Deep learning’s recent history has been one of achievement: from triumphing over humans in the game of Go to world-leading performance in image classification, voice recognition, translation, and other tasks. But this progress has come with a voracious appetite for computing power. This article catalogs the extent of this dependency, showing that progress across a wide variety of applications is strongly reliant on increases in computing power. Extrapolating forward this reliance reveals that progress along current lines is rapidly becoming economically, technically, and environmentally unsustainable. Thus, continued progress in these applications will require dramatically more computationally-efficient methods, which will either have to come from changes to deep learning or from moving to other machine learning methods.

**KEYWORDS**
Deep Learning, Computing Power, Computational Burden, Scaling, Machine Learning

**ABSTRACT**

To understand scaling in deep learning, we analyze 1,527 research papers found in the arXiv pre-print repository, as well as other sources, in the domains of image classification, object detection, question answering, named entity recognition, machine translation, speech recognition, face detection, image generation, and pose estimation. We find that computational requirements have escalated dramatically and that increases in computing power have been central to performance improvements.

This finding has important public policy implications: if current trends continue, the growing “computational burden” of deep learning will rapidly become technically and economically prohibitive. Such a rapid escalation in computing needed also implies alarming growth in deep learning’s environmental cost[26]. Faced with these challenges, the machine learning community will be pushed to either dramatically increase the efficiency of deep learning to move to more computationally-efficient machine learning techniques.

To understand why deep learning is so computationally expensive, we analyze its statistical and computational scaling in theory. We show that deep learning is not computationally expensive by accident, but by design. The same flexibility that makes it excellent at modeling diverse phenomena and outperforming expert models also makes it dramatically more computationally expensive. Despite this, we find that the actual computational burden of deep learning models is scaling more rapidly than (known) lower bounds from theory, suggesting that substantial improvements might be possible.

It would not be a historical anomaly for deep learning to become computationally constrained. Even at the creation of the first neural networks by Frank Rosenblatt, performance was limited by the available computation. In the past decade, these computational constraints have been relaxed due to speed-ups from moving to specialized hardware and a willingness to invest more resources to improve performance. But, as we show, the computational needs of deep learning scale so rapidly that they will quickly become constraining again.

**1 INTRODUCTION**

In this article, we present a comprehensive meta-analysis of how deep learning progress depends on growing computational power and use this to understand not just how particular models scale, but how the field as a whole does. Our analysis differs from previous ones in that we are (i) more precise in the models we compare than are many high-level historical analyses, which allows us to better understand how performance changes as computing scales up, and (ii) better able to account for innovation in the field than estimates where researchers have tested scaling by varying the compute used in training their own models.

To understand scaling in deep learning, we analyze 1,527 research papers found in the arXiv pre-print repository, as well as ACM Reference Format:
Neil Thompson, Kristjan Greenewald, Keeheon Lee, and Gabriel F. Manso. 2021. The Computational Limits of Deep Learning. In LIMITS ’21: Workshop on Computing within Limits, June 14–15, 2021. ACM, New York, NY, USA, 16 pages.
to believe that deep learning is intrinsically highly reliant on computing power. This arises from the role of overparameterization and how this scales as additional training data are used to improve performance.

It has been proven that there are significant benefits to having a neural network containing more model parameters than data points available for training, that is, by overparameterizing it [108]. Classically this would lead to overfitting, but stochastic gradient-based optimization methods provide a regularizing effect due to early stopping [9, 92]², moving the neural networks into an interpolation regime, where the training data is fit almost exactly while still maintaining reasonable predictions on intermediate points [10, 11]. An example of large-scale overparameterization is the current state-of-the-art image classification system, CoCa, which has 2.1B parameters for ImageNet’s 1.2M data points [128].

The challenge of overparameterization is that the number of deep learning parameters must grow as the number of data points grows. Since the cost of training a deep learning model scales with the product of the number of parameters with the number of data points, this implies that computational requirements grow at least the square of the number of data points in the overparameterized setting. This quadratic scaling, however, is an underestimate of how fast deep learning networks must grow to improve performance, because a linear improvement in performance generally requires a faster-than-linear increase in the amount of training data.

For instance, statistical learning theory tells us that, in general, root mean squared prediction error can at most drop as $1/\sqrt{n}$ (where $n$ is the number of data points) [72]. These rates indicate that at least a quadratic increase in data points would be needed to improve performance. So, combining the computational overhead from overparameterization and the data requirements for statistical learning yields a back-of-the-envelope estimate that the computation required to train an overparameterized model should grow at least as a fourth-order polynomial with respect to performance,³ i.e. Computation = $\Omega(\text{Performance}^4)$. This is, of course, just a lower bound. Due to the complexity of deep learning, performance could be considerably worse, perhaps even requiring exponential increases in computing power as has been seen in other tasks like weather prediction [113].

While the bound above was derived for root mean squared error, the result is more general, applying to the large class of performance metrics that converge as $1/\sqrt{n}$. For example, this includes any smooth loss function (or error metric) that is computed by averaging over data points, as [42] showed.⁴ In particular, this result applies to most popular neural network training losses, including the cross entropy loss.

The relationship between model parameters, data, and computational requirements in deep learning can be illustrated by analogy in the setting of linear regression, where the statistical learning theory is better developed (and, which is equivalent to a 1-layer neural network with linear activations). Under the usual conditions, the root mean squared prediction error from the ordinary least-squares (OLS) estimator scales as $O\left(\frac{d}{n}\right)$, where $d$ is the number of model parameters and $n$ the number of observations. Under these conditions, and assuming stochastic gradient descent is used for estimation, learning a model with 1,000× as many parameters would take 1,000,000× longer (due to the necessary increase in $n$ to preserve the same RMSE). Regularization (either explicit regularization or the implicit regularization created by state of the art training of neural networks) can help. For instance, the lasso estimator [117], which performs an explicit regularization, improves root mean squared error scaling to $O\left(\frac{s \log d}{n}\right)$ where $s$ is the number of nonzero coefficients in the true model [82]. We make an analogy between the role of regularized lasso estimation in linear regression to the role of deep learning in nonlinear problems, since neural networks have been shown to be implicitly regularized [9, 92].

Even with regularization, however, theory tells us that the computing power needed for improved performance still grows incredibly rapidly. For example, the computational power needed to run a highly flexible (flexibility is sometimes also called “effective model complexity” [84]) lasso model with $d = 1,000s$ parameters, is about $1,000s \times$ that for running a lasso model with just the true number of parameters, $d = s$. Figure 1 (a) generalizes this, showing the increase in computation needed as the effective model complexity ($d/s$) increases [84].

---

²This is often called implicit regularization, since there is no explicit regularization term in the model.
³Here, performance is $1/\text{RMSE}$.
⁴See [42] for precise assumptions.

---

Figure 1: a) Computational burden of running regularized models (lasso) as the effective model complexity [84], $d/s$, is increased (where $d$ is the number of parameters in the estimated model and $s$ is the number in the true model). b) Implications of flexibility for machine learning model performance.
As Figure 1 (a) shows, there is an enormous computational price that has to be paid for building models with many parameters, even when regularization is used. But while the price of including so many parameters may be high, it also offers flexibility for the model. In contrast, smaller models may be more efficient, but if they do not include the parameters that matter for the answer (in the example above, some of the $s$ coefficients), this would imply lower RMSE values being unachievable for any amount of computation. In other words, the performance of that model will eventually plateau at a low level as available computation/number of samples increase, since it lacks important predictive features. In contrast, the model with many parameters will eventually achieve a high level of performance, but at the cost of more data and computation.

Thus, we arrive at the central tradeoff between traditional machine learning methods (like regression) that use small numbers of parameters and deep learning methods that use enormous numbers of parameters. The more parameters that one adds to a model the greater the flexibility and hence potential for better performance. Indeed, it has been shown that sufficiently large neural networks are universal function approximators [55], hence in theory, any desired performance level can be achieved by making the model large enough and including enough training data. But these additional parameters also make the model more expensive to train (even before any needed increase in amount of training data) and can make it do less well when the amount of data (or computation) is not large enough. Figure 1 (b), our adaptation of a graph attributed to Andrew Ng [67], summarizes this.

3 DEEP LEARNING’S COMPUTATIONAL REQUIREMENTS IN PRACTICE

3.1 Past

Even in their early days, it was clear that computational requirements limited what neural networks could achieve. In 1960, when Frank Rosenblatt wrote about a 3-layer neural network, there were hopes that it had “gone a long way toward demonstrating the feasibility of a perceptron as a pattern-recognizing device.” But, as Rosenblatt already recognized “as the number of connections in the network increases, however, the burden on a conventional digital computer soon becomes excessive” [101]. Later that decade, in 1969, Minsky and Papert explained the limits of 3-layer networks, including the inability to learn the simple XOR function. At the same time, however, they noted a potential solution: “the experimenters discovered an interesting way to get around this difficulty by introducing longer chains of intermediate units” (that is, by building deeper neural networks) [83]. Despite this potential workaround, much of the academic work in this area was abandoned because there simply wasn’t enough computing power available at the time. As Léon Bottou later wrote “the most obvious application of the perceptron, computer vision, demands computing capabilities that far exceed what could be achieved with the technology of the 1960s” [83].

In the decades that followed, improvements in computer hardware provided, by one measure, an approximately 50,000× improvement in performance [48] and the largest neural networks being used grew their computational requirements proportionally, as shown in Figure 2(a). Since the growth in computing power per dollar closely mimicked the growth in computing power per chip [116], this meant that the economic cost of running such models was largely stable over time. Despite this large increase, deep learning models in 2009 remained “too slow for large-scale applications, forcing researchers to focus on smaller-scale models or to use fewer training examples” [96] The turning point seems to have been when deep learning was ported to GPUs, initially yielding a 5–15× speed-up [96] which by 2012 had grown to more than 35× [85], and which led to the important victory of Alexnet at the 2012 ImageNet competition [66]. But image classification was just the first of these benchmarks to fall. Shortly thereafter, deep learning systems also won at object detection [36, 105, 124], named-entity recognition [70], machine translation [60, 76, 129], question answering[56], and speech recognition [3, 45].

![Figure 2: Computing power used in: (a) the largest deep learning models in different year (across all applications) [4] as compared with the growth in hardware performance from improving processors[25], as analyzed by [48] and [69]. (b) image classification models tested on the ImageNet benchmark with computation normalized to the 2012 AlexNet model [66].](image)

The introduction of GPU-based (and later ASIC-based) deep learning led to widespread adoption of these systems. But the amount of computing power used in the largest cutting-edge systems grew even faster, at approximately 10× per year from 2012 to 2019 [4]. This rapid increase in computing burden far outpaced

---

1The ImageNet Large Scale Visual Recognition Challenge (ILSVRC) released a large visual database to evaluate algorithms for classifying and detecting objects and scenes every year since 2010 [29, 103].

2The range of hardware performance values indicates the difference between SPECInt values for the lowest core counts and SPECIntRate values for the highest core counts.
the ≈ 35× total improvement from moving to GPUs, the mea-ger improvements from the last vestiges of Moore’s Law [116], or the improvements in neural network training efficiency [49]. Instead, much of the increase came from a much-less-economically-attractive source: running models for more time on more machines. For example, in 2012 AlexNet trained using 2 GPUs for 5-6 days [66], in 2017 ResNeXt-101 [126] was trained with 8 GPUs for over 10 days, and in 2020 Meta Pseudo Labels (Efficient-Net-L2) was trained with 2048 TPU cores for 11 days [91]. Another extreme example is the machine translation system, “Evolved Transformer”, which used more than 2 million GPU hours and cost billions of dollars to run [107, 125].7 Scaling deep learning computation by scaling up hardware hours or number of chips is problematic in the long-term because it implies that costs scale at roughly the same rate as increases in computing power [4], which (as we show) will quickly make it unsustainable.

3.2 Present
To examine deep learning’s dependence on computation, we con-ducted an extensive review of all the research papers we could find that covered the domains of image classification (ImageNet, CIFAR-10, CIFAR-100), object detection (MS COCO), question answering (SQuAD 1.1), named-entity recognition (COLLNN 2003), machine translation (WMT 2014), speech recognition (ASR SWB Hub500), face detection (WIDER Face Hard), image generation (CIFAR-10), and pose estimation (MPII Human Pose) 8. We limit our analysis to benchmarks with exact right and wrong answers, so that error rates can be precise. For this reason, our analysis excludes generative models in language and audio. Using established benchmarks to measure progress in these areas is important because it ensures a common baseline for comparison (which is a significant problem in parts of the deep learning literature [23]). We source these deep learning papers from the arXiv repository as well as other benchmark sources.

In total, we gathered 1,527 deep learning papers, for which we did a detailed manual review for their performance and computa-tion burden data. Unfortunately, as is well known in deep learning, few papers report the details of the amount of computation they used [31, 35], reflecting the field’s historical focus on accuracy at the expense of other measures of progress [79]. Most papers do not report any details of the computational requirements for their mod-els, and in many others only limited computational information is provided. Table 1 summarizes our data. Because there is insufficient computational data for other benchmarks, we limit our analysis to ImageNet, MC COCO, SQuAD 1.1, COLLNN 2003, and WMT 2014 (EN-FR) and WMT 2014 (EN-DE).

Reflecting the computational information available in these pa pers, we do separate analyses for two measures of the computational burden: (1) Network Operations [4, 104], the number of floating point operations computed in the network 10, and (2) Hardware burden, the computational capacity of the hardware used to train the model.

\[
\sum_{i \in \text{pre-training, training, fine-tuning}} \text{Epochs}_i \cdot \text{FlopsPerNetworkPass}_i \cdot \text{NetworkPassesPerEpoch}_i \quad (1)
\]

\[
\sum_{i \in \text{pre-training, training, fine-tuning}} \text{Processors}_i \cdot \text{ComputationRate}_i \cdot \text{Time}_i \quad (2)
\]

We illustrate our analysis first in the area with the most data and longest history: image classification. The relevant performance metric here is the error rate for classification. As discussed in the previous section, we should expect computation to scale at least as a high order (e.g. 4th order) polynomial versus performance, which we estimate via the equation: \(\log(1/\text{error}) = \alpha + \beta \cdot \log(\text{computation})\). Figure 2 (b) shows the fall in the image classification error rate for the ImageNet dataset and its correlation to the computation used in these models. Each data point reflects a particular deep learning model from the literature. Because this is plotted on a log-log scale, a straight line indicates a polynomial growth in computing per unit of performance – that is, a power law. In particular, a polynomial relationship between computation and performance of the form \(\text{Computation} = \text{Performance}^\alpha\) yields a slope of \(-\frac{1}{\alpha}\) in our plots. Thus, our estimated slope coefficient of \(-0.084\) (p-value < 0.01) indicates that computation used for ImageNet scales as \(O(\text{Performance}^{12.9})\). Concretely, this means that every halving of the remaining error on this problem requires \(\approx 2^{12.9} > 4,000x\) as much computation.

Taking into account the standard error on this estimate yields a 95% confidence interval for scaling between \(O(\text{Performance}^{10.4})\) and \(O(\text{Performance}^{13.9})\), i.e. between \(\approx 1,350x\) and \(\approx 15,290x\) as much computation to halve the error. Not only is computational power a highly statistically significant predictor of performance, but it also has substantial explanatory power, explaining 72% of the variance in ImageNet performance. Table 2a, specification 1, reports

\[\text{Top 1 score refers to the accuracy of a classification model, indicating the percentage of correct predictions where the model’s top-ranked prediction matches the true label. BOX AP measures the precision and recall of a model by evaluating how well it localizes and classifies objects within bounding boxes. Percentage correct represents the accuracy of a model or classifier, indicating the percentage of correct predictions out of the total number of predictions made. AP is a metric used to evaluate the performance of models in information retrieval or object detection tasks. PCKH-0.5 stands for Percentage of Correct Keypoints and is often used to evaluate human pose estimation models. The F1 score is a measure of a model’s accuracy, combining precision and recall. BLEU is a metric commonly used to evaluate the quality of machine translation. Percentage error: This general term refers to the percentage difference between an observed or predicted value and the true value.}\]

\[\text{Note that, one multiply – add operation is composed of two arithmetic operations (the product of two numbers and the addition of this product to an accumulator). Therefore, for cases where authors report the number of forward-pass operations in multiply – add operations, we use a conversion factor equals to } \frac{1}{2} \text{ to convert this value to flops [4]}.\]
Table 1: Deep learning benchmark data

| Domain          | Task                        | Benchmark          | Evaluation Criteria | State-of-the-Art               | # Papers Found | # Models | Hardware Burden | Network Operations | Hardware Burden & Network Operations |
|-----------------|-----------------------------|--------------------|---------------------|------------------------------|----------------|----------|-----------------|---------------------|-------------------------------------|
| Computer Vision | Image Classification        | ImageNet           | Top-1 score         | CoCa (Top 1: 91.40)          | 127            | 12       | 13              | 7                   | 0                     |
|                 | Object Detection            | MS COCO            | BOX AP              | DMD Siam-L (Top 1: 85.55)    | 152            | 13       | 7               | 0                   | 0                     |
|                 | Face Detection              | WIDER Face (Hard)  | AP                  | TestFace (AP 95.04)          | 35             | 35       | 4               | 0                   | 0                     |
|                 | Image Generation            | CIFAR-10           | Percentage Correct | DNN-L2-SAM (Percentage Correct: 96.50) | 35             | 35       | 4               | 0                   | 0                     |
|                 | Face Estimation             | MPII Human Pose    | PCKh-0.5            | Soft-gated Skip Connections (PCKh-0.5) | 35             | 35       | 4               | 0                   | 0                     |
| Natural Language Processing | Question Answering | SQUAD 1.1 | EM                  | Arabic (EM: 96.6)             | 127            | 12       | 0               | 0                   | 0                     |
|                 | Named Entity Recognition    | CoNLL 2003         | F1-score            | ACE + document-context (F1-score: 94.36) | 127            | 12       | 0               | 0                   | 0                     |
|                 | Machine Translation         | WMT 2014 (EN-FR)   | BLEU                | Transformer - BT-ASR-124.22 (BLEU: 35.40) | 127            | 12       | 0               | 0                   | 0                     |
|                 | Machine Translation         | WMT 2014 (EN-DE)   | BLEU                | Transformer Cycle Rev (BLEU: 35.14) | 127            | 12       | 0               | 0                   | 0                     |
|                 | Speech                       | Switchboard + Hub 500 | Percentage Error   | IBM LSTM + Conformer encoder-decoder (Percentage Error: 4.9) | 127            | 12       | 0               | 0                   | 0                     |
| **Total**       |                             |                    |                     |                              | 1527           | 114      | 86              | 11                  | 11                    |

this regression result, alongside a series of alternate specifications that test the robustness of our finding.

It is known that there have been substantial improvements in the efficiency of deep learning training [49]. In specification (2) we introduce a time trend to proxy for these algorithmic changes and find that it increases the explanatory power of the model by 11.1%. As in previous work, we find clear evidence of efficiency gains: 3 years of algorithm improvement is equivalent to an increase in computing power of 10x. But, even after accounting for algorithm improvement, we continue to observe a power law between computing power and performance that this implies that every year deep learning system designers are both taking advantage of year-over-year algorithm improvement and also scaling their models according to the performance trade-offs that we have identified. And thus, while it is encouraging that algorithmic efficiency has improved, it does not alleviate the inflation in computational burden that we observe.11

In specification (3) we test a functional form where computation scales exponentially with performance, rather than polynomially. That form also results in a highly statistically significant reliance on computing power, but has less explanatory power, so we retain specification 1 as our preferred form.

In specification (4) we analyze whether focusing on the best-performing models would yield a different result that looking at all models. Using a quantile regression, we estimate the scaling at the threshold of the 10% most efficient models. This estimation shows a similar dependence of performance on computation for these cutting-edge models: \( O(Performance^{11.9}) \).

Thus, in ImageNet, where we have the most data, our baseline analysis and robustness checks paint a strongly coherent picture: deep learning performance improvement is strongly dependent on rapid scale-up of computing power, whether or not algorithmic improvement is accounted for and whether or not one looks at all models or only cutting-edge ones.

Despite the efforts of machine learning conferences to encourage more thorough reporting of experimental details (e.g. the reproducibility checklists of ICML [59] and NeurIPS [93, 106]), few papers in the other benchmark areas provide sufficient information to analyze the computational burden via network operations. More widely reported, however, are the components needed to calculate an alternative metric: hardware burden. This also estimates the computation needed, but is less precise since it depends on hardware implementation efficiency.

Table 2b and Figure 3 shows progress in the areas of image classification, object detection, question answering, and named entity recognition. We find highly-statistically significant slopes and strong explanatory power (\( R^2 \) between 42% and 87%) for all benchmarks. Interpreting the coefficients for the five remaining benchmarks shows a slightly lower polynomial dependence for ImageNet when calculated using this method (\( \approx 11.0 \)), and a dependence of 10.5 and 16.4 for question answering and object detection respectively. Named-entity recognition shows large increases in hardware burden with relatively small improvements in outcomes, implying dependencies of around 37.2, although this model explains only 42% of the variance so this result should be interpreted as preliminary. In machine translation we also observe a correlation

11In 5 we revisit this issue of efficiency improvement as part of a larger discussion about the economic and environmental implications of this rapid rise in the computation needed.
### Table 2: Regression Analysis of how Deep Learning Performance depends on Computing Power Growth

| (a) Network Operations | (1) | (2) | (3) | (4) |
|------------------------|-----|-----|-----|-----|
| log₁₀(Top 1 error)    |     |     |     |     |
| log₁₀(Top 1 error)    |     |     |     |     |
| Image Classification   |     |     |     |     |
| Image Classification   |     |     |     |     |
| OLS Regression         |     |     |     |     |
| OLS Regression         |     |     |     |     |
| Intercept              |     |     |     |     |
| Interception           |     |     |     |     |
| Observations           | 77  | 77  | 77  | 77  |
| R² / pseudo R²         | 0.720 | 0.831 | 0.628 | 0.579 |
| Adjusted R²            | 0.717 | 0.826 | 0.623 |     |
| Residual Std. Error    | 0.050 (df = 75) | 0.059 (df = 74) | 0.025 (df = 75) |     |
| F Statistic            | 193.224 (df = 1; 75) | 181.400 (df = 2; 74) | 126.667 (df = 1; 75) |     |
| Implied Polynomial Scaling Factor | 12.0 | 15.7 |     | 11.9 |
| 95% Confidence Interval | 10.4 – 13.9 | 13.3 – 18.9 |     | 10.6 – 13.3 |

| (b) Hardware Burden    | (5) | (6) | (7) | (8) |
|------------------------|-----|-----|-----|-----|
| log₁₀(TOP 1)          |     |     |     |     |
| log₁₀(BOX AP)         |     |     |     |     |
| log₁₀(EM)             |     |     |     |     |
| log₁₀(F1 score)       |     |     |     |     |
| Image Classification   |     |     |     |     |
| Object Detection (MS COCO) |     |     |     |     |
| Question Answering (SQuAD 1.1) |     |     |     |     |
| Named Entity Recognition (CoNLL 2003) |     |     |     |     |
| Intercept              |     |     |     |     |
| Interception           |     |     |     |     |
| Observations           | 102 | 20  | 12  | 12  |
| R²                     | 0.694 | 0.814 | 0.872 | 0.417 |
| Adjusted R²            | 0.690 | 0.804 | 0.859 | 0.359 |
| Residual Std. Error    | 0.053 (df = 100) | 0.050 (df = 110) | 0.080 (df = 100) | 0.056 (df = 110) |
| F Statistic            | 226.324 (df = 1; 100) | 78.916 (df = 1; 110) | 68.622 (df = 1; 110) | 7.148 (df = 1; 110) |
| Implied Polynomial Scaling Factor | 11.0 | 16.4 | 10.5 | 57.2 |
| 95% Confidence Interval | 9.7 – 12.7 | 13.3 – 21.3 | 8.3 – 14.3 | 20.4 – 250 |

Note: Network Operations is normalized relative to the 2012 AlexNet model.

between compute and performance, but there has not been enough variation in computing power for us to reliably estimate the slope. In the online supplementary materials, we also test other functional forms for estimating hardware burden [115]. As with the network operations analysis, we find that polynomial models best explain the data, but that exponential models are also plausible. Collectively, our results make it clear that, across many areas of deep learning, progress in training better models has depended on large increases in the amount of computing power being used. A dependence on computing power for improved performance is not unique to deep learning, but has also been seen in other areas such as weather prediction, computer chess, computer Go, and oil exploration [113]. In those areas there has been enormous growth in the cost of systems, with many cutting-edge models now requiring some of the largest computer systems in the world [41]. This could well be deep learning’s fate if current trends continue.

### 3.3 Future

In this section, we extrapolate the estimates from each domain to understand the projected computational power needed to train models to hit various benchmark performance levels. To make these
targets tangible, we present them not only in terms of the computational power required, but also in terms of the economic and environmental cost of training such models on current hardware.\textsuperscript{12}

\textsuperscript{12}Economic and environmental costs are measured using the methodology provided by [110] for V100 GPU training. We confirm pricing as of 2022 based on [1] and use updated carbon emissions estimates from [32] (for which we take the geometric mean). These carbon estimates are similar to those provided by [86, 88].

These projections reinforce the growing concern that deep learning’s current trajectory will have important negative effects [40, 79]. In this analysis, we focus on the training costs of these models but in section 5 we also discuss the deployment cost of these models. Because the polynomial and exponential functional forms explored...
in previous section have roughly equivalent statistical fits — but quite different extrapolations — we report both in table 3.

We do not anticipate that the computational requirements implied by the targets in Figure 3 will be hit. The hardware, environmental, and monetary costs would be prohibitive and enormous effort is going into finding ways to improving scaling performance to avoid these outcomes (as we discuss in the next section). Nevertheless, these projections do provide a scale for the efficiency improvements that would be needed to hit these performance targets. For example, even in the more-optimistic model, an additional 650× more computing would be needed to get to an error rate of 5% for ImageNet. Hitting this in an economical way will require more efficient hardware, more efficient algorithms, or other improvements such that the net impact is at least this large. Figure 4 (a) shows the large effects that improving the polynomial scaling performance would have on projections and how well these agree with current data.

The rapid escalation in computing needs in Table 3 also makes a stronger statement: without substantial efficiency improvements, it will not be possible for deep learning to hit these benchmarks. The environmental impacts are also troubling. To put these into perspective, they imply that training an ImageNet model to an accuracy of 5% would produce almost as much carbon as the city of Boulder (CO) in a month, while an accuracy of 3% would require as much carbon as New York City emits in a month.

To avoid these outcomes, fundamental rearchitecting is needed to lower the computational intensity so that the scaling of these problems becomes less onerous - which we discuss this in detail in section 5. Efforts will also be needed to minimize these carbon emissions, as we describe in section 6, but first we consider how our findings differ from other scaling studies that have been done on deep learning.

4 COMPARISON TO OTHER SCALING STUDIES

The key question for the future of deep learning is how performance scales up, that is, how much performance for the field improves as computing power increases. This article addresses this question differently from the growing body of work on deep learning scaling. In studying the performance of the whole field, rather than just a class of models, we are tracking not just mathematical scaling of models but also the pace of innovation as researchers find better ways to harness computing power, which Rich Sutton has argued is “the only thing that matters in the long run.” [111]. By measuring the average progress of the field (or in Table 2 specification 4, just the state-of-the-art) we capture cross-researcher, cross-model differences that other scaling papers miss. Not surprisingly, this results in different estimates for scaling performance.

Table 4 summarizes how our analysis compares to other papers in this field. Like most of the papers in this field, we focus on the training costs needed. Of these, other papers generally fall into two groups: within-model experiments and across-model historical analyses. Within-model experiments have a specific reference point, the benchmark being analyzed. As a result, they provide an excellent view of how deep learning performance scales as more computation is used. But this analysis has a limited scope — only the model used by the researchers — and thus these analyses cannot make any claims about innovation happening across the field. In contrast, most across-model historical analyses can capture innovation, but lack the specificity of benchmark comparisons and thus cannot articulate the performance benefits of additional computation. Our analyses sits between these, capturing both the specificity of individual benchmarks (and thus the performance vs compute trade-off), but also the breadth of looking across models that allows us to capture innovation in the field. More specifically, our approach provides the following benefits:

First, our analysis has more specificity in its comparison set because we examine performance within particular deep learning benchmarks rather than across domains. This contrasts, for example, with [4, 73, 104] that aggregate analysis across different deep learning benchmarks and domains, and therefore do not distinguish between increases in computational burden within particular tasks (e.g. image classification on ImageNet) and the application of deep learning to more computationally-intensive sub-fields (e.g. playing Go). In contrast with those papers, we are able to measure the growth in computational burden that has been needed to get better performance on particular benchmarks.

Second, our analysis tracks the evolution of performance in each field differently than within model studies where authors examine trade-offs by implementing many different model configurations [14, 19, 46, 64, 112, 130]. The weakness of these experimental studies is that the ‘space’ of potential models they explore is limited to what the authors implement themselves — for example [14] only considers particular network architectures and [64] only studies language models that use the Adam optimizer. Therefore these experimental approaches necessarily miss innovations that those authors did not consider or that take deep learning in new directions. These exclusions hinder the ability of those scaling studies to capture progress over time, whereas our across models approach is able to capture the innovations that are missed by these experimental studies.

Third, as we discuss in more detail below, our analysis is more reliable for estimating future progress, i.e., how things scale up, because analyses that estimate scaling by looking at how performance deteriorates with less computing power, i.e., by scaling down, can yield artificially rosy estimates.

In general, one would expect that our analysis would show faster performance gains from increased computing power than other studies because while both capture the improvements from scaling within models, our also accounts for innovation over time that could further improve on this performance. In practice, however, this is not what we see. While some studies do show slower model scaling than we show for the field, others show faster scaling. We hypothesize that this inconsistency arises because of differences in measurement approaches, depending on whether scaling is determined by how performance scales with additional computation versus how it scales with less. These approaches sound like they might be symmetric, but they are not.

Put simply, the ‘scaling up’ approach measures improvements to the state of the art, whereas the ‘scaling down’ approach measures performance deterioration. To see this, we consider each in turn. The scaling up approach measures how performance changes as computing power increases. As shown in Figure 4 (a), scaling up
Table 3: Implications of achieving performance benchmarks on the computation (in flops), carbon emissions (lbs), and economic costs ($USD) from deep learning based on projections from polynomial and exponential models.

| Benchmark | Error Rate | Polynomial | Exponential |
|-----------|------------|------------|-------------|
|           |            | Computation Required (flops) | Environmental Cost (CO₂) | Economic Cost ($USD) | Computation Required (flops) | Environmental Cost (CO₂) | Economic Cost ($USD) |
| Today: 9.00% | 10²¹ | 10⁴ | 10⁸ | 10¹⁴ | 10⁴ | 10⁸ | 10¹⁴ |
| Target 1: 5% | 10²⁶ | 10⁴ | 10⁹ | 10¹⁵ | 10⁵ | 10⁹ | 10¹⁵ |
| Target 2: 1% | 10³³ | 10⁷ | 10¹⁴ | 10⁲⁰ | 10⁵ | 10¹⁴ | 10⁲⁰ |
| Today: 38.7% | 10²² | 10⁵ | 10¹⁴ | 10⁲⁰ | 10⁵ | 10¹⁴ | 10⁲⁰ |
| Target 1: 10% | 10³¹ | 10⁶ | 10¹⁴ | 10⁲² | 10⁶ | 10¹⁴ | 10⁲² |
| Target 2: 1% | 10⁴⁹ | 10⁷ | 10¹⁴ | 10⁲⁴ | 10⁷ | 10¹⁴ | 10⁲⁴ |
| Today: 9.4% | 10²² | 10⁸ | 10¹⁴ | 10⁲⁶ | 10⁸ | 10¹⁴ | 10⁲⁶ |
| Target 1: 2% | 10⁷⁹ | 10¹¹ | 10¹⁵ | 10⁴⁰ | 10¹¹ | 10¹⁵ | 10⁴⁰ |
| Target 2: 1% | 10⁸⁰ | 10¹² | 10¹⁵ | 10⁴² | 10¹² | 10¹⁵ | 10⁴² |
| Today: 5.4% | 10²¹ | 10⁸ | 10¹⁴ | 10⁴⁴ | 10⁸ | 10¹⁴ | 10⁴⁴ |
| Target 1: 2% | 10⁹⁹ | 10¹⁰ | 10¹⁵ | 10⁴⁶ | 10¹⁰ | 10¹⁵ | 10⁴⁶ |
| Target 2: 1% | 10⁹⁰ | 10¹³ | 10¹⁵ | 10⁴⁸ | 10¹³ | 10¹⁵ | 10⁴⁸ |

Table 4: Comparison of articles analyzing deep learning scaling.

| Paper | Calculation | Specificity of Comparison | Scaling Trends |
|-------|-------------|---------------------------|----------------|
| Ours (2020 - updated in 2022) | × | × | × |
| AI and Compute (2018) [4] | × | × | × |
| Compute Trends Across Three Eras of Machine Learning (2022) [74] | × | × | × |
| “AI and Compute” Trend isn’t Predictive of What is Happening (2021) [77] | × | × | × |
| AI and Compute: How Much Longer Can Computing Progress Drive Artificial Intelligence Progress? (2021) [78] | × | × | × |
| Scaling Laws for Neural Language Models (2020) [30] | × | × | × |
| Deep-Benchmark Learning for Image Recognition (2018) [36] | × | × | × |
| Dual-Path Networks (2017) [19] | × | × | × |
| What is the State of Neural Network Pruning? (2020) [36] | × | × | × |
| EfficientReranking Model Scaling for Convolutional Neural Networks (2017) [14] | × | × | × |
| Learning Transferable Architectures for Image Recognition (2017) [14] | × | × | × |
| Scaling Laws for Deep Learning (2015) [36] | × | × | × |
| Deep-Learning Scaling is Predictable, Empirically (2015) [52] | × | × | × |
| Compute and Energy Consumption Trends in Deep Learning Inference (2014) [56] | × | × | × |

clearly measures the progression in the field in an intuitive way: good scaling means that as more computation is used fewer errors are being made and the state of the art is improving. A scaling up approach naturally emerges from observational analyses, like ours, where both performance and computational power are increasing over time.

Observing this same estimate (i.e. slope) has a less clear implication when scaling down a state-of-the-art model to see how it performs with less computation. In this latter case, a steep slope...
"good scaling") simply means that smaller models are farther from the frontier of what is possible. A simple example illustrates how misleading this can be: imagine a state-of-the-art model that, if it used one flop less for training, became useless (i.e. would have an error rate = 100%). Such a hypothetical system would have an enormous change in performance from a small change in computing, so it would seem to scale enormously well. But, in reality, the system simply breaks when it uses less computation. A less extreme example of this seems to be the case with ResNet scaling[46], which nominally shows dramatically better scaling than the field as a whole (i.e., to get a performance improvement requires compute to only grows to the power of 5.6, whereas for the field it grows by 12.2). In practice, however, these models are not competitive for state-of-the-art performance (as shown in Figure 4) and the rapid scaling just seems to indicate that their relative performance deteriorates more quickly as less computation is used.

The example of NasNets provides an alternative way to understand why we hypothesize that models will not scale up. The class of NasNet models reports to have vastly better scaling than the field as a whole. If one projects this reported scaling, it would indicate that a (hypothetical) NasNet using the same amount of computation as one of the largest vision models (XGTF-L24) should achieve an error rate of 6.85%, handily beating the actual state of the art by more than 2%. That we do not see NasNets easily beating state-of-the-art models is strongly suggestive that they do not in fact scale up this well.

The problem of misinterpreting rapidly deteriorating performance as an indication of good scaling is not limited to deep learning. Similar issues arise in studies of parallelism and lead to perverse conclusions including a parallel algorithm that ‘scales well’ being run on 128 cores and being outperformed by a serial algorithm running on a single core [81]. To avoid having deep learning studies fall prey to this same defect, it is important to analyze how models scale up and to compare such performance across models so that deteriorating performance is not misinterpreted as a virtue.

With this conceptual framework, we can consider how our results compare to other scaling studies. Our results notionally agree with the power law growths found by the experimental studies of [64] and [51], and the rapid growth (appearing to be approximately exponential) indicated by the plots in [14]. But we can also be more quantitative, comparing our ImageNet scaling results to those found by others [19, 46, 112, 130]. Figure 4 (b) plots these studies, and ours, on the same graph.13

As already mentioned, two sets of analyses show faster scaling for individual models that we observe for the field: ResNet and NASNet. NASNet in particular achieves cutting-edge performance and shows an impressive scaling of O(Performance5.3). Based on the argument above, however, we would expect that observing scaling faster than the whole field is just an indication that performance deteriorates rapidly for smaller models.

Of the other scaling studies that we consider, perhaps the most interesting is EfficientNet [112]. For most levels of computation, EfficientNets are at, or close to, the frontier of performance. So they do not seem to fall prey to the ‘rapid deterioration equals good scaling’ trap. Moreover, while EfficientNet scales less well than the field as a whole, it is very close (p12.3 versus p12.2) and remains near the state-of-the-art for each level of computing. All of which suggest that it may be harnessing many of the important innovations used across the field.

Thus, because our analysis accounts for innovation and because it is not misled by overly-optimistic scaling down studies, we believe it provides a better long-term view of the evolution of computing power as higher model performance is sought.

While not the focus of our work, it is also important to consider the research being done on scaling laws for generative AI, for example as summarized by [34]. Here, as well, scaling laws are emerging. Also, as we would expect given our discussion above, we see disagreements when different within-model scaling results are reported and conflict (significantly) in their recommendations [54, 118]. This suggests that across-model comparisons like ours will also be useful in such circumstances - although they will also be quite challenging because the generality of these models means there is not a single measure for comparison.

An important distinction between generative language models and our analysis of computer vision is data. Currently the largest language models are already running into challenges finding new data sources[34], which might either limit their ability to scale up

Figure 4: Scaling laws for Deep Learning. (a) schematic representation of scaling up versus scaling down where the slope is the same but the implications are quite different. (b) Comparison of ImageNet scaling estimates between this paper, ResNet [46], DPN [19], NASNet [130], and EfficientNet [112].
further, or push them to grow performance using less efficient ratios of compute and data.

5 LESSEN THE COMPUTATIONAL BURDEN

The economic and environmental burden of hitting the performance benchmarks in Section 3.3 suggest that Deep Learning is facing an important challenge: either find a way to increase performance without increasing computing power, or have performance stagnate as computational requirements become a constraint. In this section, we briefly survey approaches that are being used to address this challenge. As with the rest of the paper, we focus on just the training cost of these models, rather than including the deployment cost since the latter depends on usage and diffusion patterns for which data is not available. Since total costs must necessarily be higher than just training costs, our analysis provides a lower bound on the total computation needed for any given level of performance.

Increasing computing power: Hardware accelerators. For much of the 2010s, moving to more-efficient hardware platforms (and more of them) was a key source of increased computing power [116]. For deep learning, these included mostly GPU and TPU implementations, although it has increasingly also included FPGA and other ASICs. Fundamentally, all of these approaches sacrifice generality of the computing platform for the efficiency of increased specialization.

In recent years, hardware specialization has provided gains in both compute per dollar and compute per watt, for example TPUs improved by approximately 1.5× in compute per dollar from 2017 to 2019[62, 63] and 4.9× in compute per watt from 2017 to 2020 [62, 63, 98–100].

Even with significant gains from specialization so far, it is unclear that specialization will be able to continue to provide such gains in the future since it faces diminishing returns [69]. Other hardware frameworks are being explored as alternatives, including analog hardware with in-memory computation [2, 65], neuromorphic computing [27], optical computing [71], and quantum computing based approaches [121], as well as hybrid approaches [94]. Thus far, however, such attempts have yet to disrupt the GPU/TPU and FPGA/ASIC architectures. Of these, quantum computing is the approach with perhaps the most long-term upside, since it might offer a potential for sustained exponential increases in computing power [21, 39].

Reducing computational complexity: Network Compression and Acceleration. This body of work primarily focuses on taking a trained neural network and sparsifying or otherwise compressing the connections in the network, so that it requires less computation to use it in prediction tasks [20]. This is typically done by using optimization or heuristics such as “pruning” weights [14, 35], quantizing the network [57], or using low-rank compression [122], yielding a network that retains the performance of the original network but requires fewer floating point operations to evaluate. Not all results that have claimed success in this field have really achieved it [58], but those that have achieved gains have not been large enough to mitigate the overall orders-of-magnitude increases of computation in the field (e.g. the work [18] reduces computation by a factor of 2, and [125] reduces it by a factor of 8 on a specific NLP architecture, both without reducing performance significantly). Furthermore, many of these works focus on improving the computational cost of evaluating the deployed network, which is useful, but does not mitigate the training cost, which can itself be prohibitive.

Finding high-performing small deep learning architectures: Neural Architecture Search and Meta Learning. It has become popular to use optimization to find network architectures that are computationally efficient to train while retaining good performance on some class of learning problems, e.g. [90], [15] and [37]. Designers exploit the fact that many datasets are similar and therefore information from previously trained models can be used (meta learning [90] and transfer learning [74]). While often quite successful, the downside is that the current overhead of doing meta learning or neural architecture search is itself computationally intense (since it requires training many models on a wide variety of datasets) [90]. Promisingly, however, the size of this extra overhead cost has been falling [13, 15, 16].

An important limitation to meta learning is the scope of the data that the original model was trained on. For example, for ImageNet, [7] showed that image classification performance depends heavily on image biases (e.g. an object is often photographed at a particular angle with a particular pose), and that without these biases transfer learning performance drops 45%. Even with novel data sets purposefully built to mimic their training data, [97] finds that performance drops 11 – 14%. Hence, while there seems to be a number of promising research directions for making deep learning computation grow at a more attractive rate, they have yet to achieve the orders-of-magnitude improvements needed to allow deep learning progress to continue scaling.

Improved curation of data may also be a way to decrease the computational burden of deep learning models. In particular, models could be trained with better data, rather than more of it, to reduce the computation needed [12]. That said, since data curation can be expensive (e.g. in increased human annotator costs), while higher quality data would preferentially improve computational costs, it could still have an ambiguous effect on total costs.

Another possible approach to evade the rising computational burden of deep learning would be to move to other, perhaps as yet undiscovered or underappreciated types of machine learning. “Expert” models can be much more computationally-efficient, but their performance plateaus if those experts cannot see all the contributing factors that a flexible model might explore. One example where such techniques are already outperforming deep learning models are those where engineering and physics knowledge can be more-directly applied: the recognition of known objects (e.g. vehicles) [47, 119], and those using biologically-inspired methods, e.g. for learning neural controller architectures [68]. The recent development of symbolic approaches to machine learning take this a step further, using symbolic methods to efficiently learn and apply “expert knowledge” in some sense, e.g. [120] which learns physics laws from data, or approaches [5, 78, 127] which apply neuro-symbolic reasoning to scene understanding, reinforcement

---

14 Some works, e.g. [44] focus more on the reduction in the memory footprint of the model. [44] achieved 50× compression.
15 An exception is [38], which shows pruning during training.
learning, and natural language processing tasks, building a high-level symbolic representation of the system in order to be able to understand and explore it more effectively with less data.

A recent study which compared neural to non-neural models of text classification found, unsurprisingly, that non-neural approaches outperformed when data was limited, but that neural approaches won out when data was copious (echoing our discussion in the theory section). For those researchers, moving to non-neural approaches produced a more than 23× speedup with a 5% drop in performance [22].

Finally, exploring combinations of the above approaches to achieve even larger compounded gains may be fruitful in reducing computation [24], or reducing environmental damage [87]. Based on the reported gains of the individual approaches so far, however, we don’t believe compounding them would yet be sufficient to dramatically bring down the very severe scaling we have seen in this work.

6 LESSENING THE ENVIRONMENTAL IMPACT

As first highlighted by Strubell et al in 2019, the environmental impact of deep learning models is an important concern. For example, they noted that training a single language model could emit as much carbon dioxide equivalent as the lifetime emissions from five cars [110]. Our analysis heightens this concern, because environmental damage grows with computing power use, and our analysis predicts enormous increases in computing power are needed for future performance improvements (See Table 3). To put the scale of this challenge into perspective, our estimations suggest that if current technology was used to train ImageNet performance to the reported gains of the individual approaches so far, however, we don’t believe compounding them would yet be sufficient to dramatically bring down the very severe scaling we have seen in this work.

6.1 REDUCING THE AMOUNT OF COMPUTATION

In many ways, reducing the amount of computation needed to train or use deep learning models is the best solution to this problem. In

the same way that “Negawatts” [75] are beneficial to the energy sector because they represent joules not used, these “NegaFlops” would be the computations never needed and thus which don’t produce any carbon emissions.

In section 5 we discussed these approaches in detail. But the broad approach is to recognize that some of the model details and data representations may be more complex than are needed to solve the problem. Efforts to simplify these take many forms, including pruning weights, quantizing the network, or using low-rank compression techniques. Early stopping and model distillation can also help by reducing training iterations and computations [17, 53].

Efforts to decrease the amount of computation have shown considerable promise, doubling in efficiency every year or so [50]. While an impressive achievement, it is important to recall that this algorithmic improvement is already accounted for in our analysis and thus it does not mitigate the escalation of environmental damage we discussed if scale-up continues at the same pace.

6.2 REDUCING THE AMOUNT OF ENERGY PER COMPUTATION

Historically, the most promising way of reducing the energy per computation was the miniaturization of processor components that underpinned Moore’s Law. According to “Dennard scaling,” for about thirty-two years, the number of computations per joule doubled every 18 months [123]. More recently, hardware designers have instead turned to specialization, building chips suited to particular calculations (at the cost of not being able to do others well) [116]. Google estimates that from one Tensor Processing Unit (TPU) generation to another, they obtained a 2.1x increase in performance and improved performance/Watt by 2.7x [61].

As discussed above, most algorithmic improvements have tackled environmental concerns indirectly, optimizing for fewer calculations rather than energy usage per se. But recently, researchers have also shown interest in explicitly designing algorithms to minimize energy usage [89, 95]. Other work in this area has looked at how to efficiently update models and retrain them without unnecessary energy overhead, and how to do better model versioning and retirement practices to further reduce energy consumption [6, 109].

Another method of reducing the environmental impact of each computation is to site those computations in efficient data centers. Such data centers can be efficient for a variety of reasons: they may be situated in areas with natural cooling, or just better designed to make the needed cooling more efficient. Server management can also make data centers more efficient, for example via techniques such as server consolidation, dynamic voltage scaling, and intelligent workload distribution [8, 80].

6.3 REDUCING THE AMOUNT OF CARBON PER ENERGY

An additional determinant of the carbon produced by deep learning models is the energy source used to power data centers. For example, in its 2021 environmental report, Google claimed that they purchased enough renewable energy to cover 100% of the electricity needed by their data centers and offices [43]. This means that hosting in some areas can yield greatly less carbon [87] than others [32].
7 CONCLUSION

The explosion in computing power used for deep learning models has ended the longstanding pessimism about AI (the “AI winter”) and set new benchmarks for computer performance on a wide range of tasks. However, deep learning’s prodigious appetite for computing power limits how far it can improve performance in its current form, particularly in an era when improvements in hardware performance are slowing. This article shows that the growing computational burden of deep learning will soon be constraining for a range of applications, making the achievement of important benchmark milestones impossible if current trajectories hold. Finally, we have discussed the likely impact of these computational limits: forcing Deep Learning towards less computationally-intensive methods of improvement, and pushing machine learning towards techniques that are more computationally-efficient than current deep learning approaches.

8 ACKNOWLEDGMENTS

The authors would like to acknowledge funding from the MIT Initiative on the Digital Economy and the Korean Government. This research was partially supported by Basic Science Research Program through the National Research Foundation of Korea (NRF), funded by the Ministry of Science, ICT & Future Planning (2017R1C1B1010094), and by the National Science Foundation (2041897). This research was sponsored in part by the United States Air Force Research Laboratory and was accomplished under Cooperative Agreement Number FA8750-19-2-1000. The views and conclusions contained in this document are those of the authors and should not be interpreted as representing the official policies, either expressed or implied, of the United States Air Force or the U.S. Government. The U.S. Government is authorized to reproduce and distribute reprints for Government purposes, notwithstanding any copyright notation herein.

REFERENCES

[1] 2017. GPU pricing, Compute Engine: Virtual Machines (VMS). https://cloud.google.com/compute/gpus-pricing. (2017).
[2] Stefano Ambrogio, Prithish Narayanan, Huanyu Tsai, Robert M Shelley, Ivoon Boybat, Carmelo di Nolfo, Severin Sidler, Massimo Giordano, Martina Bodini, and Nathan CP Farinha. 2018. Equivalent-accuracy accelerated neural-network training using analogue memory. Nature 558, 7708 (2018), 60–67.
[3] Dario Amodei, Sundaram Ananthanarayanan, Rishita Anubhai, Jingliang Bai, Earat Burger, Carl Case, Jared Casper, Bryan Catanzaro, Qiang Cheng, and Guoliang Chen. 2016. Deep speech 2: End-to-end speech recognition in english and mandarin. In International conference on machine learning. PMLR, 173–182.
[4] Dario Amodei and Danny Hernandez. 2018. AI and Compute. Open AI Blog Article (2018). https://openai.com/blog/ai-and-compute/.
[5] Masatoshi Asai and Christian Muise. 2020. Learning Neural-Symbolic Descriptive Planning Models via Cube-Space Priors: The Voyage Home (to STRIPS). arXiv preprint arXiv:2004.12850 (2020).
[6] Soheil Bahramzour, Naveen Ramakrishnan, Lukas Schott, and Mohak Shah. 2015. Comparative Study of Caffe, Neon, Theano, and Torch for Deep Learning. CoRR abs/1511.06435 (2015). http://arxiv.org/abs/1511.06435
[7] Andre Barbu, David Mayo, Julian Alverio, William Luo, Christopher Wang, Dan Gutfreund, Josh Tenenbaum, and Boris Katz. 2019. ObjectNet: A large-scale bias-controlled dataset for pushing the limits of object recognition models. In Advances in Neural Information Processing Systems 32. H. Wallach, H. Larochelle, A. Beygelzimer, F. Alch"{e}-Buc, E. Fox, and R. Garnett (Eds.). Curran Associates, Inc., 9453–9463.
[8] Luiz André Barroso, Urs Hölzle, and Parthasarathy Ranganathan. 2019. The Datacenter as a Computer. Springer International Publishing. DOI: http://dx.doi.org/10.1007/978-3-030-01761-2
[9] Mikhail Belkin, Daniel Hsu, Siyuan Ma, and Soumik Mandal. 2019. Reconciling modern machine-learning practice and the classical bias–variance trade-off. Proceedings of the National Academy of Sciences 116, 32 (2019), 15849–15854. DOI: http://dx.doi.org/10.1073/pnas.1903070116
[10] Mikhail Belkin, Daniel J Hsu, and Partha Mitra. 2018. Overfitting or perfect fitting? risk bounds for classification and regression rules that interpolate. In Advances in Neural Information Processing Systems. 2308–2311.
[11] Mikhail Belkin, Alexander Rakhlin, and Alexandre B Tsybakov. 2019. Does data interpolation contradict statistical optimality?. In The 22nd International Conference on Artificial Intelligence and Statistics. 1611–1619.
[12] Emily M. Bender, Timnit Gebru, Angelina McMillan-Major, and Shmargaret Shmitchell. 2021. On the Dangers of Stochastic Parrots: Can Language Models Be Too Big?. In Proceedings of the 2021 ACM Conference on Fairness, Accountability and Transparency (FACT ’21). Association for Computing Machinery, New York, NY, USA, 610–623. DOI: http://dx.doi.org/10.1145/3442188.345922
[13] Gabriel Bender, Pieter-Jan Kindermans, Barret Zoph, Vijay Vasudevan, and Quoc Le. 2018. Understanding and Simplifying One-Shot Architecture Search. In Proceedings of the 35th International Conference on Machine Learning (Proceedings of Machine Learning Research), Jennifer Dy and Andreas Krause (Eds.). Vol. 80. PMLR, 550–559. https://proceedings.mlr.press/v80/bender18a.html
[14] Davis Blalock, Jose Javier Gonzalez Ortiz, Jonathan Franklin, and John Guttag. 2020. What is the State of Neural Network Pruning?. In Proceedings of Machine Learning and Systems, I. Dhillon, D. Papailiopoulos, and V. Sze (Eds.), Vol. 2. 129–146.
[15] Han Cai, Chuang Gan, Tianzhe Wang, Zhehai Zhang, and Song Han. 2020. Once for All: Train One Network and Specialize it for Efficient Deployment. In International Conference on Learning Representations.
[16] Han Cai, Ligang Zhu, and Song Han. 2019. ProxylessNAS: Direct Neural Architecture Search on Target Task and Hardware. In International Conference on Learning Representations.
[17] Rich Caruana, Steve Lawrence, and Lee Giles. 2001. Overfitting in neural nets: Backpropagation, conjugate gradient, and early stopping. In Advances in Neural Information Processing Systems 13 - Proceedings of the 2000 Conference, NIPS 2000. Neural information processing systems foundation. 14th Annual Neural Information Processing Systems Conference, NIPS 2000 ; Conference date: 27-11-2000 Through 02-12-2000.
[18] Chun-Fu Chen, Quanfu Fan, Neil Mallinar, Tom Sercu, and Rogerio Feris. 2018. Big-lit-net: An efficient multi-scale feature representation for visual and speech recognition. arXiv preprint arXiv:1807.03848 (2018).
[19] Yungpeng Cheng, Jianan Li, Huaxin Xiao, Xiaojuo Jin, Shuicheng Yan, and Jiashi Feng. 2017. Dual Path Networks. CoRR abs/1707.01629 (2017). arXiv.org/abs/1707.01629
[20] Yu Cheng, Duo Wang, Pan Zhou, and Tao Zhang. 2017. A survey of model compression and acceleration for deep neural networks. arXiv preprint arXiv:1710.09282 (2017).
[21] Andrew W Cross, Lev S Bishop, Sarah Sheldon, Paul D Nation, and Jay M Gambett. 2019. Validating quantum computers using randomized model circuits. Physical Review A 100, 3 (2019), 032328.
[22] Washington Cunha, Ð’aptor Mangaravite, Christian Gomes, Sà­ľrgio Canuto, Elaine Resende, Cecilia Nascimento, Felipe Viegas, Celso FranÃ¡gua, Wellington Santos Martins, Jussara M. Almeida, Thierry Rosa, Leonardo Rocha, and Marcos AndrÃ­Al GonÃ¡lves. 2021. On the cost-effectiveness of neural and non-neural approaches and representations for text classification: A comprehensive comparative study. Information Processing & Management 58, 5 (2021), 102481. DOI: https://dx.doi.org/10.1016/j.ipm.2020.102481
[23] Maurizio Ferrari Dacrema, Paolo Cremonesi, and Dietmar Jannach. 2019. Are We Fitting? risk bounds for classification and regression rules that interpolate. In Proceedings of the 13th ACM Conference on Recommender Systems (RecSys ’19). Association for Computing Machinery, New York, NY, USA, 101â–ÅŠ109. DOI: http://dx.doi.org/10.1145/3296689.3340758
[24] William J. Dally, Yatish Turakhia, and Song Han. 2020. Domain-Specific Hardware Accelerators. Commun. ACM 63, 7 (June 2020), 48â–ÅŠ57. DOI: http://dx.doi.org/10.1145/3361682
[25] Andrew Danowitz, Kyle Kelley, James Mao, John P. Stevenson, and Mark Horowitz. 2020. Validating quantum computers using randomized model circuits. Queue 18, 2 (June 2020), 285–299. DOI: http://dx.doi.org/10.1145/3361682
[26] Peter Duvenege. 2020. The globalization of artificial intelligence: consequences for the politics of environmentalism. Globalizations 17, 2 (June 2020), 285–299. DOI: http://dx.doi.org/10.1080/14747731.2020.1785670
[27] Mike Davies. 2019. Progress in Neuromorphic Computing: Drawing Inspiration from Nature for Gains in AI and Computing. In 2019 International Symposium on VLSI Technology, Systems and Application (VLSI-TSA). IEEE, 1–1.
[28] Jeff Dean. Sustainable Computation and Machine Learning Platforms at Google. Commun. ACM 63, 7 (June 2020), 48â–ÅŠ57. DOI: http://dx.doi.org/10.1145/3361682
[29] Andrew Danowitz, Kyle Kelley, James Mao, John P. Stevenson, and Mark Horowitz. 2020. Validating quantum computers using randomized model circuits. Physical Review A 100, 3 (2019), 032328.
(2017). http://arxiv.org/abs/1707.07012