High-dimensional Optimization and Compute-Optimal Scaling Laws

Introduction au domaine de recherche, Damien Ferbach

Supervised by Gauthier Gidel

Contents

1 Introduction

1.1 Motivation

The recent and impressive progress of Large Langage Models (LLMs) (Brown, 2020) has been largely driven by the increasing availability of immense compute resources. The performances of a model are in fact highly correlated to the amount of compute that was used to train it. This had shed light on neural scaling laws, namely precise upper and lower bounds on the performance of a model as we increase its number of parameters and training compute. Understanding neural scaling laws is crucial for the developmenet and efficient training of large models. Indeed, in the training pipeline of a model, a substantial part of the compute is wasted in expensive trial and searches procedures to find the best hyper-parameters to train a model. For example, it can be its architecture, its size, or the parameters of the algorithm used to train it, such as learning rate or batch size in the case of gradient descent. Instead, neural scaling laws have the potential to predict, before training the model, which hyper-parameters will be most efficient by taking into account that one has only a finite amount of compute available.

1.2 Compute optimal scaling laws

To formalize this idea, consider the following general learning problem:

$$\arg\min_{\theta \in \mathbb{R}^d} \mathcal{P}(\theta) := \mathbb{E}_{x,y}[\mathcal{R}(\theta, x, y)]$$

Here θ are the parameters of the model, $\mathcal{P}(\theta)$ is the loss, x, y follows the data distribution, and $\mathcal{R}(\theta, x, y)$ is a cost function measuring how much the model with parameters θ predicts the output y when given the input x. To solve this problem, we assume to have a fixed amount of compute available, representing the number of GPU-hours at disposal. We can use a standard formula to measure compute:

$$\text{Compute } (\mathfrak{f}) = \text{parameters (d)} \times \text{iterations of alg. (r)}$$

Here, $\mathfrak{f}$ represents the amount of compute that one has to train a model (in GPU.hours, or Floating Point Operations, FLOPs), d represents the number of parameters of the trained model, and r the number of iterations of the learning algorithm, usually stochastic gradient descent (SGD) and its variants. In particular, for d large, the model has a large expressive power, but cannot be trained for a long time, since the compute $\mathfrak{f}$ is fixed. On the other hand, for r large the model can be trained for a long time but can potentially not capture all the complexity in the data. We aim at finding the dimension which allows for the lowest loss after training the parameters for r iterations to θ_r, i.e. $d_* = \min_d \mathcal{P}(\theta_r, d) = \min_d \mathcal{P}(r, d) = \min_d \mathcal{P}(\frac{\mathfrak{f}}{d}, d)$.

Empirical evidence suggest power law relationships between $d_*, \mathcal{P}(d_*)$ and $\mathfrak{f}$, of the form $d_* = \mathfrak{f}^{\xi}$ and $\mathcal{P}(d_*) = f^{-\eta}$ whith ξ the parameter count exponent, and η the scaling law exponent. For example, (Hoffmann et al., 2022) empirically shows that $\xi = \eta = \frac{1}{2}$. On the theory side, (Paquette et al., 2024) show that stochastic gradient descent (SGD) on a power law random features model (PLRF) exhibits multiple different phases with different scaling law and parameter count exponent depending on the data complexity. However, in a large part of these phases, they recover the scaling of (Hoffer et al., 2017).

Plan In this thesis, we will show how to obtain neural scaling laws for stochastic gradient descent. In section 2, we first give some independent background on results in convex optimization first and then in high dimensional optimization. In section 3 we introduce a useful and rich data model to study: the power law random features (PLRF) model and give some background in random matrix theory. Finally in section 4 we results in establishing compute optimal scaling laws for SGD on the PLRF model.

2 Background on high dimensional optimization

2.1 Worst Case Analysis of Gradient Descent Algorithms

The gradient descent algorithm on the risk $\mathcal{P}(\theta) : \mathbb{R}^d \to \mathbb{R}$ with learning rate $\gamma(r)$ and initialization $\theta_0 \in \mathbb{R}^d$ consists in updates of the form

$$\theta_{r+1} = \theta_r - \gamma(r)\nabla\mathcal{P}(\theta_r).$$

Assuming that $\mathcal{P}(\theta) := \mathbb{E}_{x,y}[\mathcal{R}(\theta, x, y)]$, streaming SGD consists in replacing the gradient $\nabla\mathcal{P}(\theta_r)$ by the unbiased estimator using one data point (x_r, y_r) sampled iid from the data distribution as

$$\theta_{r+1} = \theta_r - \gamma(r)\nabla\mathcal{R}(\theta, x_r, y_r).$$

The good assumptions to make on $\mathcal{P}$ to study gradient descent algorithms algorithms are smoothness and convexity. A function $\mathcal{P} : \mathbb{R}^d \to$ is L-smooth if its gradient is L-Lipschitz, ie:

$$\forall x, y \in \mathbb{R}^d, \quad \|\nabla\mathcal{P}(x) - \nabla\mathcal{P}(y)\|_2 \leq L\|x - y\|_2$$

Proposition 2.1. *Let $\mathcal{P} : \mathbb{R}^d \to \mathbb{R}$ an L-smooth convex function with a global minimizer θ_* and any $\theta_0 \in \mathbb{R}^d$. Then the iterates of gradient descent with $\gamma(r) = \frac{1}{L}$ satisfy after r steps:*

$$\mathcal{P}(\theta_r) - \mathcal{P}(\theta_*) \leq \frac{L}{2r}\|\theta_0 - \theta_*\|^2$$

We therefore see that GD converges in $\frac{1}{r}$ speed on smooth convex functions. A modification of this algorithm called Nesterov momentum yields updates as:

$$\theta_r = \eta_{r-1} - \frac{1}{L}\nabla\mathcal{P}(\eta_{r-1})$$
$$\eta_t = \theta_t + \frac{t-1}{t+2}(\theta_t - \theta_{t-1})$$

In that case, it is possible to show that on smooth convex functions we have:

Proposition 2.2. *Let $\mathcal{P} : \mathbb{R}^d \to \mathbb{R}$ an L-smooth convex function with a global minimizer θ_* and any $\theta_0 \in \mathbb{R}^d$. Then the iterates of gradient descent with Nesterov momentum with $\gamma(r) = \frac{1}{L}$ satisfy after r steps:*

$$\mathcal{P}(\theta_r) - \mathcal{P}(\theta_*) \leq \frac{2L}{(r+1)^2}\|\theta_0 - \theta_*\|^2$$

The convergence rate is now $\frac{1}{r^2}$ which accelerates with respect to Gradient Descent.

2.2 Worst case vs average case analysis

Propositions 2.1 and 2.2 are results in worst case in the sense that they hold for any L-smooth convex function and any initialization. This is a very nice property but also a weakness. Indeed, in most of cases of interests, empirical evidence shows that the convergence is much faster or with different conditions on the learning rates. For example, consider he minimization of a quadratic $\mathcal{P}(\theta) = \frac{1}{2}\theta^T H\theta$ with H a positive definite symmetric matrix with eigenvalues $\lambda_1 \leq \lambda_2 \leq \cdots \leq \lambda_d$. Then, a classic result states that to ensure convergence of gradient descent, we must choose the learning rate $\gamma \leq \frac{2}{\lambda_d}$. In that case, if λ_d is very large, it imposes a strong bound on the possible learning rate. However, we will see below that in high dimensions, and considering stochastic gradient descent, averaging effects allow to weaken the bound as $\gamma \leq \frac{2d}{tr(H)}$. Only the mean eigenvalue of H now matters for the convergence of the algorithm. Especially, in high dimensions, averaging and concentration effects happen which leads us to move beyond the worst case setting.

2.3 Optimization of a quadratic in high dimension

We consider the minimization of a quadratic using stochastic gradient descent. Suppose $x \sim \mathcal{N}(0, K)$ where $K \in \mathbb{R}^{d\times d}$ is symmetric positive definite. Let $b \in \mathbb{R}^d$ the ground truth parameters and $\varepsilon \sim \mathcal{N}(0, \eta^2)$ a gaussian noise. In this problem, we aim at minimizing:

$$\mathcal{P}(\theta) = \frac{1}{2}\mathbb{E}\left[(\langle x, \theta\rangle - (\langle x, b\rangle + \varepsilon))^2\right] = \frac{1}{2}\|\theta - b\|_K^2 + \frac{1}{2}\eta^2$$

To that end, we perform stochastic gradient descent starting from $\theta_0 = 0 \in \mathbb{R}^d$:

$$\begin{aligned}\theta_{r+1} &= \theta_r - \frac{\gamma}{d}\nabla\mathcal{R}(\theta_r; x_{r+1}, \langle x_{r+1}, b\rangle + \varepsilon_{r+1}) \\ &= \theta_{r+1} - \frac{\gamma}{d}\left(\langle x_{r+1}, \theta_r\rangle - (\langle x_{r+1}, b\rangle + \varepsilon_{r+1})\right) x_{r+1}\end{aligned}$$

Notice that we scaled the learning rate by the dimension $\gamma \to \frac{\gamma}{d}$ to ensure the good scaling of the algorithm.

We then introduce the following stochastic differential equation (SDE):

Definition 2.3 (Homogenized SGD)**.** *Define Θ_t the solution of the following SDE with $\Theta_0 = 0$:*

$$d\Theta_t = -\gamma\nabla\mathcal{P}(\Theta_t) + \gamma\sqrt{\frac{2\mathcal{P}(\Theta_t)K}{d}}dB_t$$

where B_t is a standard d-dimensional Brownian motion.

It can then be shown that the homogenized SGD solution stays close to the SGD discrete updates as the dimension goes to infinity. More precisely it was shown in the following lemma:

Lemma 2.4 (Homegenized SGD stays close to SGD)**.** *Let $q : \mathbb{R}^d \to \mathbb{R}$ be a quadratic with $\|q\|_{C^2} := \sup_x \|\nabla^2 q(x)\|_\sigma + \|\nabla q(0)\| + |q(0)|$ where $\|\cdot\|_\sigma$ is the nuclear norm. Then, for any $\varepsilon > 0$, there is a constant $C(\|K\|_\sigma)$ such that the processes $\{\theta_r\}_{r=0}^n, \{\Theta_{r/d}\}_{r=0}^n$ satisfy for $n \le d\log(d)/C(\|K\|_\sigma)$:*

$$\sup_{k=0\dots n} |q(\theta_k) - q(\Theta_{k/d})| \le \|q\|_{C^2} e^{C(\|K\|_\sigma)n/d} d^{-1/2+9\varepsilon}$$

with overwhelming probability.

The proof uses martingale concentration tools. The next step, is to describe the risk curve for homogenized SGD. To that end, we will rewrite the risk using the Spectral mapping theorem.

Definition 2.5 (Resolvent)**.** *Let a matrix $A \in \mathbb{R}^{d\times d}$. The resolvent of A is the mapping $R : \mathbb{C} \setminus Spec(A) \to \mathbb{R}^{d\times d}$:*

$$R(z; A) = (A - zI_d)^{-1}$$

We can then rewrite the risk:

Lemma 2.6 (Spectral mapping theorem)**.** *Define*

$$Q_t(z) = \frac{1}{2}\langle R(z, K), (\Theta_t - b)^{\otimes 2}\rangle.$$

Suppose to simplify that $\eta = 0$. Then if Γ is a contour of $\mathbb{C}$ enclosing the eigenvalues of K,

$$\mathcal{P}(\Theta_t) := \frac{1}{2}\|\theta - b\|_K^2 = \frac{1}{2i\pi}\oint_\Gamma zQ_t(z)dz$$

Proof. This is a simple application of the Spectral Mapping theorem by writing:

$$\mathcal{P}(\theta_t) = \frac{1}{2}\langle K, (\theta_t - b)^{\otimes 2}\rangle$$

□

The advantage of this decomposition is that the familiy $Q_t(z), z \in \Gamma$ is a family of statistics that closes. In other words, the evolution equation of $\mathcal{P}(\Theta_t)$ that comes from applying Ito formula can be simplified. Especially, applying Ito formula on each $Q_t(z)$ and integrating along the contour Γ yields the following equation:

Definition 2.7 (Convolution Volterra equation)**.** *Let $\Psi(t)$ solve the following convolution equation:*

$$\Psi(t) = \mathcal{F}(t) + \int_0^t \mathcal{K}(t-s)\Psi(s)ds$$

where:

$$\begin{cases} \mathcal{F}(t) = \mathcal{P}(\mathcal{X}_t) \\ \mathcal{K}(t) = \gamma^2 \frac{Tr(K^2 e^{-2\gamma K t})}{d} \end{cases}$$

where $\mathcal{X}(t)$ is the solution of gradient flow:

$$\frac{d\mathcal{X}_t}{dt} = -\gamma \nabla \mathcal{P}(\mathcal{X}_t)$$

It has been shown that homogenized SGD approximately solves the convolution Volterra equation:

Lemma 2.8 (Homogenized SGD solves the Volterra equation)**.** *$\forall \varepsilon > 0, T > 0$ we have with overwhelming probability:*

$$\sup_{t \le T} |\mathcal{P}(\Theta_t) - \Psi(t)| \le C(T, \|K\|_\sigma) d^{-1/2+\varepsilon}$$

Now it is possible to obtain average case results on the learning rate that allows convergence of the algorithm. In particular, it is standard from renewal equation theory that $\Psi(t)$ converges when $\mathcal{F}(t)$ itself converges and the kernel satisfies $\|\mathcal{K}\|_{L^1} < 1$. The second condition directly brings:

Lemma 2.9 (Convergence condition)**.** *$\Psi(t)$ converges if $\gamma \le \frac{2d}{Tr(K)}$.*

3 The Power-Law Random Features Model

3.1 Motivation for this model

The power-law random features model (PLRF) has been introduced in Maloney et al. (2024) to represent the distribution of random features on real world

datasets, and as a model that can induce power laws of the loss with respect to dimension or number of iterations of the training algorithm. The model can be summarized as follows:

Definition 3.1 (PLRF model)**.** *Let $v \geq d \geq 1$, let $\alpha, \beta > 0$. The PLRF model consists in solving:*

$$\min_{\theta} \mathbb{E}\left[\left(\langle W^T x, \theta\rangle - \langle x, b\rangle\right)^2\right]$$

where $\theta \in \mathbb{R}^d$ are the parameters to optimize, $W \in \mathbb{R}^{v\times d}$ is a random matrix sampled as $W_{i,j} \overset{iid}{\sim} \mathcal{N}(0, \frac{1}{d})$, $x, b \in \mathbb{R}^v$ with $x_j \overset{iid}{\sim} j^{-\alpha}\mathcal{N}(0,1)$, $b_j = j^{-\beta}$.

α, β are parameters measuring the complexity of the data distribution. W is a random projection matrix. The observable is $W^T x$ and θ are the parameters to reproduce the target $\langle x, b\rangle$.

3.2 Random Matrix Theory and the Dyson Equation

The dynamics of gradient descent on the PLRF model, are determined by the eigenvalue distribution of the data covariance matrix $\check{K} := W^T D W$, where $D = \mathrm{Diag}(j^{-2\alpha}, j = 1 \cdots d)$. Since W is random, $W^T D W$ is a random matrix. To study its eigenvalues, a general method is to use a detreministic equivalent on its resolvent which obeys a Dyson fixed point equation. Paquette et al. (2024) expressed this fixed point equation as:

$$R(z) = \mathrm{Diag}\left(\frac{1}{j^{-2\alpha} m(z) - z}, 1 \leq j \leq d\right) \quad \text{with} \quad m(z) = \frac{1}{1 + \frac{1}{d}\sum_{j=1}^{d} \frac{j^{-2\alpha}}{j^{-2\alpha} m(z) - z}}$$

It is possible from this fixed point equation to obtain equivalents, as $d \to \infty$ on $R(z), z \in \mathbb{C}$. This will be crucial to compute contour integrals in section 4.

4 Compute optimal scaling laws for Stochastic Gradient Descent

We will now describe how to obtain compute optimal scaling laws on the PLRF model for stochastic gradient descent as was done in Paquette et al. (2024). The update equation of SGD is:

$$\theta_{r+1} = \theta_r - \gamma \times W^T x\big(\langle W^T x, \theta_r\rangle - \langle x, b\rangle\big) \tag{1}$$

with γ a step size parameter.

4.1 Simulation of the dynamics

It is possible to simulate the SGD updates dynamics. In fig. 1 we directly see that fig. 1 induce a power law compute optimal frontier (dotted red line).

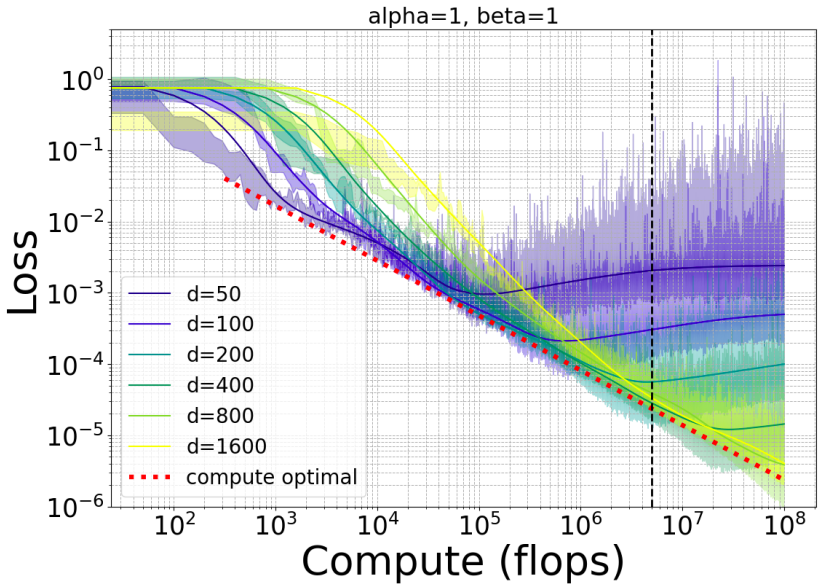

Figure 1: The discrete updates dynamic induces a power law compute optimal frontier. For a compute $\mathfrak{f} = 5.10^6$ flops, the optimal dimension is $d_* = 400$

4.2 Volterra Equation

The previous updates directly imply that the risk is solution of a convolution Volterra equation. The loss is given by

$$\mathcal{P}(r) = \underbrace{\|D^{1/2}\dot{b}\|^2 + \langle \check{K}(I - 2\gamma\check{K} + 2\gamma^2\check{K}^2)^r, (\theta_0 - \check{b})^{\otimes 2}\rangle}_{\mathcal{F}(t)} \tag{2}$$

$$+ \sum_0^r \mathcal{P}(s) \underbrace{\gamma^2 Tr(\check{K}^2(I - 2\gamma\check{K} + 2\gamma^2\check{K}^2)^{r-s-1})}_{\mathcal{K}(r-s-1)} ds \tag{3}$$

$$= \mathcal{F}(t) + \sum_0^r \mathcal{K}(r-s-1)\mathcal{P}(s)ds \tag{4}$$

This form of equation is also known as renewal equation in probability theory. It is known that under mild conditions such as $\mathcal{F}(r)$ bounded and $\|\mathcal{K}\|_{L_1} \overset{\text{def}}{=} \sum_0^\infty \mathcal{K}(r) < 1$, then there exists a unique solution to that equation of the form:

$$\mathcal{P}(r) = \mathcal{F}(r) + (\mathcal{F} * \mathcal{K})(r) + (\mathcal{F} * \mathcal{K} * \mathcal{K})(r) + ...$$

In particular, it is possible to bound for a constant C sufficiently large:

$$\frac{1}{C}\left(\mathcal{F}(r) + (\mathcal{F} * \mathcal{K})(r)\right) \leq \mathcal{P}(r) \leq C\left(\mathcal{F}(r) + (\mathcal{F} * \mathcal{K})(r)\right)$$

Finally, when $\mathcal{F}, \mathcal{K}$ do not decrease too fast, it is possible to bound:

$$\frac{1}{C}(\mathcal{F}(r) + \mathcal{K}(r)) \leq (\mathcal{F} * \mathcal{K})(r) \leq C(\mathcal{F}(r) + \mathcal{K}(r))$$

As a conclusion, **the asymptotics of $\mathcal{P}$ are entirely determined by the asymptotics of the forcing and kernel functions $\mathcal{F}, \mathcal{K}$.**

4.3 Random matrix theory to understand the spectrum of $W^T DW$

We therefore need to compute the sums:

$$\begin{cases} \mathcal{F}(r) = \|D^{1/2}\dot{b}\|^2 + \langle \check{K}(I - 2\gamma\check{K} + 2\gamma^2\check{K}^2)^r, (\theta_0 - \check{b})^{\otimes 2}\rangle \\ \mathcal{K}(r) = \gamma^2 Tr(\check{K}^2(I - 2\gamma\check{K} + 2\gamma^2\check{K}^2)^{r-s-1}) \end{cases}$$

The problem is that we do not know the eigenvalues λ_j of the random matrix $W^T DW$. However, as explained in section 3.2, we can characterize a deterministic equivalent of the resolvent $R(z; W^T DW) = (W^T DW - zId)^{-1}$.

We can then formally use Cauchy theorem and write the sum as an integral around a contour Γ enclosing the eigenvalues of $\hat{K} := D^{1/2}WW^T D^{1/2}$:

$$\begin{cases} \mathcal{F}(r) = -\frac{1}{2i\pi}\oint_\Gamma (1 - 2\gamma z + 2\gamma^2 z^2)^r \cdot \langle R(z; \hat{K}), (D^{1/2}b)^{\otimes 2}\rangle) dz \\ \mathcal{K}(r) = -\frac{1}{2i\pi}\oint_\Gamma Tr(z^2(1 - 2\gamma z + 2\gamma^2 z^2)^r \cdot R(z; \hat{K})) dz \end{cases}$$

5 Conclusion

We have seen how the asymptotic of the risk for the PLRF model optimized by SGD, is related to the forcing and kernel function of a Volterra equation. By studying the spectrum of the data covariance matrix, we can get asymptotic on these two functions hence characterizing the speed of convergence of the algorithm.

References

Brown, T. B. (2020). Language models are few-shot learners. *arXiv preprint arXiv:2005.14165*.

Hoffer, E., Hubara, I., and Soudry, D. (2017). Train longer, generalize better: closing the generalization gap in large batch training of neural networks. *Advances in Neural Information Processing Systems (NeurIPS)*, 30.

Hoffmann, J., Borgeaud, S., Mensch, A., Buchatskaya, E., Cai, T., Rutherford, E., de Las Casas, D., Hendricks, L., Welbl, J., Clark, A., Hennigan, T., Noland, E., Millican, K., van den Driessche, G., Damoc, B., Guy, A., Osindero, S., Simonyan, K., Elsen, E., Vinyals, O., Rae, J., and Sifre, L. (2022).

An empirical analysis of compute-optimal large language model training. In *Advances in Neural Information Processing Systems*, volume 35.

Maloney, A., Roberts, D., and Sully, J. (2024). A solvable model of neural scaling laws. *arXiv preprint arXiv:2210.16859*.

Paquette, E., Paquette, C., Xiao, L., and Pennington, J. (2024). 4+ 3 phases of compute-optimal neural scaling laws. *arXiv preprint arXiv:2405.15074*.