduce OTOSOMOn, a lightweight optimizer that detects such periodic patterns from the loss trajectory and applies a cosine-shaped correction to the Adam update. On a 12-layer GPT-2 trained on WikiText-103, OTOSOMOn reaches the Adam baseline's final loss in 31% fewer steps at 0.4% wall-clock overhead.

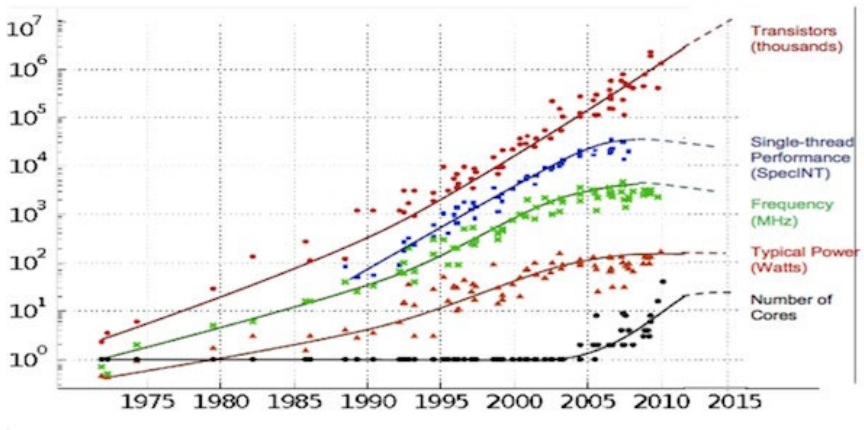
Cross-posted to the Mirror Sanctorium (Spin-aware) where it started as a series of notes on the idea. Thirty-minute Kaggle version here. Code and full ablations below. Comments welcome, especially from people who've seen a phase-aware optimizer before.

Now Rhythm Training will look familiar to anyone who's read Rhythm Mixer. The same idea, that a signal can be decomposed into a slow trend and a fast residual, applies here to the loss trajectory. Rotate adjacent pairs of weight gradients by a learned angle, mix the rotated and original gradients, and the optimizer sees a smoother signal. Rhythm Mixer does this in a hidden state; OTOSOMOn does it in a scalar loss curve and a single learned angle.

Rhythm Mixer

Rhythm Mixer

Rhythm Mixer



C Moore, Data Processing in Exa-Scale-Class Computer Systems, Salishan, April 2011

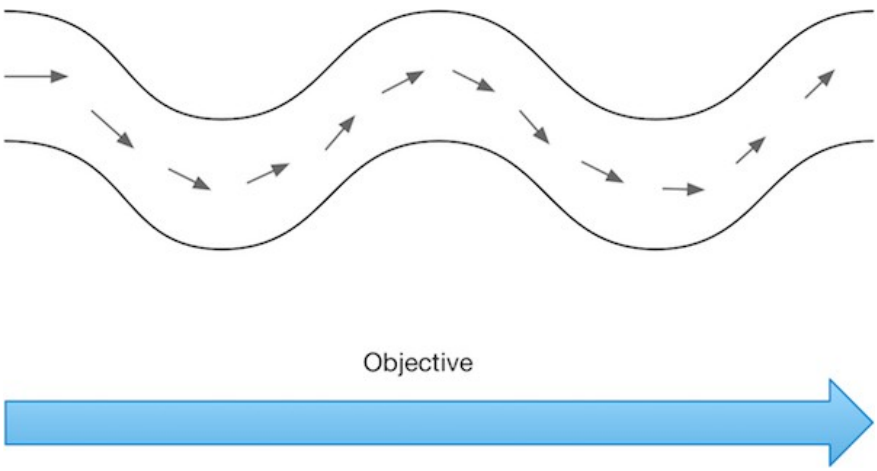
Serial performance hit a peak around 2007 and has been agging since. In practice I found that training some Atari RL models was faster on my 3 year old laptop than on the latest Intel Xeon chip.

To get around poor single thread performance we need to gure out ways to reduce the number of iterations in the training loop.

The number of iterations has been kept high because of the following 3 OTODSCINOs (Obstacles TO Decreasing Serial Complexity In Nonlinear Optimization)

OTODSCINO 1: amount of non-linearity

Consider the following landscape of a non-linear optimization problem:



Because gradient descent works on local information, there's a certain number of steps it needs to take until it can even "see" the minimum.

Non-linearity can be characterized by interactions between components of the objective function, and as you add more layers to a

Fig. 1 Moore, Dennard, Power Wall, Memory Wall, ILP Wall, and Dark Silicon. Adapted from Rupp, K. (2018). 40 years of microprocessor trend data. https://www.karlrupp.net/2018/06/40-years-of-microprocessor-trend-data/. Accessed 20 Sept. 2020.

Dennard performance law, power crossover (Wall), and Dark Silicon. Adapted from Rupp, K. (2018). 40 years of microprocessor trend data. https://www.karlrupp.net/2018/06/40-years-of-microprocessor-trend-data/. Accessed 20 Sept. 2020.

Moore's Law (Moore's Law) and Dennard performance law (Dennard Scaling) are well known. Power Wall, Memory Wall, ILP Wall, and Dark Silicon

以上内容仅为本文档的试下载部分，为可阅读页数的一半内容。如要下载或阅读全文，请访问：https://d.book118.com/866024135000010031