Have a Larger Cake and Eat it Faster Too: A Guideline to Train Larger Models Faster

Newsha Ardalani
Baidu Silicon Valley AI Lab
ardalaniebaidu.com

Joel Hestness
Baidu Silicon Valley AI Lab
hestness@baidu.com

Gregory Diamos
Baidu Silicon Valley AI Lab
gregdiamos@baidu.com

ABSTRACT
With the increasing prevalence of deep neural networks and their growing demand for more powerful hardware, understanding the interplay of model architecture parameters, hardware architecture parameters, and model and data parallelism on overall model performance (training time and accuracy) becomes ever more important in order to design next-generation deep learning (DL) hardware. To aid such understanding, this work studies the effect of scaling model size on overall performance, and debunks a long-held belief that larger models must take longer to train. We first break the total training time into number of steps and time/step. We analytically model the training time per step and empirically study the number of steps to convergence. We observe that larger models take fewer steps to reach to minimum validation loss (halting point). Therefore, the burden is on the hardware community to improve hardware design such that the growth in training time/step would be slower than the decrease in the number of steps as model size scales. If successful, larger models will converge faster, and therefore we can have a larger cake and eat it faster too.

1 INTRODUCTION
The recent success in deep learning has been a driving force in the hardware industry for designing more powerful, energy-efficient GPUs [3] and ASICs [1, 2, 7–11, 15] with special support for deep learning. Recent studies suggest that deep learning accuracy scales with training data [4, 12, 14]. Therefore, there is an expectation that model size and as a result computation demands to grow rapidly with dataset size. In this paper, we make an observation that suggests the growth in model size and computation demand is expected to be even faster than the growth in dataset size. We observe that larger models take fewer steps to reach to similar levels of accuracy. Figure 1 shows this pattern for character-level language models trained on 0.01% of the Billion Word dataset [6], with batch size of 128, and using Adam optimizer with an initial learning rate of 0.001. As depicted, larger models take fewer steps to converge, while training time per step grows with model size. On this particular design, we also get better overall training time on the existing GPU hardware (Nvidia Maxwell). This overall trend of total training time dropping by model size depends on the underlying hardware, model architecture and algorithmic properties of the implementation. Note here that the number of steps to convergence is only a function of model architecture, while the training time per step is a function of hardware architecture and the efficiency of the implementation. If we can design our hardware and/or algorithms such that the training time per step grows with a lower pace than the decrease in the number of steps, we can practically train larger models faster than smaller ones.

2 METHODOLOGY
We define time-to-convergence as the number of steps to reach within 1% of the minimum validation loss. We use early stopping (at minimum validation loss) to control overfitting. We increase the model size by increasing the number of nodes per layer, while keeping all the other architecture parameters the same (same learning rate, same batch size, etc.). We evaluate our finding on three established DL models: character-level language model (LM), word-level LM and speech recognition.
Word LM: We implement LSTM-based word LMs as described in [16]. We restrict the vocabulary to the top 10,000 most frequent words in the Billion Word Dataset [6]. The networks are 2-layer LSTMs, with sequence length of 80, the same number of hidden nodes in each layer.

Character LM: We implement char-LM using Recurrent Highway Networks (RHNs) [21]. Specifically, we train a 1-layer, depth 10 RHN, sequence length 150.

Speech recognition: We train models similar to Deep Speech 2 (DS2) [5] which consist of two convolution layers followed by four bidirectional LSTM recurrent layers.

3 RESULTS AND ANALYSIS

In this Section, we study the trade-off between convergence time, accuracy and model size. Our preliminary results suggest that there is a potential to increase model size to improve training time without hurting accuracy.

3.1 Number of Steps to Minimum Validation Loss

Figure 2 shows the number of steps to convergence for the character, word and speech model. X-axis and Y-axis are in logarithmic scale. We increase the model size on X-axis by increasing the width of each layer. Different lines represent different dataset sizes (percentage of a full dataset). As shown, the number of steps to convergence declines with model size and grows with dataset size. In general, the number of steps to convergence can be approximated by \( \frac{D^{k_1}}{M^{k_2}} \), where \( D \) is the size of training set, \( M \) is model size, and \( k_1 \) and \( k_2 \) are controlled by dataset characteristics and model architecture parameters, respectively. We also observe that this power law relationship eventually plateaus, i.e there exists a model size beyond which improving model size does not improve training time.

3.2 Accuracy

Although theoretical results suggest over-parametrizing models may lead to worse generalization error, our empirical results show that the change in accuracy is insignificant (Figure 3). As shown, loss remains almost the same after it reaches to its best performance.

3.3 Sensitivity Analysis

We also study the effect of change in depth and learning rate on the number of steps and the power-law relationship. We observe that the number of steps to minimum validation loss also declines as we sweep the number of layers from 1 to 128. We also observe that adaptively changing learning rate (using Adam optimizer) improves the number of steps in two ways: It shifts down the power-law curve, and also makes it steeper.

3.4 System/Hardware Implications

Moving forward, we expect an increasing demand for larger models, not only because they are more accurate with more data but also because they train faster. Therefore, system/hardware designers need to focus on larger models by providing support for model parallelism through improving inter-device bandwidth and computational throughput per device.
REFERENCES