WIKIPEDIA
The Free Encyclopedia

Neural scaling law

In machine learning, a **neural scaling law** is a scaling law relating parameters of a family of neural networks.[1][2]

Introduction

In general, a neural model can be characterized by 4 parameters: size of the model, size of the training dataset, cost of training, error rate after training. Each of these four variables can be precisely defined into a real number, and they are empirically found to be related by simple statistical laws, called "scaling laws". These are usually written as N, D, C, L (number of parameters, dataset size, computing cost, loss).

Size of the model

In most cases, the size of the model is simply the number of parameters. However, one complication arises with the use of sparse models, such as mixture-of-expert models.[3] In sparse models, during every inference, only a fraction of the parameters are used. In comparison, most other kinds of neural networks, such as Transformer networks, always use all their parameters during every inference.

Size of the training dataset

The size of the training dataset is usually quantified by the number of data points it contains. Larger training datasets are typically preferred as they provide a richer and more diverse source of information for the model to learn from. This in turn can lead to improved generalization performance when the model is applied to unseen data.[4] However, increasing the size of the training dataset also increases the computational resources and time required for model training.

With the "pretrain, then finetune" method used in most large language models, there are two kinds of training dataset: the pretraining dataset and the finetuning dataset. Their sizes would have different effects on model performance. Generally, the finetuning dataset is less than 1% the size of pretraining dataset.[5]

In some cases, a small amount of high quality data suffices for finetuning, and more data does not improve performance.[5]

Cost of training

The cost of training is typically measured in terms of time (how long it takes to train the model) and computational resources (how much processing power and memory are required to train the model). It's important to note that the cost of training can be significantly reduced with efficient training algorithms, optimized software libraries, and parallel computing on specialized hardware like GPUs or TPUs.

The cost of training a neural model is a function of several factors including the size of the model, the size of the training dataset, the complexity of the training algorithm, and the computational resources available.[4] In particular, doubling the training dataset does not necessarily double the cost of training, because one may train the model for several times over the same dataset (each being an "epoch").

Performance

The performance of a neural model is evaluated based on its ability to accurately predict the output given the input data. Common metrics for evaluating model performance include:[4]

- accuracy, precision, recall, and F1 score for classification tasks;

- mean squared error (MSE) or mean absolute error (MAE) for regression tasks;
- negative log-likelihood per token (logarithm of perplexity) for language modeling.
- Elo rating in a competition against other models, such as gameplay[6] or preference by a human judge[7]

Performance can be improved by using more data, larger models, different training algorithms, regularizing the model to prevent overfitting, and early stopping using a validation set.

Examples

(Hestness, Narang, et al, 2017)

The 2017 paper[2] is a common reference point for neural scaling laws fitted by statistical analysis on experimental data. Previous works before the 2000s, as cited in the paper, were either theoretical or orders of magnitude smaller in scale. Whereas previous works generally found the scaling exponent to scale like $L \propto D^{-\alpha}$, with $\alpha \in \{0.5, 1, 2\}$, the paper found that $\alpha \in [0.07, 0.35]$.

Of the factors they varied, only task can change the exponent α. Changing the architecture optimizers, regularizers, and loss functions, would only change the proportionality factor, not the exponent. For example, for the same task, one architecture might have $L = 1000D^{-0.3}$ while another might have $L = 500D^{-0.3}$. The also found that for a given architecture, the number of parameters necessary to reach lowest levels of loss, given a fixed dataset size, grows like $N \propto D^{\beta}$ for another exponent β.

They studied machine translation with LSTM ($\alpha \sim 0.13$), generative language modelling with LSTM ($\alpha \in [0.06, 0.09], \beta \approx 0.7$), ImageNet classification with ResNet ($\alpha \in [0.3, 0.5], \beta \approx 0.6$), and speech recognition ($\alpha \approx 0.3$).

(Henighan, Kaplan, et al, 2020)

A 2020 analysis [8] studied statistical relations between C, N, D, L over a wide range of values and found similar scaling laws, over the range of $N \in [10^3, 10^9]$, $C \in [10^{12}, 10^{21}]$, and over multiple modalities (text, video, image, text to image, etc.).[8]

In particular, the scaling laws it found are (Table 1 of [8]):

- For each modality, they fixed one of the two C, N, and varying the other one (D is varied along using $D = C/6N$), the achievable test loss satisfies

  $$L = L_0 + \left(\frac{x_0}{x}\right)^{\alpha}$$

  where x is the varied variable, and L_0, x_0, α are parameters to be found by statistical fitting. The parameter α is the most important one.

  - When N is the varied variable, α ranges from 0.037 to 0.24 depending on the model modality. This corresponds to the $\alpha = 0.34$ from the Chinchilla scaling paper.
  - When C is the varied variable, α ranges from 0.048 to 0.19 depending on the model modality. This corresponds to the $\beta = 0.28$ from the Chinchilla scaling paper.
- Given fixed computing budget, optimal model parameter count is consistently around

  $$N_{opt}(C) = \left(\frac{C}{5 \times 10^{-12}\text{petaFLOP-day}}\right)^{0.7} = 9.0 \times 10^{-7} C^{0.7}$$

  The parameter 9.0×10^{-7} varies by a factor of up to 10 for different modalities. The exponent parameter 0.7 varies from 0.64 to 0.75 for different modalities. This exponent corresponds to the ≈ 0.5 from the Chinchilla scaling paper.

- It's "strongly suggested" (but not statistically checked) that $D_{opt}(C) \propto N_{opt}(C)^{0.4} \propto C^{0.28}$. This exponent corresponds to the ≈ 0.5 from the Chinchilla scaling paper.

The scaling law of $L = L_0 + (C_0/C)^{0.048}$ was confirmed during the training of GPT-3 (Figure 3.1 [9]).

Chinchilla scaling (Hoffmann, et al, 2022)

One particular scaling law ("Chinchilla scaling") states that, for a large language model (LLM) autoregressively trained for one epoch, with a cosine learning rate schedule, we have:[10]

$$\begin{cases} C = C_0 N D \\ L = \frac{A}{N^\alpha} + \frac{B}{D^\beta} + L_0 \end{cases}$$

where the variables are

- C is the cost of training the model, in FLOPS.
- N is the number of parameters in the model.
- D is the number of tokens in the training set.
- L is the average negative log-likelihood loss per token (nats/token), achieved by the trained LLM on the test dataset.
  - L_0 represents the loss of an ideal generative process on the test data
  - $\frac{A}{N^\alpha}$ captures the fact that a Transformer language model with N parameters underperforms the ideal generative process
  - $\frac{B}{D^\beta}$ captures the fact that the model trained on D tokens underperforms the ideal generative process

and the statistical parameters are

- $C_0 = 6$, meaning that it costs 6 FLOPs per parameter to train on one token. This is estimated by Kaplan et al.[11] Note that training cost is much higher than inference cost, as training entails both forward and backward passes, whereas inference costs 1 to 2 FLOPs per parameter to infer on one token.
- $\alpha = 0.34, \beta = 0.28, A = 406.4, B = 410.7, L_0 = 1.69$.

Although Besiroglu et. al.[12] claims that the statistical estimation is slightly off, and should be $\alpha = 0.35, \beta = 0.37, A = 482.01, B = 2085.43, L_0 = 1.82$.

The statistical laws were fitted over experimental data with $N \in [7 \times 10^7, 1.6 \times 10^{10}], D \in [5 \times 10^9, 5 \times 10^{11}], C \in [10^{18}, 10^{24}]$.

Since there are 4 variables related by 2 equations, imposing 1 additional constraint and 1 additional optimization objective allows us to solve for all four variables. In particular, for any fixed C, we can uniquely solve for all 4 variables that minimizes L. This provides us with the optimal $D_{opt}(C), N_{opt}(C)$ for any fixed C:

$$N_{opt}(C) = G\left(\frac{C}{6}\right)^a, \quad D_{opt}(C) = G^{-1}\left(\frac{C}{6}\right)^b, \quad \text{where} \quad G = \left(\frac{\alpha A}{\beta B}\right)^{\frac{1}{\alpha+\beta}}, \quad a = \frac{\beta}{\alpha+\beta}, \text{and } b = \frac{\alpha}{\alpha+\beta}.$$

Plugging in the numerical values, we obtain the "Chinchilla efficient" model size and training dataset size, as well as the test loss achievable:

$$\begin{cases} N_{opt}(C) = 0.6\ C^{0.45} \\ D_{opt}(C) = 0.3\ C^{0.55} \\ L_{opt}(C) = 1070\ C^{-0.154} + 1.7 \end{cases}$$

Similarly, we may find the optimal training dataset size and training compute budget for any fixed model parameter size, and so on.

There are other estimates for "Chinchilla efficient" model size and training dataset size. The above is based on a statistical model of $L = \frac{A}{N^\alpha} + \frac{B}{D^\beta} + L_0$. One can also directly fit a statistical law for $D_{opt}(C), N_{opt}(C)$ without going through the detour, for which one obtains:

$$\begin{cases} N_{opt}(C) = 0.1\ C^{0.5} \\ D_{opt}(C) = 1.7\ C^{0.5} \end{cases}$$

or as tabulated:

$N_{opt}(C)$	C / FLOP	C / FLOPs of training Gopher	$D_{opt}(C)$
400 Million	1.92e+19	1/29968	8.0 Billion
1 Billion	1.21e+20	1/5706	20.2 Billion
10 Billion	1.23e+22	1/2819	205.1 Billion
67 Billion	5.76e+23	1	1.5 Trillion
175 Billion	3.85e+24	6.7	3.7 Trillion
280 Billion	9.90e+24	17.2	5.9 Trillion
520 Billion	3.43e+25	59.5	11.0 Trillion
1 Trillion	1.27e+26	221.3	21.2 Trillion
10 Trillion	1.30e+28	22515.9	216.2 Trillion

In simpler terms, the Chinchilla scaling law for training Transformer language models suggests that when given an increased budget (in FLOPs), to achieve compute-optimal, the number of model parameters (N) and the number of tokens for training the model (D) should scale in approximately equal proportions. This conclusion differs from the previous scaling law for neural language models,[11] which states that N should be scaled faster than D. The discrepancy arises from setting different cycle lengths for cosine learning rate schedulers. In estimating the Chinchilla scaling, the authors set the cycle length to be the same as the training steps, as experimental results indicate that larger cycles overestimate the loss of the models.

Beyond Chinchilla scaling

As Chinchilla scaling has been the reference point for many large-scaling training runs, there had been a concurrent effort to go "beyond Chinchilla scaling", meaning to modify some of the training pipeline in order to obtain the same loss with less effort, or deliberately train for longer than what is "Chinchilla optimal".

Usually, the goal is to make the scaling law exponent larger, which means the same loss can be trained for much less compute. For instance, filtering data can make the scaling law exponent larger.[13]

Another strand of research studies how to deal with limited data, as according to Chinchilla scaling laws, the training dataset size for the largest language models already approaches what is available on the internet. [14] found that augmenting the dataset with a mix of "denoising objectives" constructed from the dataset improves performance. [15] studies optimal scaling when all available data is already exhausted (such as in rare languages), so one must train multiple epoches over the same dataset (whereas Chinchilla scaling requires only one epoch). The Phi series of small language models were trained on textbook-like data generated by large language models, for which data is only limited by amount of compute available.[16]

Chinchilla optimality was defined as "optimal for training compute", whereas in actual production-quality models, there will be a lot of inference after training is complete. "Overtraining" during training means better performance during inference.[17] LLaMA models were overtrained for this reason. Subsequent studies discovered scaling laws in the overtraining regime, for dataset sizes up to 32x more than Chinchilla-optimal.[18]

Broken Neural Scaling Laws (BNSL)

A 2022 analysis[19] found that many scaling behaviors of artificial neural networks follow a smoothly broken power law functional form:

$$y = a + \left(bx^{-c_0}\right)\prod_{i=1}^{n}\left(1+\left(\frac{x}{d_i}\right)^{1/f_i}\right)^{-c_i * f_i}$$

in which x refers to the quantity being scaled (i.e. C, N, D, number of training steps, number of inference steps, or model input size) and y refers to the *downstream* (or upstream) performance evaluation metric of interest (e.g. prediction error, cross entropy, calibration error, AUROC, BLEU score percentage, F1 score, reward, Elo rating, solve rate, or FID score) in zero-shot, prompted, or fine-tuned settings. The parameters $a, b, c_0, c_1 \ldots c_n, d_1 \ldots d_n, f_1 \ldots f_n$ are found by statistical fitting.

On a log–log plot, when f_i is not too large and a is subtracted out from the y-axis, this functional form looks like a series of linear segments connected by arcs; the n transitions between the segments are called "breaks", hence the name *Broken Neural Scaling Laws (BNSL)*.

The scenarios in which the scaling behaviors of artificial neural networks were found to follow this functional form include large-scale vision, language, audio, video, diffusion, generative modeling, multimodal learning, contrastive learning, AI alignment, AI capabilities, robotics, out-of-distribution (OOD) generalization, continual learning, transfer learning, uncertainty estimation / calibration, out-of-distribution detection, adversarial robustness, distillation, sparsity, retrieval, quantization, pruning, fairness, molecules, computer programming/coding, math word problems, arithmetic, emergent abilities, double descent, supervised learning, unsupervised/self-supervised learning, and reinforcement learning (single agent and multi-agent).

The architectures for which the scaling behaviors of artificial neural networks were found to follow this functional form include ResNets, Transformers, MLPs, MLP-Mixers, Recurrent Neural Networks, Convolutional Neural Networks, Graph Neural Networks, U-Nets, Encoder-Decoder (and Encoder-only) (and Decoder-only) Models, Ensembles (and Non-Ensembles), MoE (Mixture of Experts) (and Non-MoE) Models, and Sparse Pruned (and Non-Sparse Unpruned) Models.

Other examples

Vision transformers

Vision transformers, similar to language transformers, exhibit scaling laws. A 2022 research trained vision transformers, with parameter counts $N \in [5 \times 10^6, 2 \times 10^9]$, on image sets of sizes $D \in [3 \times 10^7, 3 \times 10^9]$, for computing $C \in [0.2, 10^4]$ (in units of TPUv3-core-days).[20]

After training the model, it is finetuned on ImageNet training set. Let L be the error probability of the finetuned model classifying ImageNet test set. They found $\min_{N,D} L = 0.09 + \frac{0.26}{(C+0.01)^{0.35}}$.

Neural machine translation

Ghorbani, Behrooz et al.[21] studied scaling laws for neural machine translation (specifically, English as source, and German as target) in encoder-decoder Transformer models, trained until convergence on the same datasets (thus they did not fit scaling laws for computing cost C or dataset size D). They varied $N \in [10^8, 3.5 \times 10^9]$ They found three results:

- L is a scaling law function of N_E, N_D, where N_E, N_D are encoder and decoder parameter count. It is not simply a function of total parameter count $N = N_E + N_D$. The function has form $L\left(N_e, N_d\right) = \alpha\left(\frac{\bar{N}_e}{N_e}\right)^{p_e}\left(\frac{\bar{N}_d}{N_d}\right)^{p_d} + L_\infty$, where $\alpha, p_e, p_d, L_\infty, \bar{N}_e, \bar{N}_d$ are fitted parameters. They found that $N_d/N \approx 0.55$ minimizes loss if N is held fixed.
- L "saturates" (that is, it reaches L_∞) for smaller models when the training and testing datasets are "source-natural" than "target-natural". A "source-natural" data point means a pair of English-German sentences, and the model is asked to translate the English sentence into German, and the English sentence is written by a natural English writer, while the German sentence is translated from the English sentence by a machine translator.[22] To construct the two kinds of datasets, the authors collected natural English and German sentences online, then used machine translation to generate their translations.
- As models grow larger, models trained on source-original datasets can achieve low loss but bad BLEU score. In contrast, models trained on target-original datasets achieve low loss and good BLEU score in tandem (Figure 10, 11 [21]).

The authors hypothesize that source-natural datasets have uniform and dull target sentences, and so a model that is trained to predict the target sentences would quickly overfit.

[23] trained Transformers for machine translations with sizes $N \in [4 \times 10^5, 5.6 \times 10^7]$ on dataset sizes $D \in [6 \times 10^5, 6 \times 10^9]$. They found the Kaplan et al (2020)[11] scaling law applied to machine translation: $L(N, D) = \left[\left(\frac{N_C}{N}\right)^{\frac{\alpha_N}{\alpha_D}} + \frac{D_C}{D}\right]^{\alpha_D}$. They also found the BLEU score scaling as $BLEU \approx Ce^{-kL}$.

Transfer learning

Hernandez, Danny et al.[24] studied scaling laws for transfer learning in language models. They trained a family of Transformers in three ways:

- pretraining on English, finetuning on Python
- pretraining on an equal mix of English and Python, finetuning on Python
- training on Python

The idea is that pretraining on English should help the model achieve low loss on a test set of Python text. Suppose the model has parameter count N, and after being finetuned on D_F Python tokens, it achieves some loss L. We say that its "transferred token count" is D_T, if another model with the same N achieves the same L after training on $D_F + D_T$ Python tokens.

They found $D_T = 1.9e4(D_F)^{.18}(N)^{.38}$ for pretraining on English text, and $D_T = 2.1e5(D_F)^{.096}(N)^{.38}$ for pretraining on English and non-Python code.

References

1. Bahri, Yasaman; Dyer, Ethan; Kaplan, Jared; Lee, Jaehoon; Sharma, Utkarsh (2021-02-12). "Explaining Neural Scaling Laws". arXiv:2102.06701 (https://arxiv.org/abs/2102.06701) [cs.LG (https://arxiv.org/archive/cs.LG)].
2. Hestness, Joel; Narang, Sharan; Ardalani, Newsha; Diamos, Gregory; Jun, Heewoo; Kianinejad, Hassan; Patwary, Md Mostofa Ali; Yang, Yang; Zhou, Yanqi (2017-12-01). "Deep Learning Scaling is Predictable, Empirically". arXiv:1712.00409 (https://arxiv.org/abs/1712.00409) [cs.LG (https://arxiv.org/archive/cs.LG)].

3. Rajbhandari, Samyam; Li, Conglong; Yao, Zhewei; Zhang, Minjia; Aminabadi, Reza Yazdani; Awan, Ammar Ahmad; Rasley, Jeff; He, Yuxiong (2022-06-28). "DeepSpeed-MoE: Advancing Mixture-of-Experts Inference and Training to Power Next-Generation AI Scale" (https://proceedings.mlr.press/v162/rajbhandari22a.html). *Proceedings of the 39th International Conference on Machine Learning*. PMLR: 18332–18346. arXiv:2201.05596 (https://arxiv.org/abs/2201.05596).
4. Goodfellow, I., Bengio, Y., & Courville, A. (2016). Deep Learning. MIT Press.
5. Zhou, Chunting; Liu, Pengfei; Xu, Puxin; Iyer, Srini; Sun, Jiao; Mao, Yuning; Ma, Xuezhe; Efrat, Avia; Yu, Ping; Yu, Lili; Zhang, Susan; Ghosh, Gargi; Lewis, Mike; Zettlemoyer, Luke; Levy, Omer (2023-05-01). "LIMA: Less Is More for Alignment". arXiv:2305.11206 (https://arxiv.org/abs/2305.11206) [cs.CL (https://arxiv.org/archive/cs.CL)].
6. Jones, Andy L. (2021). "Scaling Scaling Laws with Board Games". arXiv:2104.03113 (https://arxiv.org/abs/2104.03113) [cs.LG (https://arxiv.org/archive/cs.LG)].
7. LMSYS Chatbot leaderboard (https://huggingface.co/spaces/lmsys/chatbot-arena-leaderboard)
8. Sam, Henighan, Tom Kaplan, Jared Katz, Mor Chen, Mark Hesse, Christopher Jackson, Jacob Jun, Heewoo Brown, Tom B. Dhariwal, Prafulla Gray, Scott Hallacy, Chris Mann, Benjamin Radford, Alec Ramesh, Aditya Ryder, Nick Ziegler, Daniel M. Schulman, John Amodei, Dario McCandlish (2020-10-27). *Scaling Laws for Autoregressive Generative Modeling* (http://worldcat.org/oclc/1228442047). OCLC 1228442047 (https://www.worldcat.org/oclc/1228442047).
9. Brown, Tom B.; Mann, Benjamin; Ryder, Nick; Subbiah, Melanie; Kaplan, J.; Dhariwal, Prafulla; Neelakantan, Arvind; Shyam, Pranav; Sastry, Girish; Askell, Amanda; Agarwal, Sandhini; Herbert-Voss, Ariel; Krueger, Gretchen; Henighan, T.; Child, Rewon (2020-05-28). "Language Models are Few-Shot Learners". arXiv:2005.14165 (https://arxiv.org/abs/2005.14165) [cs.CL (https://arxiv.org/archive/cs.CL)].
10. Hoffmann, Jordan; Borgeaud, Sebastian; Mensch, Arthur; Buchatskaya, Elena; Cai, Trevor; Rutherford, Eliza; Casas, Diego de Las; Hendricks, Lisa Anne; Welbl, Johannes; Clark, Aidan; Hennigan, Tom; Noland, Eric; Millican, Katie; Driessche, George van den; Damoc, Bogdan (2022-03-29). "Training Compute-Optimal Large Language Models". arXiv:2203.15556 (https://arxiv.org/abs/2203.15556) [cs.CL (https://arxiv.org/archive/cs.CL)].
11. Kaplan, Jared; McCandlish, Sam; Henighan, Tom; Brown, Tom B.; Chess, Benjamin; Child, Rewon; Gray, Scott; Radford, Alec; Wu, Jeffrey; Amodei, Dario (2020). "Scaling Laws for Neural Language Models". *CoRR*. abs/2001.08361. arXiv:2001.08361 (https://arxiv.org/abs/2001.08361).
12. Besiroglu, Tamay; Erdil, Ege; Barnett, Matthew; You, Josh (2024-04-15), *Chinchilla Scaling: A replication attempt*, arXiv:2404.10102 (https://arxiv.org/abs/2404.10102)
13. Sorscher, Ben; Geirhos, Robert; Shekhar, Shashank; Ganguli, Surya; Morcos, Ari S. (2023-04-21), *Beyond neural scaling laws: beating power law scaling via data pruning*, arXiv:2206.14486 (https://arxiv.org/abs/2206.14486)
14. Tay, Yi; Wei, Jason; Chung, Hyung Won; Tran, Vinh Q.; So, David R.; Shakeri, Siamak; Garcia, Xavier; Zheng, Huaixiu Steven; Rao, Jinfeng (2022-11-16), *Transcending Scaling Laws with 0.1% Extra Compute*, arXiv:2210.11399 (https://arxiv.org/abs/2210.11399)
15. Muennighoff, Niklas; Rush, Alexander; Barak, Boaz; Le Scao, Teven; Tazi, Nouamane; Piktus, Aleksandra; Pyysalo, Sampo; Wolf, Thomas; Raffel, Colin A. (2023-12-15). "Scaling Data-Constrained Language Models" (https://proceedings.neurips.cc/paper_files/paper/2023/hash/9d89448b63ce1e2e8dc7af72c984c196-Abstract-Conference.html). *Advances in Neural Information Processing Systems*. **36**: 50358–50376. arXiv:2305.16264 (https://arxiv.org/abs/2305.16264).
16. Li, Yuanzhi; Bubeck, Sébastien; Eldan, Ronen; Del Giorno, Allie; Gunasekar, Suriya; Lee, Yin Tat (2023-09-11), *Textbooks Are All You Need II: phi-1.5 technical report*, arXiv:2309.05463 (https://arxiv.org/abs/2309.05463)
17. Sardana, Nikhil; Frankle, Jonathan (2023-12-31), *Beyond Chinchilla-Optimal: Accounting for Inference in Language Model Scaling Laws*, arXiv:2401.00448 (https://arxiv.org/abs/2401.00448)
18. Gadre, Samir Yitzhak; Smyrnis, Georgios; Shankar, Vaishaal; Gururangan, Suchin; Wortsman, Mitchell; Shao, Rulin; Mercat, Jean; Fang, Alex; Li, Jeffrey (2024-03-13), *Language models scale reliably with over-training and on downstream tasks*, arXiv:2403.08540 (https://arxiv.org/abs/2403.08540)
19. Caballero, Ethan; Gupta, Kshitij; Rish, Irina; Krueger, David (2022). "Broken Neural Scaling Laws". arXiv:2210.14891 (https://arxiv.org/abs/2210.14891) [cs.LG (https://arxiv.org/archive/cs.LG)].
20. Zhai, Xiaohua; Kolesnikov, Alexander; Houlsby, Neil; Beyer, Lucas (2022). "Scaling Vision Transformers" (https://openaccess.thecvf.com/content/CVPR2022/html/Zhai_Scaling_Vision_Transformers_CVPR_2022_paper.html). *CVPR*: 12104–12113.
21. Ghorbani, Behrooz; Firat, Orhan; Freitag, Markus; Bapna, Ankur; Krikun, Maxim; Garcia, Xavier; Chelba, Ciprian; Cherry, Colin (2021-09-01). "Scaling Laws for Neural Machine Translation". arXiv:2109.07740 (https://arxiv.org/abs/2109.07740) [cs.LG (https://arxiv.org/archive/cs.LG)].

22. Chen, Mia Xu; Firat, Orhan; Bapna, Ankur; Johnson, Melvin; Macherey, Wolfgang; Foster, George; Jones, Llion; Schuster, Mike; Shazeer, Noam; Parmar, Niki; Vaswani, Ashish; Uszkoreit, Jakob; Kaiser, Lukasz; Chen, Zhifeng; Wu, Yonghui (July 2018). "The Best of Both Worlds: Combining Recent Advances in Neural Machine Translation" (https://aclanthology.org/P18-1008). *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. Melbourne, Australia: Association for Computational Linguistics: 76–86. arXiv:1804.09849 (https://arxiv.org/abs/1804.09849). doi:10.18653/v1/P18-1008 (https://doi.org/10.18653%2Fv1%2FP18-1008).
23. Gordon, Mitchell A; Duh, Kevin; Kaplan, Jared (2021). "Data and Parameter Scaling Laws for Neural Machine Translation". *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*. Stroudsburg, PA, USA: Association for Computational Linguistics. pp. 5915–5922. doi:10.18653/v1/2021.emnlp-main.478 (https://doi.org/10.18653%2Fv1%2F2021.emnlp-main.478).
24. Hernandez, Danny; Kaplan, Jared; Henighan, Tom; McCandlish, Sam (2021-02-01). "Scaling Laws for Transfer". arXiv:2102.01293 (https://arxiv.org/abs/2102.01293) [cs.LG (https://arxiv.org/archive/cs.LG)].
