Scaling Laws for Fine-Grained Mixture of Experts

Jan Ludziejewski [*1 2] **Jakub Krajewski** [*1 2] **Kamil Adamczewski** [1] **Maciej Pióro** [1 3] **Michał Krutul** [1 2] **Szymon Antoniak** [1 2] **Kamil Ciebiera** [1 2] **Krystian Król** [1 2] **Tomasz Odrzygóźdź** [4] **Piotr Sankowski** [1 2] **Marek Cygan** [2 5] **Sebastian Jaszczur** [*1 2]

Abstract

Mixture of Experts (MoE) models have emerged as a primary solution for reducing the computational cost of Large Language Models. In this work, we analyze their scaling properties, highlighting certain arbitrary assumptions present in the existing literature. In particular, we introduce a new hyperparameter, granularity, the modification of which allows for the optimal adjustment of the size of experts. Subsequently, we present scaling laws for fine-grained MoE, taking into account the number of training tokens, model size, and granularity. Using these scaling laws, we derive the optimal training configuration for a given computational budget. Furthermore, in contrast with previous works, we demonstrate that the gap in efficiency between dense and MoE models grows as we scale up the model size and training budget.

1. Introduction

In recent years, we have witnessed Large Language Models (LLMs) achieve exceptional performance in tasks across numerous domains (Chowdhery et al., 2022; Yin et al., 2023; Agostinelli et al., 2023). However, training those massive models incurs high computational costs, measured in millions of GPU-hours, (Touvron et al., 2023b) enabled only by enormous budgets (Scao et al., 2023) and leading to non-negligible carbon footprints (Faiz et al., 2024). To combat these obstacles, the research community has been striving to increase the efficiency of LLMs. One promising approach that has lately been gaining visibility is the use of Mixture of Experts (MoE) methods. Models such as Switch (Fedus et al., 2022) and Mixtral (Jiang et al., 2024) have already demonstrated that MoE models can achieve effectiveness comparable to dense models with significantly lower computational costs.

In the context of the current trend of increasing budgets for training models, a question arises: will MoE models continue to be attractive in the future? This is an important issue, as results from other studies (Clark et al., 2022) suggest that the traditional dense models may outperform MoE as the size of models increases.

In this paper, we argue that previous claims lose their validity when we relax certain implicit assumptions regarding the training process present in previous research (Clark et al., 2022). In particular, we refer to the fixed training duration and the constant size of experts in MoE models.

Our results suggest that a compute-optimal MoE model trained with a budget of 10^{20} FLOPs will achieve the same quality as a dense Transformer trained with a $20\times$ greater computing budget, with the compute savings rising steadily, exceeding $40\times$ when budget of 10^{25} FLOPs is surpassed (see Figure 1).

Our main contributions are:

1. Introducing a new hyperparameter - granularity. Adjusting this parameter allows us to determine the optimal size of experts in MoE models, which translates into increased efficiency (see Figure 1b and Figure 5).

2. Deriving new scaling laws for MoE models that incorporate variable training duration, the number of parameters, and granularity. Such scaling laws allow us to calculate optimal training hyperparameters for MoE models.

3. Demonstrating that, with optimal settings, MoE models can always outperform traditional Transformers at any computing budget. This is a conclusion contrary to the results from (Clark et al., 2022), see Section 6.3.

Additionally, we open-source the code used to produce the results described in this work at github.com/llm-random/llm-random.

[*]Equal contribution [1]IDEAS NCBR [2]University of Warsaw [3]Polish Academy of Sciences [4]TradeLink [5]Nomagic. Correspondence to: Sebastian Jaszczur <s.jaszczur@uw.edu.pl>.

Proceedings of the 41st International Conference on Machine Learning, Vienna, Austria. PMLR 235, 2024. Copyright 2024 by the author(s).

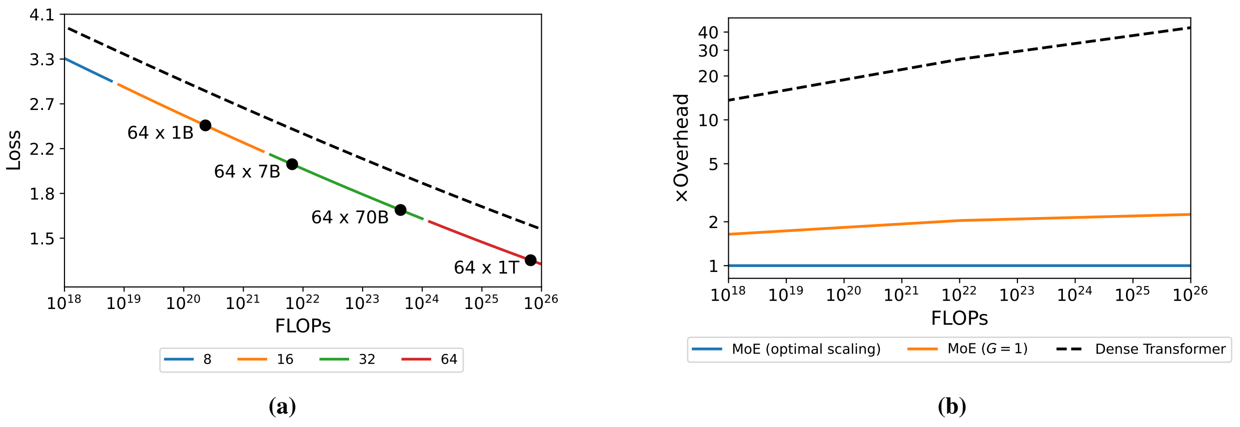


Figure 1. Mixture-of-Experts can be *always* considered more efficient than dense Transformers, regardless of the model size. **(a)** Compute Optimal scaling curves for granular models and standard Transformers. The dashed line represents a dense Transformer. Colors denote optimal granularity for the given FLOPs training budget. **(b)** Relative number of FLOPs needed to train Transformer and Vanilla MoE (MoE with $G = 1$) to achieve the performance of MoE with compute optimal G.

2. Related Work

Mixture of Experts. In the context of language modeling, MoE was first introduced by (Shazeer et al., 2017) as a sparsely gated layer between stacked blocks of LSTM (Hochreiter & Schmidhuber, 1997). A similar technique was proposed in the context of Transformers by (Shazeer et al., 2018) and (Lepikhin et al., 2020). (Fedus et al., 2022) proposed to route each input to only a single expert and designed a modified initialization scheme to reduce training instability.

Numerous studies have proposed to modify the original routing method. (Lewis et al., 2021) used a linear assignment algorithm to postprocess token-expert mappings and ensure even expert selections. (Roller et al., 2021) suggested another approach involving deterministic hash functions. (Zhou et al., 2022) proposed expert choice routing, eliminating the need for additional load balancing losses. (Puigcerver et al., 2023) designed a fully-differentiable Soft MoE architecture. Concurrently to our work, (Dai et al., 2024) proposed to modify the MoE layer by segmenting experts into smaller ones and adding shared experts to the architecture. Independently, (Liu et al., 2023) suggested a unified view of sparse feed-forward layers, considering, in particular, varying the size of memory blocks. Both approaches can be interpreted as modifying granularity. However, we offer a comprehensive comparison of the relationship between training hyperparameters and derive principled selection criteria.

Scaling Laws. Scaling laws are empirically derived equations relating the loss of a model with variables such as the number of parameters, training samples, or the computational budget. In the case of dense Transformers, scaling laws were first studied by (Kaplan et al., 2020), who observed power law relationships between the final model perplexity and model and dataset size. This work was extended by (Hoffmann et al., 2022), by considering variable cosine cycle lengths, and formulating a modified functional form of the scaling equation.

Scaling laws have also been proposed for other architectures and training scenarios. (Henighan et al., 2020) studied autoregressive modeling across various modalities, while (Ghorbani et al., 2021) considered machine translation. (Frantar et al., 2023) explored the impact of pruning on vision and language Transformers, deriving optimal sparsity for a given compute budget. (Clark et al., 2022) studied the scaling of MoE when changing model size and number of experts on a fixed dataset, concluding that routed models are more efficient only until a certain model size. In this work, we challenge that claim by considering a variable, optimal dataset size for both model families (see Section 6.3).

3. Background

3.1. Model Architecture

Transformer. A standard decoder-only Transformer (Radford et al., 2018a;b; Kaplan et al., 2020; Brown et al., 2020) consists of an embedding layer, a stack of alternating attention and feed-forward layers, and an unembedding layer. In the model, each input token is converted by the embedding layer into a vector of size d_{model}, the dimension maintained across all the layers in the residual stream.

The feed-forward component consists of two linear transformations and a nonlinearity ϕ in between. It can be described

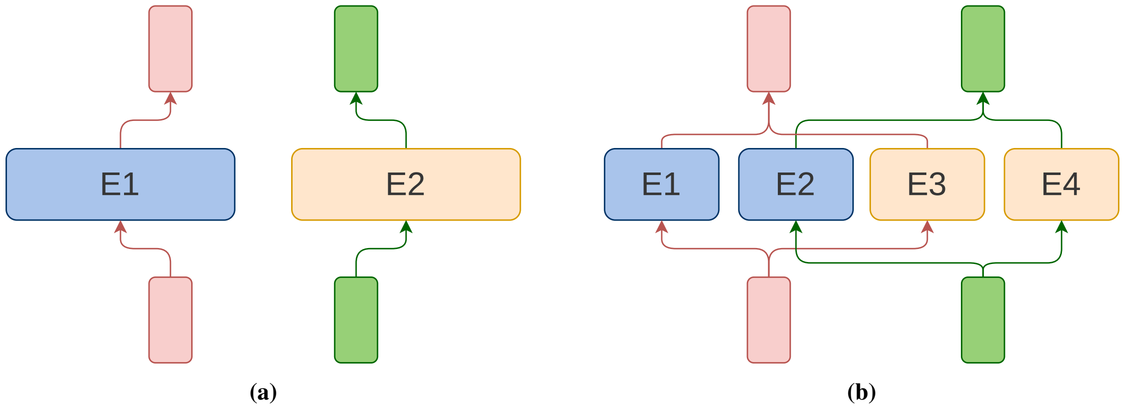


Figure 2. **(a)** Standard MoE layer with $G = 1$ **(b)** Corresponding MoE layer with $G = 2$. Each of the original experts is split into two granular ones. The split occurs in the hidden dimension of an expert. Increasing G allows for a more precise mapping between experts and tokens. Since for granularity G, the token is routed to G granular experts, the number of parameters activated per token is the same in both cases.

as $\text{FFN}(x) = \phi(xW_1 + b_1)W_2 + b_2$, with W_1 mapping from d_{model} to d_{ff}, and W_2 back to the original d_{model}. It is standard (Radford et al., 2018a; Rae et al., 2022; Touvron et al., 2023a; Jiang et al., 2023) to set the hidden dimension as $d_{\text{ff}} = 4 \cdot d_{\text{model}}$.

Feed-forward layers contain the majority of Transformer parameters and require the biggest computational budget counted in terms of FLOPs. Subsequently, they are the main focus of the Mixture of Experts models considered in this work.

Mixture of Experts. The core idea behind MoE in Transformers is to replace the feed-forward layer with a set of *experts*. The size of each expert is typically (Fedus et al., 2022; Zhou et al., 2022; 2023; Jiang et al., 2024) set to mirror the original dimensions of the layer, with the hidden expert dimension d_{expert} equal to d_{ff}. Therefore, the total number of parameters in MoE scales linearly with the number of experts. However, the computational cost remains approximately constant as each input is routed and then processed by a subset of experts.

3.2. Scaling Laws

Dense Transformers. Large Transformer-based models are known to approximately obey the power-law relationship between final loss $\mathcal{L}$, model size N, and number of training tokens D. This relationship is often called *Chinchilla scaling laws* described in (Hoffmann et al., 2022) as

$$\mathcal{L}(N, D) = c + \frac{a}{N^{\alpha}} + \frac{b}{D^{\beta}}. \tag{1}$$

The power-law formula is composed of three distinct terms that characterize the intrinsic entropy of data, constraints of the model, and limitations in the training data. The term c represents the minimum possible error intrinsic to the data. The remaining two terms are suboptimality terms, which address the limitations in function representation owing to the size of the model and in data signified by the number of tokens. In the limit, with infinite data and model size, the loss is reduced to c.

Mixture of Experts. For MoE Transformer-based models, (Clark et al., 2022) formulated the final loss for a constant dataset size D of 130B tokens, allowing for variations in the number of experts E, as:

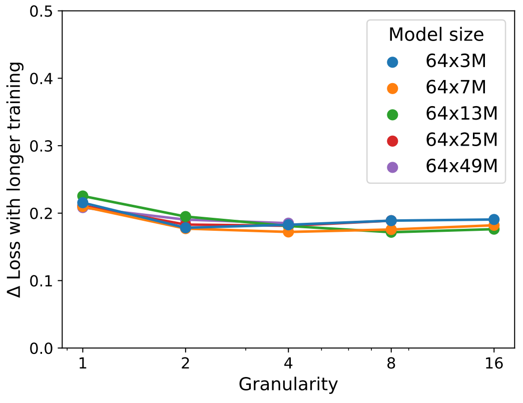
$$\mathcal{L}(N, E) = \left(\frac{10^{d/a}}{N}\right)^{a} \left(\frac{1}{E}\right)^{b + c \log N}. \tag{2}$$

However, this result has a notable limitation as it can be applied only to the original dataset size. The scalability and effectiveness are constrained in this scenario because it is crucial to align the number of training samples with the available computational resources for optimal use. As per (Kaplan et al., 2020) and (Hoffmann et al., 2022), maintaining a constant dataset size while scaling up the neural network size leads to undertraining, resulting in a model that does not perform to its full potential.

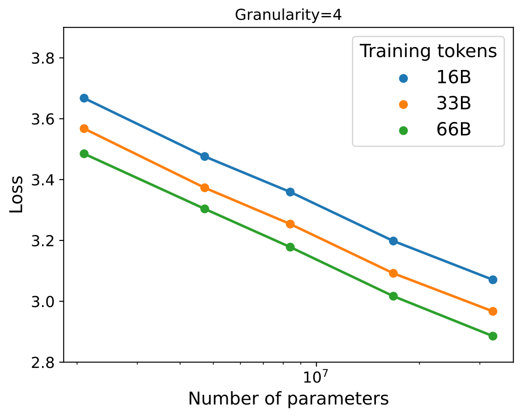
4. Granularity

As described in Section 3, in the standard setting, the inner dimension of each expert network $d_{\text{expert}} = d_{\text{ff}}$, the same size as the feed-forward layer of the base model.

In this work, we suggest an alternative approach where the hidden dimension of the expert is not necessarily set to



(a)



(b)

Figure 3. **(a)** The difference in the loss between training for 16B and 65B tokens for all model sizes and granularity values. The model size is reported as the expansion rate and the number of active parameters. **(b)** The impact of varying the number of parameters N on the loss for fixed granularity G. For other granularity values, see Appendix I.

mirror that of the standard feed-forward layer. Instead, it can be adjusted to a value that is the most effective. This approach allows the configuration of MoE to be articulated in terms of two key hyperparameters: *granularity* (G) and *expansion rate* (R). In the following parts of this work, we will also use the term *active* parameters to refer to the non-embedding parameters used to produce output for a single token, except routing. The number of active parameters is denoted as N_{act}.

Let d_{expert} be the hidden dimension of a single expert. Granularity is defined as

$$G = \frac{d_{\text{ff}}}{d_{\text{expert}}}.$$

In other words, granularity denotes the multiplier factor for the change in the size of an expert from the original standard model, defined as $G = 1$. In this work, we investigate $G > 1$ where experts are smaller than in the standard layer.

Note that increasing granularity does not affect the number of active parameters since, as G increases, the number of experts that process the token grows proportionally to G. That is, for granularity G, a token is routed to G fine-grained experts, keeping the number of active parameters constant. See Fig. 2 for visualization.

We then define the *expansion rate*, which describes the increase in the number of parameters from a standard transformer layer to an MoE layer. Given that, N_{MoE} and N_{ff} denote the total number of parameters in an MoE layer excluding routing and the standard feed-forward layer, respectively. The expansion rate R is then defined as

$$R = \frac{N_{\text{MoE}}}{N_{\text{ff}}}.$$

Expansion rate can also be seen as the total number of parameters in an MoE layer compared to active parameters. The relationship between the number of experts (N_{expert}), the expansion rate, and the granularity is described by the following equation:

$$N_{\text{expert}} = G \cdot R. \tag{3}$$

For non-granular models, i.e., $G = 1$, the expansion rate is equal to the number of experts.

Intuitively, increasing granularity for a given expansion rate gives the model more flexibility in mapping datapoints to experts, potentially improving performance. We incorporate the notion of granularity into our scaling laws in Section 5. The discussion about practical tradeoffs in changing this parameter is given in Section 6.

5. Scaling Laws

Granularity determines changes in the architecture of MoE. In this section, we answer a central question of this work: whether the granular MoE models follow scaling laws and, if so, how granularity affects them. Thus, we aim to derive a parametric scaling law for predicting the final loss value $\mathcal{L}$ based on granularity G, total number of non-embedding parameters N, and number of training tokens D.

We run over 100 experiments on the decoder-only Transformer architecture, with each feed-forward component replaced by a Mixture of Experts layer. Those experiments involve training models with sizes ranging from 129M to 3.7B parameters across different training durations, from 16B to 130B tokens. We consider logarithmically spaced values of granularity between 1 and 16. To constrain the search

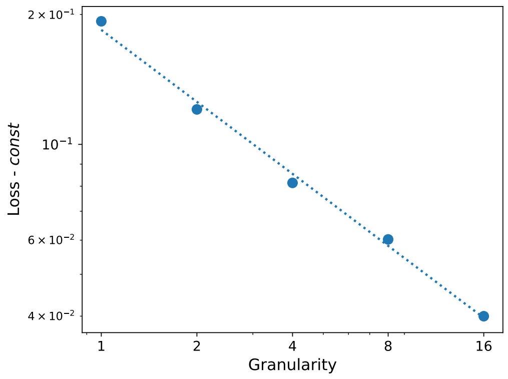


Figure 4. We plot the effect of G on $\mathcal{L}_{N,D}(G)$ for constant N and D. Both axes are in the log-scale. The results suggest the linear relationship between $\log(G)$ and $\log(\mathcal{L}-c)$. The given values are $N = 64 \times 25M$, $D = 16B$, $const = 3.12$. The plots for additional values of N and D can be found in Appendix I.

space, $R = 64$ is fixed, following the recommendations of (Clark et al., 2022). In addition, we also run experiments with dense Transformers to compare their performance with MoE. The details of all architectures, the training procedure, and hyperparameter choices are described in detail in Appendix A.

In the subsequent part of this paper, we will use the notation $R \times N_{\text{act}}$ to describe a MoE model with N_{act} active parameters and expansion rate R.

5.1. Power Law With Respect to Granularity

We first answer the question of whether granular models follow the scaling laws. In Figure 4, we can notice that increasing granularity results in a lower loss. The returns follow approximately an exponential pattern, converging to a positive constant. The empirical relationship given by Figure 4 suggests, for given N and D, the following power-law dependence of loss $\mathcal{L}_{D,G}$ on a varying granularity G, parametrized by $g_{N,D}$, $\gamma_{N,D}$, and $h_{N,D}$,

$$\mathcal{L}_{N,D}(G) = \frac{g_{N,D}}{G^{\gamma_{N,D}}} + h_{N,D}. \tag{4}$$

5.2. Scaling the Model and Dataset Size

As outlined in Section 3.2, the power-law given by Eq. (1) consists of three terms that describe inherent data entropy and limitations in function representation and data. This derivation is independent of the architecture. In particular, the Eq. (1) also holds for constant granularity. Empirically, we observe a power law relationship in N and D analogous to that in dense models (see also Figure 1 in (Kaplan et al., 2020)), as depicted in Figure 3 for a fixed value of granularity. Furthermore, the validity of this functional form is verified by fit in Section 5.4.

Since we know that separate scaling laws are valid for given granularities, in the general form, the parameters in Eq. (1) can be dependent on the model's granularity:

$$\mathcal{L}_G(N, D) = c_G + \frac{a_G}{N^{\alpha_G}} + \frac{b_G}{D^{\beta_G}}. \tag{5}$$

5.3. The Form of the Joint Scaling Law

Following the above observation that models with constant granularity obey Chinchilla scaling laws given by Eq. (1), the key question arises as to how the general notion of granularity G can be incorporated into the joint scaling law. Moreover, the scaling law formula from Eq. (5) for constant N and D has to be representable by Eq. (4). This is because the former is a more general equation, encompassing shared hyper-parameters across all N, D, and G. It is anticipated to align with the latter, consisting of distinct power laws, each with specific parameters for different N and D values. Consequently, the objective is to identify a function that fulfills these criteria.

$$\begin{aligned}\mathcal{L}(N, D, G) = \quad \mathcal{L}_{N,D}(G) \quad &= \quad \mathcal{L}_G(N, D) \\ = \frac{g_{N,D}}{G^{\gamma_{N,D}}} + h_{N,D} &= c_G + \frac{a_G}{N^{\alpha_G}} + \frac{b_G}{D^{\beta_G}}\end{aligned} \tag{6}$$

In the subsequent sections, we aim to determine which of these parameters remain independent of G and identify their functional form. Additionally, we present some rationale for the structure of our formula.

Lower Bound. Consider the limit of Eq. (5) for N and D growing to infinity:

$$\lim_{\substack{N\to\infty \\ D\to\infty}} \mathcal{L}(N, D, G) = c_G. \tag{7}$$

with the constant term c_G dependent on granularity.

This dependence is contradictory to the fact that the term captures the inherent entropy of the dataset, as also defined in (Hoffmann et al., 2022) appendix D.2 *'the minimal loss achievable for next-token prediction on the full distribution P, a.k.a the "entropy of natural text."'*.

The lower bound of the achievable loss for training bigger models on more samples should not depend on the architecture since it is a function of a dataset, not of a model. Therefore, the parameter $c_G = c$ is constant for all granularities.

Granularity and Number of Tokens D. As seen in Figure 3(a), the benefit of training a model on a larger dataset is almost the same for each granularity value. This suggests that there is no interaction between D and G. Therefore, we can assume that

$$\frac{b_G}{D^{\beta_G}} = \frac{b}{D^{\beta}}. \tag{8}$$

Granularity and Model Size N. We consider α to be a constant that describes how the function scales with N. In this work, we assume polynomial functional forms that rule out the potential dependency of α on G given the form of Eq. 4. Therefore, the only element dependent on G is a_G:

$$\mathcal{L}(N, D, G) = c + \left(\frac{g}{G^{\gamma}} + a\right)\frac{1}{N^{\alpha}} + \frac{b}{D^{\beta}}. \tag{9}$$

Finally, one could consider omitting the constant a in the equation above, and it would still reduce to (4) for constant N and D. However, this would mean that a model with infinite granularity and a small number of active parameters can achieve the perfect perplexity of the lower bound. We think that MoE sparse model should not exceed the performance of its dense counterpart matched by a total number of parameters and with all of them activated. This means that constant a can act as a marginal improvement from granularity.

5.4. Fitting the Parametric Scaling Law

Subsequently, we fit parameters in (9) to describe the scaling of MoE. For comparison, we also perform fitting for dense transformer given by (1). Similarly to (Hoffmann et al., 2022), we use Huber loss (Huber, 1964), with $\delta = 0.1$. The optimization is performed using the BFGS algorithm. We include a weight decay of $5e-4$ to enhance generalization. We start with fitting parameters in (9) and then find architecture-dependent coefficients α, β, a and b in (1). The values are presented in Table 1. We depict the fit of the equation in Figure 5. We generally observe a good fit, with RMSE $= 0.015$.

Table 1. Values of the fitted coefficients.

Model	a	α	b	β	g	γ	c
MoE	18.1	0.115	30.8	0.147	2.1	0.58	0.47
Dense	16.3	0.126	26.7	0.127	-	-	0.47

We validate the stability of the fit by excluding the top 20% of models with the lowest perplexity and finding the coefficients based on the remaining experiments. We observe that the formula remains almost unchanged in this scenario (see Table 6 in Appendix C). The validation RMSE is 0.019. Results are depicted in Figure 6.

5.5. MoE Scaling Properties

Comparing the part of the formula that approximates underfitting (that is, dependent on training tokens) in MoE ($30.8D^{-0.147}$) and Transformer ($26.7D^{-0.127}$), we can infer that MoE models need longer training to perform competitively but scale better after reaching that point. Nonetheless, this moment may still precede the compute-optimal for both models. On the other hand, we can see that the exponent on dense models $\alpha = -0.126$ scales better with a total number of parameters than the MoE counterpart $\alpha = -0.115$. This should not be surprising since dense models use all parameters on each token contrary to MoE, which gains a computational advantage by activating only a subset of them. Therefore, the fair comparison of the performance has to take into account FLOPs used by each model type. In the next section, we find compute-optimal granularity for a given FLOP budget.

6. Optimal Allocation of Computational Budget

The goal of this section is to find optimal N, D, G for a given computational budget F. This can be done by solving the following optimization problem,

$$\begin{aligned} \underset{N,D,G}{\text{minimize}} \quad & \mathcal{L}(N, D, G) \\ \text{subject to} \quad & \text{FLOPs}(N, D, G) = F. \end{aligned}$$

6.1. Computational Cost of Granularity

It is important to acknowledge that increasing granularity can lead to some challenges in training the model, namely higher computational and communication costs and a larger memory footprint.

The main component responsible for higher costs is the increase in routing operations due to a larger pool of granular experts. This increase is proportional to the value of G. For standard, non-granular MoE models ($G = 1$), the routing overhead still exists, although it has been considered negligible.

Taking into account the routing operation overhead, the number of used FLOPs F is described by the following formula:

$$F = (12{d_{\text{model}}}^2 c_f + d_{\text{model}} R G c_r) \cdot D \cdot n_{\text{blocks}}, \tag{10}$$

given expansion rate R, granularity G, and constants that denote FLOPs per active parameter ratio, respectively, within routing (c_r) and within the rest of the network (c_f). The term $12{d_{\text{model}}}^2$ is the number of active parameters within a transformer block, while $d_{\text{model}} R G c_r$ is the number of active parameters within a routing network. The in-depth analysis of constants c_r and c_f can be found in Appendix G.

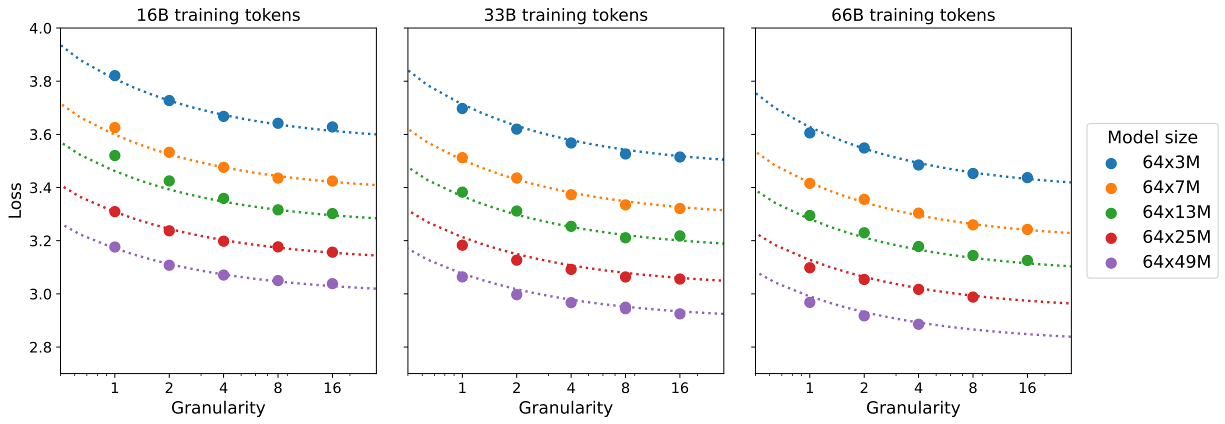


Figure 5. We present the fit of the scaling law compared to experimental results.

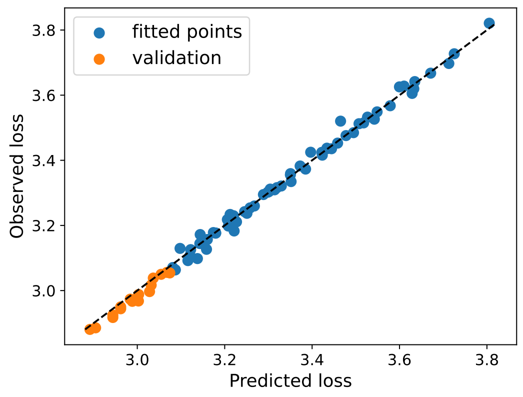


Figure 6. Validation of the fit.

We exclude embedding and unembedding from the FLOPs calculations, following (Hoffmann et al., 2022).

Observe that, in contrast to scenarios where routing operations are omitted, the FLOPs calculation that incorporates routing overhead relies on both d_{model} and n_{blocks}. Consequently, an additional condition is required to determine the scaling of d_{model} and n_{blocks} in relation to an increase in N, the number of parameters. It is noted that minor variations in the depth-to-width ratio are not significant (Kaplan et al., 2020). Following this analysis, we opt to adopt the assumption that $d_{\text{model}} = 64 n_{\text{blocks}}$.

The total number of parameters in the feed-forward layer, excluding the routing matrix, is $2Rd_{\text{ff}}d_{\text{model}} = 8R{d_{\text{model}}}^2$, and $4{d_{\text{model}}}^2$ in attention (key, query, value, and output projection). This results in the following formula for $N = {d_{\text{model}}}^2 \cdot (8R+4) \cdot n_{\text{blocks}}$.

6.2. Compute Optimal Formula

Putting all together we need to solve the following optimization problem, given F,

$$\begin{aligned}
\underset{N,D,G}{\text{minimize}} \quad & \mathcal{L}(N, D, G) \\
\text{subject to} \quad & F = (12{d_{\text{model}}}^2 c_f + d_{\text{model}} R G c_r) \cdot D \cdot n_{\text{blocks}} \\
& N = d_{\text{model}}^2 \cdot (8R+4) \cdot n_{\text{layers}}, \\
& d_{\text{model}} = 64 \cdot n_{\text{layers}}.
\end{aligned}$$

All these constraints are reducible to a one-dimensional optimization problem, which is, however, hard to solve analytically. Therefore we approximate the solution using Brent's method (Brent, 1971). The results of this optimization for varying FLOPs budgets are plotted in Figure 1 while the optimal configurations of parameters for selected model sizes are presented in Table 2. To validate the uncertainty of these predictions, we follow (Hoffmann et al., 2022) and calculate the 10th and 90th percentiles estimated via bootstrapping data (see Appendix D for the detailed results).

6.3. MoE is Always More Efficient

Contrary to the results from (Clark et al., 2022), in Figure 1 we can see, that Mixture-of-Experts can be always considered more efficient than dense Transformers, regardless of the model size. According to our previous observations from Section 5.5, MoE models scale better with optimal training. However, for short training schedules, they may underperform dense models. This means that for constant training time and increasing model size, there exists a point where both models will become very under-trained, in which scenario dense models surpass MoE. This shows why in (Clark et al., 2022), where varying the number of training tokens has not been considered, MoE was predicted to be underperforming for models bigger than $1T$. However, when all

Table 2. Compute optimal training hyper-parameters for MoE models. Optimal N and D follow approximately similar relation to these of (Hoffmann et al., 2022) for active parameters around the range of $1B$ to $10B$ parameters, requiring comparably longer training for smaller models and shorter for bigger ones. Note that, this also considers optimal granularity and its FLOPs cost.

N	D	G	FLOPs	Loss
64 x 100M	4.37B	8	2.95e+18	3.133
64 x 1B	28.94B	16	1.93e+20	2.491
64 x 3B	72.90B	16	1.41e+21	2.245
64 x 7B	137.60B	32	6.46e+21	2.076
64 x 70B	941.07B	32	4.16e+23	1.694
64 x 300B	2.96T	64	5.69e+24	1.503
64 x 1T	7.94T	64	4.97e+25	1.367

training hyper-parameters N, D, G are properly selected to be compute-optimal for each model, the gap between dense and sparse models only increases as we scale.

7. Discussion

Extreme Granularity. In Section 5, we argue that model performance improves with increasing granularity. This postulate largely aligns with the empirical findings of our study. Nonetheless, at exceedingly high granularity levels, such as $G = 64$ in models characterized by $d_{\text{model}} = 256$ and $R = 64$, there is an observable decline in performance. This phenomenon is particularly evident in scenarios where the number of parameters in the routing mechanism exceeds active parameters in actual experts. Additionally, as described in Section 6, the utility of such high granularity is predominantly restricted to models of substantial size. In alignment with the principles outlined in (Hoffmann et al., 2022), this research focuses more on findings that can be broadly applied rather than delving into the specific details of these corner-case situations. However, it is hypothesized that the efficiency of models with significantly high granularity could be potentially enhanced through careful expert initialization or modifications to the routing algorithm. These ideas are set aside to be investigated in future studies.

Varying Expansion Rate. In this study, due to computational resources constraint, we focus on $R = 64$, as recommended by (Clark et al., 2022). This value of R was also used for the largest models in other works (Du et al., 2022; Zhou et al., 2022) and the best-performing configuration in (Fedus et al., 2022). Nonetheless, we acknowledge the importance of considering different expansion rates, as different levels of R may be chosen based on factors like the target size of the model in memory. Therefore, in Appendix E, we present the results of the study for $R = 16$ and show that the main findings of this work are still valid in such cases.

Including R in the Formula. Another possible advancement would be to unify all of the factors N, D, G and R in one formula. While this would open the possibility of studying the relationships between coefficients in more detail, it would also be hard to practically recommend the optimal configuration in such a scenario using only FLOPs. This is because larger values of R typically lead to better performance but also incur additional memory requirements. Therefore, the choice of expansion rate may be heavily dependent on the available hardware configuration. We leave a detailed study of these factors for future work.

Modeling the Cost of Granularity. It is important to note that the exact estimation of the training cost of MoE models is dependent on the training setup, hardware, and implementation. Specifically, increasing G can lead to higher transfer costs, depending on the adopted model of distributed training. Therefore, the precise selection of hyperparameters should be made considering these factors. In this work, we model the cost of operations using FLOPs, which is common in the Scaling Laws literature (Kaplan et al., 2020; Hoffmann et al., 2022; Frantar et al., 2023). Additionally, we would like to note that in our setup, we observe significant gains of granular models measured as wall-clock time needed to achieve given perplexity (see Appendix H for an example).

8. Conclusions

This study introduces a novel hyperparameter, granularity (G), and underscores the significance of adjusting it for optimizing the efficiency of experts within MoE models. A central finding of this research is that a standard granularity of $G = 1$ is suboptimal across a broad range of FLOPs, leading to the recommendation of using higher granularity values to enhance MoE model performance and efficiency. Simultaneously, this work emphasizes the importance of varying training duration for compute-optimal settings. Consequently, both granularity and variable training length are incorporated into new scaling laws. These laws confidently demonstrate that MoE models consistently outperform dense transformers in terms of efficiency and scaling. This work not only sheds new light on the scaling laws applicable to MoE models but also provides practical guidance for improving computational efficiency in large language models. The insights are critical for the development and optimization of large-scale language models, marking a significant advancement in the field.

Reproducibility

The full code used to run experiments in this work is open-sourced under Apache License 2.0. The repository is available on GitHub, github.com/llm-random/llm-random .

Impact statement

This paper presents work whose goal is to advance the field of Machine Learning in general and language modeling in particular, especially with respect to scaling the models. There are many potential societal consequences of our work, none of which we feel must be specifically highlighted here.

Acknowledgements

We would like to express sincere gratitude to Piotr Miłoś and Tomasz Trzciński for valuable feedback and to Aleksandra Weglarz for her help with graphic design.

This work was funded by IDEAS NCBR, which also provided significant computational resources, a supportive research environment, and direction. The research was supported by PL-Grid infrastructure (grants PLG/2023/016148 and PLG/2024/017060). Sebastian Jaszczur was partially supported by the Polish National Science Centre grant 2022/45/N/ST6/02222. Marek Cygan was partially supported by an NCBiR grant POIR.01.01.01-00-0392/17-00. Piotr Sankowski was partially supported by ERC PoC grant 101082299-EXALT.

Contributions

We would like to highlight and list the individual contributions of the authors in this project.

Jakub Krajewski implemented fine-grained MoE, ran experiments, and oversaw the course of the project. Jan Ludziejewski designed and implemented the scaling laws, also optimized and tuned the fine-grained MoE implementation. Kamil Adamczewski provided significant advice on many aspects of the project. Maciej Pióro experimented with the block design and, with Michał Krutul, provided considerable technical support. Szymon Antoniak, Kamil Ciebiera, Krystian Król, and Tomasz Odrzygóźdź contributed to the project and the engineering in various ways. Marek Cygan, along with Piotr Sankowski, provided high-level scientific advice. Sebastian Jaszczur came up with the initial idea, started the project, and supervised it while setting the research direction and leading experiments and analyses.

References

Agostinelli, A., Denk, T. I., Borsos, Z., Engel, J., Verzetti, M., Caillon, A., Huang, Q., Jansen, A., Roberts, A., Tagliasacchi, M., Sharifi, M., Zeghidour, N., and Frank, C. Musiclm: Generating music from text, 2023.

Brent, R. P. An algorithm with guaranteed convergence for finding a zero of a function. *Comput. J.*, 14:422–425, 1971. URL `https://api.semanticscholar.org/CorpusID:10312755`.

Brown, T. B., Mann, B., Ryder, N., Subbiah, M., Kaplan, J., Dhariwal, P., Neelakantan, A., Shyam, P., Sastry, G., Askell, A., Agarwal, S., Herbert-Voss, A., Krueger, G., Henighan, T., Child, R., Ramesh, A., Ziegler, D. M., Wu, J., Winter, C., Hesse, C., Chen, M., Sigler, E., Litwin, M., Gray, S., Chess, B., Clark, J., Berner, C., McCandlish, S., Radford, A., Sutskever, I., and Amodei, D. Language models are few-shot learners, 2020.

Chowdhery, A., Narang, S., Devlin, J., Bosma, M., Mishra, G., Roberts, A., Barham, P., Chung, H. W., Sutton, C., Gehrmann, S., Schuh, P., Shi, K., Tsvyashchenko, S., Maynez, J., Rao, A., Barnes, P., Tay, Y., Shazeer, N., Prabhakaran, V., Reif, E., Du, N., Hutchinson, B., Pope, R., Bradbury, J., Austin, J., Isard, M., Gur-Ari, G., Yin, P., Duke, T., Levskaya, A., Ghemawat, S., Dev, S., Michalewski, H., Garcia, X., Misra, V., Robinson, K., Fedus, L., Zhou, D., Ippolito, D., Luan, D., Lim, H., Zoph, B., Spiridonov, A., Sepassi, R., Dohan, D., Agrawal, S., Omernick, M., Dai, A. M., Pillai, T. S., Pellat, M., Lewkowycz, A., Moreira, E., Child, R., Polozov, O., Lee, K., Zhou, Z., Wang, X., Saeta, B., Diaz, M., Firat, O., Catasta, M., Wei, J., Meier-Hellstern, K., Eck, D., Dean, J., Petrov, S., and Fiedel, N. Palm: Scaling language modeling with pathways, 2022.

Clark, A., de las Casas, D., Guy, A., Mensch, A., Paganini, M., Hoffmann, J., Damoc, B., Hechtman, B., Cai, T., Borgeaud, S., van den Driessche, G., Rutherford, E., Hennigan, T., Johnson, M., Millican, K., Cassirer, A., Jones, C., Buchatskaya, E., Budden, D., Sifre, L., Osindero, S., Vinyals, O., Rae, J., Elsen, E., Kavukcuoglu, K., and Simonyan, K. Unified scaling laws for routed language models, 2022.

Dai, D., Deng, C., Zhao, C., Xu, R. X., Gao, H., Chen, D., Li, J., Zeng, W., Yu, X., Wu, Y., Xie, Z., Li, Y. K., Huang, P., Luo, F., Ruan, C., Sui, Z., and Liang, W. Deepseekmoe: Towards ultimate expert specialization in mixture-of-experts language models, 2024.

Du, N., Huang, Y., Dai, A. M., Tong, S., Lepikhin, D., Xu, Y., Krikun, M., Zhou, Y., Yu, A. W., Firat, O., Zoph, B., Fedus, L., Bosma, M., Zhou, Z., Wang, T., Wang, Y. E., Webster, K., Pellat, M., Robinson, K., Meier-Hellstern, K., Duke, T., Dixon, L., Zhang, K., Le, Q. V., Wu, Y., Chen, Z., and Cui, C. Glam: Efficient scaling of language models with mixture-of-experts, 2022.

Faiz, A., Kaneda, S., Wang, R., Osi, R., Sharma, P., Chen, F., and Jiang, L. Llmcarbon: Modeling the end-to-end carbon footprint of large language models, 2024.

Fedus, W., Zoph, B., and Shazeer, N. Switch transformers: Scaling to trillion parameter models with simple and efficient sparsity. *The Journal of Machine Learning Research*, 23(1):5232–5270, 2022.

Frantar, E., Riquelme, C., Houlsby, N., Alistarh, D., and Evci, U. Scaling laws for sparsely-connected foundation models, 2023.

Ghorbani, B., Firat, O., Freitag, M., Bapna, A., Krikun, M., Garcia, X., Chelba, C., and Cherry, C. Scaling laws for neural machine translation, 2021.

Henighan, T., Kaplan, J., Katz, M., Chen, M., Hesse, C., Jackson, J., Jun, H., Brown, T. B., Dhariwal, P., Gray, S., Hallacy, C., Mann, B., Radford, A., Ramesh, A., Ryder, N., Ziegler, D. M., Schulman, J., Amodei, D., and McCandlish, S. Scaling laws for autoregressive generative modeling, 2020.

Hochreiter, S. and Schmidhuber, J. Long short-term memory. *Neural computation*, 9(8):1735–1780, 1997.

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

Huber, P. J. Robust Estimation of a Location Parameter. *The Annals of Mathematical Statistics*, 35(1):73 – 101, 1964. doi: 10.1214/aoms/1177703732. URL https://doi.org/10.1214/aoms/1177703732.

Jiang, A. Q., Sablayrolles, A., Mensch, A., Bamford, C., Chaplot, D. S., de las Casas, D., Bressand, F., Lengyel, G., Lample, G., Saulnier, L., Lavaud, L. R., Lachaux, M.-A., Stock, P., Scao, T. L., Lavril, T., Wang, T., Lacroix, T., and Sayed, W. E. Mistral 7b, 2023.

Jiang, A. Q., Sablayrolles, A., Roux, A., Mensch, A., Savary, B., Bamford, C., Chaplot, D. S., de las Casas, D., Hanna, E. B., Bressand, F., Lengyel, G., Bour, G., Lample, G., Lavaud, L. R., Saulnier, L., Lachaux, M.-A., Stock, P., Subramanian, S., Yang, S., Antoniak, S., Scao, T. L., Gervet, T., Lavril, T., Wang, T., Lacroix, T., and Sayed, W. E. Mixtral of experts, 2024.

Kaplan, J., McCandlish, S., Henighan, T., Brown, T. B., Chess, B., Child, R., Gray, S., Radford, A., Wu, J., and Amodei, D. Scaling laws for neural language models, 2020.

Lepikhin, D., Lee, H., Xu, Y., Chen, D., Firat, O., Huang, Y., Krikun, M., Shazeer, N., and Chen, Z. Gshard: Scaling giant models with conditional computation and automatic sharding, 2020.

Lewis, M., Bhosale, S., Dettmers, T., Goyal, N., and Zettlemoyer, L. Base layers: Simplifying training of large, sparse models, 2021.

Liu, Z. L., Dettmers, T., Lin, X. V., Stoyanov, V., and Li, X. Towards a unified view of sparse feed-forward network in pretraining large language model, 2023.

Loshchilov, I. and Hutter, F. Decoupled weight decay regularization, 2019.

Puigcerver, J., Riquelme, C., Mustafa, B., and Houlsby, N. From sparse to soft mixtures of experts, 2023.

Radford, A., Narasimhan, K., Salimans, T., and Sutskever, I. Improving language understanding by generative pretraining. 2018a.

Radford, A., Wu, J., Child, R., Luan, D., Amodei, D., and Sutskever, I. Language models are unsupervised multitask learners. 2018b. URL https://d4mucfpksywv.cloudfront.net/better-language-models/language-models.pdf.

Rae, J. W., Borgeaud, S., Cai, T., Millican, K., Hoffmann, J., Song, F., Aslanides, J., Henderson, S., Ring, R., Young, S., Rutherford, E., Hennigan, T., Menick, J., Cassirer, A., Powell, R., van den Driessche, G., Hendricks, L. A., Rauh, M., Huang, P.-S., Glaese, A., Welbl, J., Dathathri, S., Huang, S., Uesato, J., Mellor, J., Higgins, I., Creswell, A., McAleese, N., Wu, A., Elsen, E., Jayakumar, S., Buchatskaya, E., Budden, D., Sutherland, E., Simonyan, K., Paganini, M., Sifre, L., Martens, L., Li, X. L., Kuncoro, A., Nematzadeh, A., Gribovskaya, E., Donato, D., Lazaridou, A., Mensch, A., Lespiau, J.-B., Tsimpoukelli, M., Grigorev, N., Fritz, D., Sottiaux, T., Pajarskas, M., Pohlen, T., Gong, Z., Toyama, D., de Masson d'Autume, C., Li, Y., Terzi, T., Mikulik, V., Babuschkin, I., Clark, A., de Las Casas, D., Guy, A., Jones, C., Bradbury, J., Johnson, M., Hechtman, B., Weidinger, L., Gabriel, I., Isaac, W., Lockhart, E., Osindero, S., Rimell, L., Dyer, C., Vinyals, O., Ayoub, K., Stanway, J., Bennett, L., Hassabis, D., Kavukcuoglu, K., and Irving, G. Scaling language models: Methods, analysis & insights from training gopher, 2022.

Raffel, C., Shazeer, N., Roberts, A., Lee, K., Narang, S., Matena, M., Zhou, Y., Li, W., and Liu, P. J. Exploring the limits of transfer learning with a unified text-to-text transformer, 2023.

Roller, S., Sukhbaatar, S., Szlam, A., and Weston, J. Hash layers for large sparse models, 2021.

Scao, T. L., Fan, A., Akiki, C., Pavlick, E., Ilić, S., Hesslow, D., Castagné, R., Luccioni, A. S., Yvon, F., Gallé, M., Tow, J., Rush, A. M., Biderman, S., Webson, A., Ammanamanchi, P. S., Wang, T., Sagot, B., Muennighoff, N., del Moral, A. V., Ruwase, O., Bawden, R., Bekman, S., McMillan-Major, A., Beltagy, I., Nguyen, H., Saulnier, L., Tan, S., Suarez, P. O., Sanh, V., Laurençon, H., Jernite, Y., Launay, J., Mitchell, M., Raffel, C., Gokaslan, A., Simhi, A., Soroa, A., Aji, A. F., Alfassy, A., Rogers, A., Nitzav, A. K., Xu, C., Mou, C., Emezue, C., Klamm, C., Leong, C., van Strien, D., Adelani, D. I., Radev, D., Ponferrada, E. G., Levkovizh, E., Kim, E., Natan, E. B., Toni, F. D., Dupont, G., Kruszewski, G., Pistilli, G., Elsahar, H., Benyamina, H., Tran, H., Yu, I., Abdulmumin, I., Johnson, I., Gonzalez-Dios, I., de la Rosa, J., Chim, J., Dodge, J., Zhu, J., Chang, J., Frohberg, J., Tobing, J., Bhattacharjee, J., Almubarak, K., Chen, K., Lo, K., Werra, L. V., Weber, L., Phan, L., allal, L. B., Tanguy, L., Dey, M., Muñoz, M. R., Masoud, M., Grandury, M., Šaško, M., Huang, M., Coavoux, M., Singh, M., Jiang, M. T.-J., Vu, M. C., Jauhar, M. A., Ghaleb, M., Subramani, N., Kassner, N., Khamis, N., Nguyen, O., Espejel, O., de Gibert, O., Villegas, P., Henderson, P., Colombo, P., Amuok, P., Lhoest, Q., Harliman, R., Bommasani, R., López, R. L., Ribeiro, R., Osei, S., Pyysalo, S., Nagel, S., Bose, S., Muhammad, S. H., Sharma, S., Longpre, S., Nikpoor, S., Silberberg, S., Pai, S., Zink, S., Torrent, T. T., Schick, T., Thrush, T., Danchev, V., Nikoulina, V., Laippala, V., Lepercq, V., Prabhu, V., Alyafeai, Z., Talat, Z., Raja, A., Heinzerling, B., Si, C., Taşar, D. E., Salesky, E., Mielke, S. J., Lee, W. Y., Sharma, A., Santilli, A., Chaffin, A., Stiegler, A., Datta, D., Szczechla, E., Chhablani, G., Wang, H., Pandey, H., Strobelt, H., Fries, J. A., Rozen, J., Gao, L., Sutawika, L., Bari, M. S., Al-shaibani, M. S., Manica, M., Nayak, N., Teehan, R., Albanie, S., Shen, S., Ben-David, S., Bach, S. H., Kim, T., Bers, T., Fevry, T., Neeraj, T., Thakker, U., Raunak, V., Tang, X., Yong, Z.-X., Sun, Z., Brody, S., Uri, Y., Tojarieh, H., Roberts, A., Chung, H. W., Tae, J., Phang, J., Press, O., Li, C., Narayanan, D., Bourfoune, H., Casper, J., Rasley, J., Ryabinin, M., Mishra, M., Zhang, M., Shoeybi, M., Peyrounette, M., Patry, N., Tazi, N., Sanseviero, O., von Platen, P., Cornette, P., Lavallée, P. F., Lacroix, R., Rajbhandari, S., Gandhi, S., Smith, S., Requena, S., Patil, S., Dettmers, T., Baruwa, A., Singh, A., Cheveleva, A., Ligozat, A.-L., Subramonian, A., Névéol, A., Lovering, C., Garrette, D., Tunuguntla, D., Reiter, E., Taktasheva, E., Voloshina, E., Bogdanov, E., Winata, G. I., Schoelkopf, H., Kalo, J.-C., Novikova, J., Forde, J. Z., Clive, J., Kasai, J., Kawamura, K., Hazan, L., Carpuat, M., Clinciu, M., Kim, N., Cheng, N., Serikov, O., Antverg, O., van der Wal, O., Zhang, R., Zhang, R., Gehrmann, S., Mirkin, S., Pais, S., Shavrina, T., Scialom, T., Yun, T., Limisiewicz, T., Rieser, V., Protasov, V., Mikhailov, V., Pruksachatkun, Y., Belinkov, Y., Bamberger, Z., Kasner, Z., Rueda, A., Pestana, A., Feizpour, A., Khan, A., Faranak, A., Santos, A., Hevia, A., Unldreaj, A., Aghagol, A., Abdollahi, A., Tammour, A., HajiHosseini, A., Behroozi, B., Ajibade, B., Saxena, B., Ferrandis, C. M., McDuff, D., Contractor, D., Lansky, D., David, D., Kiela, D., Nguyen, D. A., Tan, E., Baylor, E., Ozoani, E., Mirza, F., Ononiwu, F., Rezanejad, H., Jones, H., Bhattacharya, I., Solaiman, I., Sedenko, I., Nejadgholi, I., Passmore, J., Seltzer, J., Sanz, J. B., Dutra, L., Samagaio, M., Elbadri, M., Mieskes, M., Gerchick, M., Akinlolu, M., McKenna, M., Qiu, M., Ghauri, M., Burynok, M., Abrar, N., Rajani, N., Elkott, N., Fahmy, N., Samuel, O., An, R., Kromann, R., Hao, R., Alizadeh, S., Shubber, S., Wang, S., Roy, S., Viguier, S., Le, T., Oyebade, T., Le, T., Yang, Y., Nguyen, Z., Kashyap, A. R., Palasciano, A., Callahan, A., Shukla, A., Miranda-Escalada, A., Singh, A., Beilharz, B., Wang, B., Brito, C., Zhou, C., Jain, C., Xu, C., Fourrier, C., Periñán, D. L., Molano, D., Yu, D., Manjavacas, E., Barth, F., Fuhrimann, F., Altay, G., Bayrak, G., Burns, G., Vrabec, H. U., Bello, I., Dash, I., Kang, J., Giorgi, J., Golde, J., Posada, J. D., Sivaraman, K. R., Bulchandani, L., Liu, L., Shinzato, L., de Bykhovetz, M. H., Takeuchi, M., Pàmies, M., Castillo, M. A., Nezhurina, M., Sänger, M., Samwald, M., Cullan, M., Weinberg, M., Wolf, M. D., Mihaljcic, M., Liu, M., Freidank, M., Kang, M., Seelam, N., Dahlberg, N., Broad, N. M., Muellner, N., Fung, P., Haller, P., Chandrasekhar, R., Eisenberg, R., Martin, R., Canalli, R., Su, R., Su, R., Cahyawijaya, S., Garda, S., Deshmukh, S. S., Mishra, S., Kiblawi, S., Ott, S., Sang-aroonsiri, S., Kumar, S., Schweter, S., Bharati, S., Laud, T., Gigant, T., Kainuma, T., Kusa, W., Labrak, Y., Bajaj, Y. S., Venkatraman, Y., Xu, Y., Xu, Y., Xu, Y., Tan, Z., Xie, Z., Ye, Z., Bras, M., Belkada, Y., and Wolf, T. Bloom: A 176b-parameter open-access multilingual language model, 2023.

Shazeer, N., Mirhoseini, A., Maziarz, K., Davis, A., Le, Q., Hinton, G., and Dean, J. Outrageously large neural networks: The sparsely-gated mixture-of-experts layer, 2017.

Shazeer, N., Cheng, Y., Parmar, N., Tran, D., Vaswani, A., Koanantakool, P., Hawkins, P., Lee, H., Hong, M., Young, C., Sepassi, R., and Hechtman, B. Mesh-tensorflow: Deep learning for supercomputers, 2018.

Touvron, H., Lavril, T., Izacard, G., Martinet, X., Lachaux, M.-A., Lacroix, T., Rozière, B., Goyal, N., Hambro, E., Azhar, F., Rodriguez, A., Joulin, A., Grave, E., and Lample, G. Llama: Open and efficient foundation language models, 2023a.

Touvron, H., Martin, L., Stone, K., Albert, P., Almahairi, A., Babaei, Y., Bashlykov, N., Batra, S., Bhargava, P., Bhosale, S., Bikel, D., Blecher, L., Ferrer, C. C., Chen, M., Cucurull, G., Esiobu, D., Fernandes, J., Fu, J., Fu, W., Fuller, B., Gao, C., Goswami, V., Goyal, N., Hartshorn, A., Hosseini, S., Hou, R., Inan, H., Kardas, M., Kerkez, V., Khabsa, M., Kloumann, I., Korenev, A., Koura, P. S., Lachaux, M.-A., Lavril, T., Lee, J., Liskovich, D., Lu, Y., Mao, Y., Martinet, X., Mihaylov, T., Mishra, P., Molybog, I., Nie, Y., Poulton, A., Reizenstein, J., Rungta, R., Saladi, K., Schelten, A., Silva, R., Smith, E. M., Subramanian, R., Tan, X. E., Tang, B., Taylor, R., Williams, A., Kuan, J. X., Xu, P., Yan, Z., Zarov, I., Zhang, Y., Fan, A., Kambadur, M., Narang, S., Rodriguez, A., Stojnic, R., Edunov, S., and Scialom, T. Llama 2: Open foundation and fine-tuned chat models, 2023b.

Yin, S., Fu, C., Zhao, S., Li, K., Sun, X., Xu, T., and Chen, E. A survey on multimodal large language models, 2023.

Zhou, Y., Lei, T., Liu, H., Du, N., Huang, Y., Zhao, V., Dai, A., Chen, Z., Le, Q., and Laudon, J. Mixture-of-experts with expert choice routing, 2022.

Zhou, Y., Du, N., Huang, Y., Peng, D., Lan, C., Huang, D., Shakeri, S., So, D., Dai, A., Lu, Y., Chen, Z., Le, Q., Cui, C., Laundon, J., and Dean, J. Brainformers: Trading simplicity for efficiency, 2023.

A. Architecture and Training Setup

All of the models considered in this work are decoder-only Transformers trained on the C4 dataset (Raffel et al., 2023). We use GPT2 tokenizer (Radford et al., 2018a). Each batch consists of 0.5M tokens packed into 2048 sequences. Our optimizer is AdamW (Loshchilov & Hutter, 2019), with a weight decay of 0.1. In each training run, we use the maximum learning rate of $2\text{e}{-4}$, with linear warmup for 1% steps and cosine decay to $2\text{e}{-5}$. To improve stability, we initialize weights using the truncated normal distribution with reduced scale, as advised in (Fedus et al., 2022). The models are trained using mixed precision; we always keep the attention mechanism and router in high precision. We assume the *infinite data* regime, as the number of training tokens for any of the runs in less than the number of tokens in the corpus. We follow (Hoffmann et al., 2022) and perform our analysis on the smoothed training loss.

In MoE, we use the Expert Choice routing algorithm, as it guarantees a balanced expert load without tuning additional hyperparameters. To maintain compatibility with autoregressive language modeling, we apply the recipe described in (Zhou et al., 2022): tokens are grouped by position across different sequences. The group size is always set to 256. We match the number of FLOPs for MoE and dense models with the same d_{model} (meaning we activate an average of $8d^2_{\text{model}}$ parameters per token in each MoE layer). In the router, softmax is performed over the expert dimension, while we choose tokens over the token dimension, as this leads to the best performance (as opposed to performing softmax over the token dimension). We put an additional layer normalization before the output of MoE layer. This gives a small improvement for standard MoE, but is crucial for the performance of models with $G > 1$.

Table 3 and Table 4 list the considered architecture and training variants for dense and MoE models, respectively.

Table 3. Architecture and training variants (MoE models).

#parameters (nonemb)	d_{model}	n_{blocks}	n_{heads}	D (in #tokens)	G
64x3M	256	4	4	16B, 33B, 66B	1, 2, 4, 8, 16
64x7M	384	4	6	16B, 33B, 66B	1, 2, 4, 8, 16
64x13M	512	4	8	16B, 33B, 66B	1, 2, 4, 8, 16
64x13M	512	4	8	130B	1, 2, 4
64x25M	512	8	8	16B, 33B,	1, 2, 4, 8, 16
64x25M	512	8	8	66B	1, 2, 4, 8
64x49M	640	10	10	16B, 33B	1, 2, 4, 8, 16
64x49M	640	10	10	66B	1, 2, 4
64x85M	768	12	12	33B	1, 2, 4

Table 4. Architecture and training variants (dense models).

#parameters (nonemb)	d_{model}	n_{blocks}	n_{heads}	D (in #tokens)
3M	256	4	4	16B, 24B, 33B, 66B
6M	256	8	4	16B, 24B, 33B, 66B
13M	512	4	8	16B, 24B, 33B, 66B
25M	512	8	8	16B, 24B, 33B, 66B
49M	640	10	10	16B, 24B, 33B, 66B
85M	768	12	12	16B, 33B

B. Comparison to (Clark et al., 2022) with Effective Parameter Count Curve

One of the main conclusions of our work, that MoE is always more efficient, is contradictory to the thesis from (Clark et al., 2022). Their work focuses on a fixed dataset length $D = 130B$, and they define the effective parameter count of any MoE model as the size of a dense model that achieves the same perplexity as a given MoE with a certain number of active parameters. Subsequently, they claim that as the number of parameters grows, this effective parameter count curve crosses with the active parameter curve, which means that using MoE will lead to worse performance in the future. Despite this incompatibility, we do not claim that their experiments and extrapolations are not valid. If we use our fitted scaling laws with a fixed number of training tokens D, the resulting *effective parameter count* curve (which can be properly defined only for a fixed D) would indeed cross at some parameter count in the same way as it did in (Clark et al., 2022).

This crossing point of *effective parameter count* curve, that is, a number of parameters where dense Transformers surpass MoE models for a given number of training tokens D, can be calculated by determining where our scaling laws for MoE and for dense models cross, solving the following equation for N, given D: $0.47 + (\frac{2.1}{G^{0.58}} + 18.1)N^{-0.115} + 30.8D^{-0.147} = 0.47 + 16.3N^{-0.126} + 26.7D^{-0.127}$ This solves for:

Table 5. Crossing point for MoE Effective Parameter Count Curve with Dense

D	10B	130B	1T
(N) MoE/dense Crossing Point	251B	1.9T	10T

This crossing point will be placed further away for each increase in D.

Importantly, the lines will never cross if we train each of these models in a compute-optimal manner using the same computational budget (Figure 1). Our results show that regular dense models perform better than MoE only when both are severely undertrained, as in the extrapolation from (Clark et al., 2022), where a comparison is made between $1T$ parameter models trained on $130B$ steps. Moreover, in industry settings, it is common to overtrain LLMs and extremely rare to undertrain them, and our scaling laws indicate even better performance of MoE models in an overtrained regime.

C. Validation of the Scaling Law

In this section, we provide coefficients of the scaling law fitted with 20% of datapoints with the lowest perplexity excluded for the purpose of validation.

Table 6. Values of the fitted coefficients.

Model	a	α	b	β	g	γ	c
MoE	17.6	0.114	26.7	0.140	2.07	0.570	0.472

D. Reliability of Compute Optimal Formula

In this section, we assess the stability of our predictions presented in Section 6.1. Similarly to (Hoffmann et al., 2022) we calculate the 10th and 90th percentiles estimated via bootstrapping data (80% of the data is sampled 100 times). See Table 7 for the details.

E. Varying Expansion Rate

In this section, we provide results for $R = 16$. The training procedure is the same as described in App. A. The models considered in this part are listed in Table 8.

We fit Eq. 9 using the same procedure as described in Section 5.4. The results are detailed in Table 9.

Using the coefficients and FLOPs calculation formulas, we can derive the compute optimal training parameters. The results are presented in Table 10.

We can observe that similarly to the case when $R = 64$, larger compute budgets imply larger optimal values of G. Note

Table 7. 10th and 90th percentiles estimated via bootstraping data.

N	D	G
64 x 100M	(2.97B, 5.98B)	(8, 8)
64 x 1B	(21.17B, 40.73B)	(16, 16)
64 x 3B	(50.20B, 105.88B)	(16, 32)
64 x 7B	(101.06B, 205.40B)	(32, 32)
64 x 70B	(638.49B, 1.59T)	(32, 64)
64 x 300B	(1.99T, 5.62T)	(64, 64)
64 x 1T	(5.29T, 16.87T)	(64, 64)

Table 8. Architecture and training variants (MoE models).

#parameters (nonemb)	d_{model}	n_{blocks}	n_{heads}	D (in #tokens)	G
64x3M	256	4	4	8B, 16B, 33B	1, 2, 4, 8, 16
64x7M	256	8	4	8B, 16B, 33B	1, 2, 4, 8, 16
64x13M	512	4	8	8B, 16B, 33B	1, 2, 4, 8, 16
64x13M	512	4	8	66B	1, 2, 4
64x25M	512	8	8	8B, 16B, 33B	1, 2, 4, 8, 16
64x49M	640	10	10	8B	1, 2, 4, 8, 16

Table 9. Values of the fitted coefficients.

Model	a	α	b	β	g	γ	c
MoE ($R = 16$)	19.64	0.124	57.07	0.169	1.18	0.986	0.472

Table 10. 10th and 90th percentiles estimated via bootstrapping data for $R = 16$.

N	D	G
16 x 100M	(10.29B, 17.73B)	(8 , 16)
16 x 1B	(53.74B, 103.54B)	(16, 32)
16 x 3B	(106.22B, 261.04B)	(16, 32)
16 x 7B	(177.65B, 511.43B)	(16, 32)
16 x 70B	(721.60B, 3.22T)	(32, 64)
16 x 300B	(1.73T, 10.69T)	(32, 64)
16 x 1T	(3.60T, 28.22T)	(32, 128)

that the values for 10th and 90th percentiles form larger intervals in this case, as in this part we run a smaller number of experiments and keep shorter training durations. However, we believe that this preliminary study forms a valuable addition to the results in the main part.

F. Choosing Granularity

Table 7 (for expansion rate 64) and Table 9 (for expansion rate 16) list the optimal granularity values for various compute budgets. Fig. 1 (a) presents an alternative presentation for $R = 64$. We observe that the optimal granularity values are generally similar between different expansion rates. We can generally provide the following guidelines:

- The standard value of $G = 1$ is almost never optimal.
- For a reasonable default value of G refer to Table 7 and Table 9 . For constant N or D, one can calculate optimal G and the trade-off between predicted loss and training FLOPs directly from our scaling laws using the coefficients from Table 6.
- The exact optimal value of granularity may differ slightly from our setup, but based on other works on scaling laws, we expect only slight differences.

G. FLOPs Constants

The number of FLOPs F used in Transformer training, considering the routing operation overhead in MoE, can be described by the following formula:

$$F = (12{d_{\text{model}}}^2 c_f + d_{\text{model}} E G c_r) \cdot n_{\text{tokens}} \cdot n_{\text{blocks}} \tag{11}$$

Following (Hoffmann et al., 2022), we assume c_f to be 6. This is interpreted as 6 FLOPs for each pair of an active parameter (in linear projection) and a processed token. The breakdown of operations is as follows:

- During the forward pass, 2 operations (single multiplication and single addition) are used to compute the matrix multiplication of an input and linear projection.
- During the backward pass, 2 operations are used to compute gradients wrt. the input.
- During the backward pass, 2 operations are used to compute gradients wrt. the weights of linear projection.

In our work, we have assumed the routing constant, c_r, to be 14, with the breakdown presented below. The exact number of operations may depend on the implementation of routing, but it will be between 6 and 20. However, our main conclusions of the paper are resistant to different assumptions of this constant.

- During the forward pass, 2 operations are used to compute the expert logits based on an input and ”routing linear projection”.
- During the backward pass, 2 operations are used to compute gradients for ”routing linear projection” wrt. the input.
- During the backward pass, 2 operations are used to compute gradients for ”routing linear projection” wrt. the weights of linear projection.
- During the forward pass, 2 operations are used to route input tokens to chosen experts.
- During the forward pass, 2 operations are used to route expert outputs to chosen tokens and multiply those outputs by the routing score.
- During the backward pass, 2 operations are used to route gradients from output tokens to experts.
- During the backward pass, 2 operations are used to route gradients from experts to input tokens.

Similarly to the calculation of FLOPs for c_f, FLOPs come in pairs as each multiplication is followed by an addition (used to accumulate outputs or gradients).

H. Measuring Wall-clock Time

In this section, we provide an example of training curves for models with different levels of granularity, measured in terms of wall-clock training time on NVIDIA A100 GPU. We can see that the model with $G = 8$ achieves the best performance in this case.

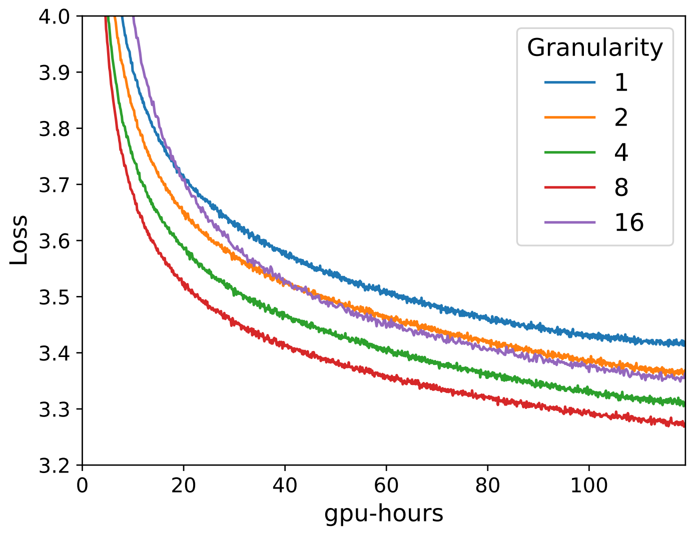
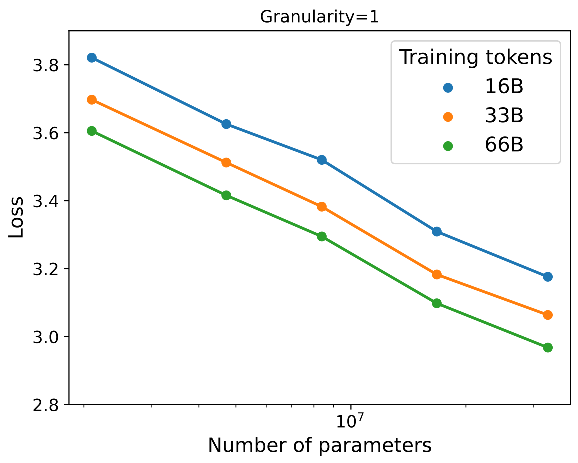
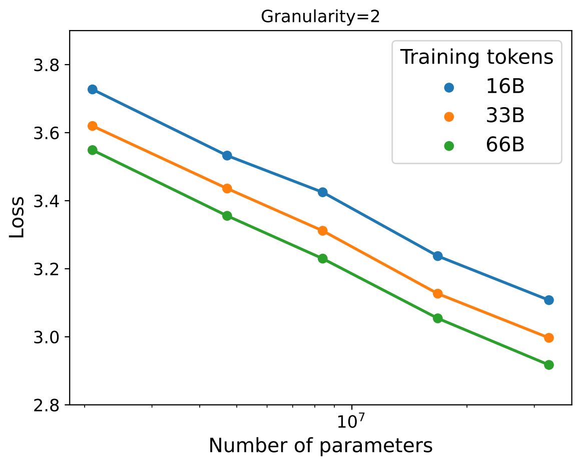


Figure 7. Training loss curves for model with $N = 64 \times 7M$, $D = 66B$ tokens.

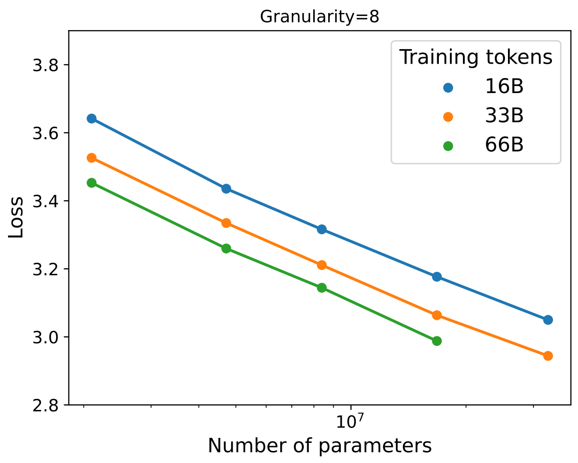
I. Additional Visualizations



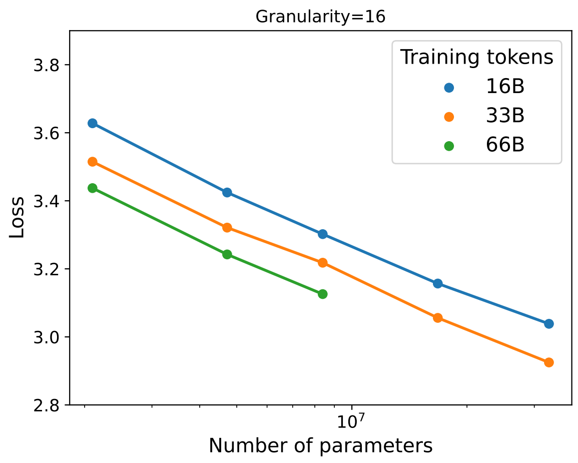
(a)



(b)

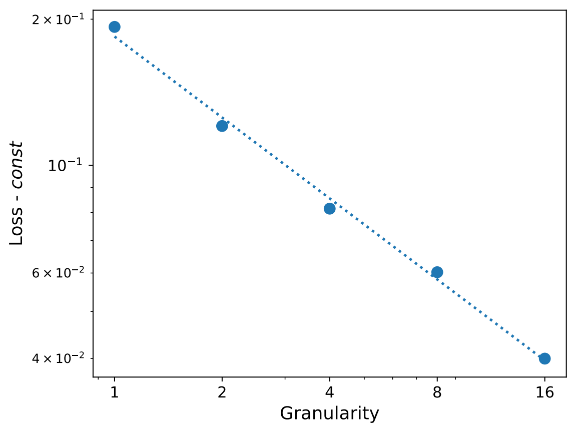


(c)

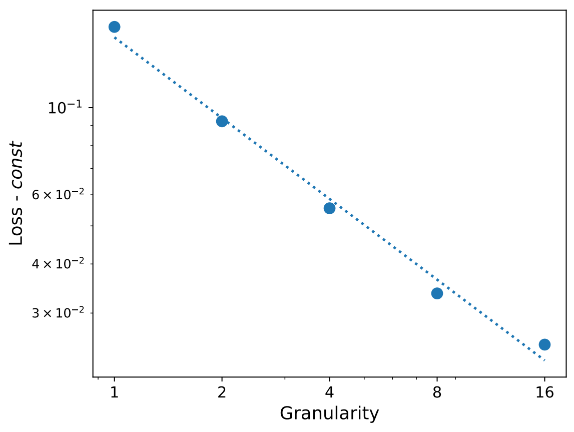


(d)

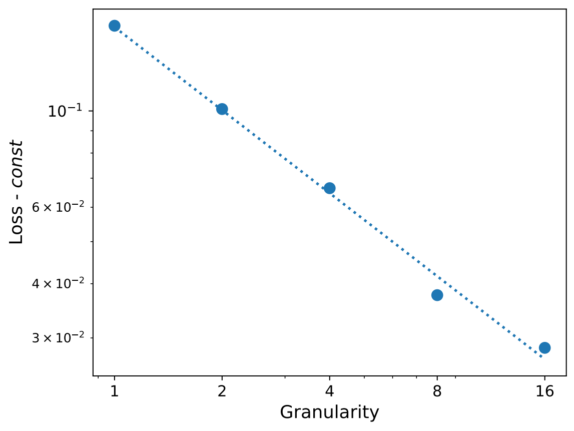
Figure 8. Illustration of scaling N and D for constant granularity value of: **(a)** $G = 1$ **(b)** $G = 2$ **(c)** $G = 8$ **(d)** $G = 16$.



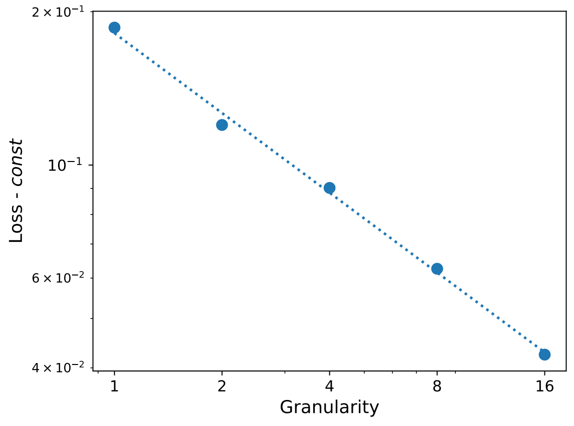
(a)



(b)



(c)



(d)

Figure 9. Illustration of scaling granularity when N, D are fixed for: **(a)** $N = 64 \times 25M$, $D = 16B$, $const = 3.12$ **(b)** $N = 64 \times 49M$, $D = 16B$, $const = 3.02$ **(c)** $N = 64 \times 25M$, $D = 32B$, $const = 3.03$ **(d)** $N = 64 \times 49M$, $D = 32B$, $const = 2.88$