Improving Computer Vision efficiency and inference performance through meta-understanding and cognitive robustness techniques in the real world

Sergio Gabardo
Pan American School of Porto Alegre
2025sgabardo@panamerican.com.br

***Abstract*—This research article, recognizing the exponentially-increasing power requirements large AI models are having in the last couple of years, provides a model-agnostic approach to increase model accuracy, reduce overfitting and training requirements via SAM, as well as prune model size and distillate knowledge to reduce overparameterization. These three main objectives highlight the approaches analyzed in this paper for a more efficient and lower-latency image classification high-dimentional ML model.**

Note: This is not the final work. December 2025 version. Check for updates in the GitHub linked below.

I. Introduction

[1] classifies the task of *Artificial Intelligence* (AI) to "build intelligent entities," emphasizing its "truly universal field." *Machine Learning* (ML), a subset of AI, leveraged by the power of *Deep Learning* (DL), has seen astounding recent increases in influence, advancements, and general reach to the end consumers. Although current *algorithms* and *architectures* that of *chatbots* or *classification models*, when boiled down to their fundamentals, are not very different from that of past years, they have a giant distinction in how they are perceived and utilized by consumers. Take the *Transformer* Architecture that powers modern chatbots: it was already 5 years old when first made available to the general public through *ChatGPT* [2, 3].

Optical Character Recognition (OCR) [4], which falls on the umbrella *Computer Vision* (CV) field, widely utilized to convert images of car plates, printed numbers, or text into characters computers can interpret, has seen its first idealizations in the early 20th century (take Goldberg's 1927 *Statistical Machine* [5]), with formal modern idealizations emerging later that same century [6, 7], streamlined by the widespread use of computers, digitalization of information, and implementations of new algorithms [8, 9]. It becomes no doubt that, despite time passing by, advancements of ML, aiming to make computers as smart as possible [1], are not a new technology per se, but rather ideas that are present in the idealizations of automation, intrinsic to human consciousness [10].

This paper, henceforth, aims to analyze more efficient and accurate image-classification approaches in congruence with the recent astounding increase in energetical needs for training and running high-dimensional parameter spaces that of, but not limited to, *Large Language Models* (LLMs) and *Vision Transformers* (ViTs).

A. How are models trained?

A *neural network* (NN) takes an input in the form of a *feature vector*, which is fed into multiple *neurons* which randomly activate based on the input. A model can be thought of as a black box analogy:

$$\text{input} \rightarrow \boxed{\text{NN}} \rightarrow \text{output} \tag{1}$$

The first step to understand how models succeed (or fail) to learn is to think of ML models the same way as we think of teaching others: we may not be aware of how they are reaching a conclusion, yet if they consistently reach one, it may be assumed their process is consistently reaching appropriate results (hence a low error or mistake rate). If students, however, consistently gets good grades in Test A, yet gets a statistically significant lower grade in Test B of the same content, it may be inferred that the student has not learned the actual content of the test, but rather its non-generalizable structure—this is called *overfitting* in ML [11]. The notion of a student learning will henceforth be used, given the analogy's robustness and ability to capture well all the necessary concepts to be outlined in this paper. Training a ML algorithm, therefore, boils down to adapting teaching methods in order to conform to the best way students (models) learn.

Simplifying the calculation to a one-variable linear equation regression, there are two adjustable parameters: slope and y-intercept, and one output—the error. Fig. 1 demonstrates the evolution of a regression model as a "descent" into the valley. The valley in and of itself represents the optimal combination of parameters m and b to best fit the data.

There are some points that have to be cleared up prior to delving deeper, though: (i) this is a simplified model.

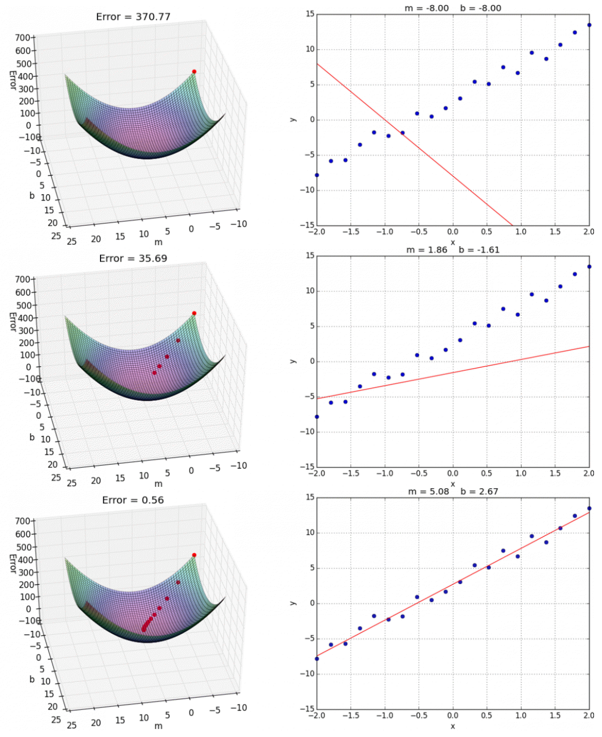

Fig. 1. Visualization of error minimization: The loss landscape evolution (left) and corresponding best-fit line adjustments (right) demonstrate the iterative optimization process [12].

It is clear to see what the answer is, but rather than two dimensions, current AI models such are already going into the trillions of parameters (i.e. dimensions); (ii) the error never, in a statistical scenario, reaches 0 due to sampling and data variability, and (iii) a valley may only be a *local minima*, not necessarily the *global* minima. The task in question is to minimize the chances of **not** reaching the optimal parameters.

With the scalar value from the *objective function* (OF), one may understand that the focus is to minimize it. This is where backpropagation comes in. By analyzing how much changing one parameter affects it and its following neurons up to the final output layer through Calculus' *Chain Rule*, a model is able to grasp how tuning different parameters evolves the **current model state** on the paraboloid shape provided by Fig. 1's loss lanscape, yet in a much higher-dimensional space.

It is additionally relevant to note that the evolution of the OF with respect to model training generally follows that of a decaying exponential function (2)

$$f(n) = A\exp(-b \cdot n) + C, \quad A,\, b,\, C \in \mathbb{R}^+ \tag{2}$$

where each value varies on a case-by-case basis depending on the evolution of the training and $n \in \mathbb{N}$ is the epoch number. C is the *steady state* (SS), that is, value the function approaches as $n \to \infty$. As the number of parameters of a function or regularization techniques applied increase, the SS tends to decrease given a better generalization over dataset D. However, every data set has an intrinsic entropy H, denoted $H(D)$ under the Shannon entropy formula [13].

The entropy of a dataset, often measured in bits per character (bpc), quantifies the average information content per character based on Shannon's formula [**?**]. For example, the entropy of the whole English language, calculated in 1951, is c. 1 bpc as the number of characters $N \to \infty$. Howbeit, contingent on all 26 characters being randomly arranged in random order (remaining some arbitrary meaning), the average amount of information is of $\log_2(26) \approx 4.7$ bpc. This comes to show us that although different methods of conveying information exist, a pattern is always needed, typically in the form of data-loss protection. This explains why QR codes can function even with parts obscured, and why languages can omit certain words in sentences, yet both remain comprehensible to computers and humans.

The relevance of understanding both the general model evolution pattern and the entropy of a dataset is to firstly understand that, initially, models learn quickly yet begin to stall once finding smaller-error minima is harder than in the beginning. Moreover, the error of a non-overfit model can never be smaller than the intrinsic variability of the dataset itself. Note the last frame of Fig. 1: neither is the best-fit line error or the MSE approaching zero, so much as they mathematically are not able to, at least with a line. The idea of the overparametrization of a dataset (as can be seen in Fig. 2) is a common issue in ML, which is be going to be further developed in section III. If Fig. 1 was to approach a zero-error model, more parameters would have to be in play, but more would not represent a better generalization of the dataset.

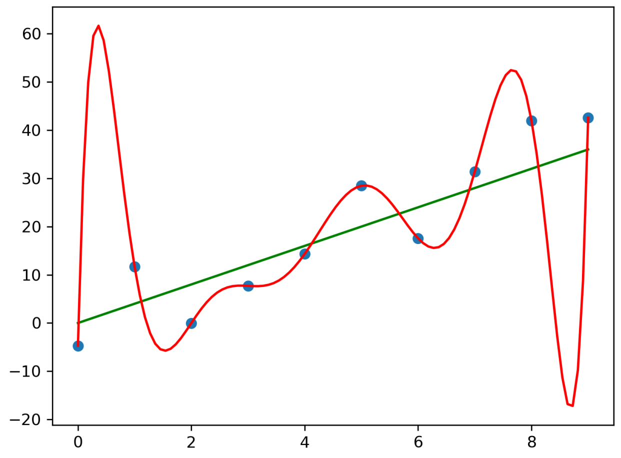

Fig. 2. Illustration of overfitting: A high-degree polynomial (red) excessively conforms to a roughly linear dataset (blue), contrasting with the simpler best-fit linear model (green) [14].

B. Backpropagation

Although many of the training AI does is based on more complex and distinct processes than mere linear regression, the idea remains the same: we have a variable that measures by how off is a prediction, and we have to minimize it (hence the widespread use of $\min_\theta$ in many ML equations, where θ is the parameter to be minimized). The specific tuning of NNs is via a process called *backpropagation*, where the *gradient* (slope) of the loss function is used to calculate the *vector field* (matching points of a graph with a vector) of the a function, such as that in Fig. 3. This is the opposite process of *forward propagation*: reaching conclusions given inputs.

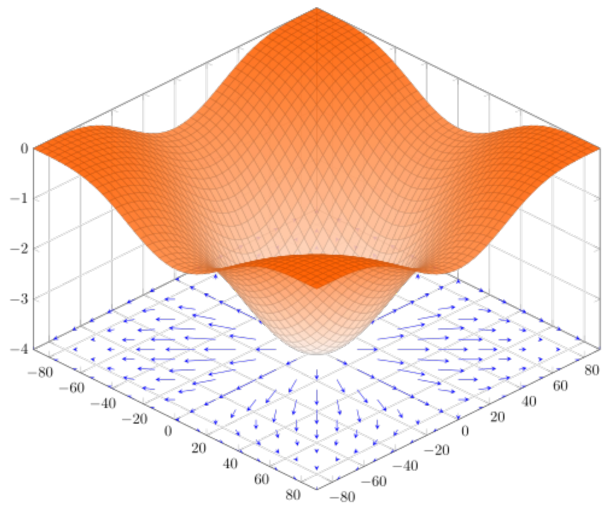

Fig. 3. Visualization of the gradient of the function $f(x, y) = -(\cos^2 x + \cos^2 y)^2$, showing the corresponding *vector field* on the bottom plane. The arrows represent the gradient's direction and magnitude, illustrating how the function's slope guides optimization processes like backpropagation [15].

In simple terms, the gradient at any point in a function returns the vector of steepest ascent at any point. To reach local minima, consequently, we must take the negative gradient of the cost function at our current parameters, with the direction representing the parameters we have to change, and the magnitude by how much we must do so. The issues present with standard approaches to learning, together with ways of minimizing them are further detailed in section III. A simplified formula (3) can be thought of as

$$\theta_{t+1} = \theta_t - \eta \nabla_\theta \mathcal{L}(\theta_t) \tag{3}$$

where θ is the set of model parameters, t is the epoch number (at what pass over the whole dataset the model is at), η is the learning rate (also known as the step size), ∇_θ is the gradient at the current parameters, and $\mathcal{L}$ is the loss function, taking the parameters as input.

II. Related Work

A. Other methodologies employed

All but one of the top 25 best image classifier models for at the CIFAR-100 dataset (90% accuracy and up) showed use of "extra training data" [16]. Given the focus of this paper is to (i) provide means of reduced energy usage and (ii) improve robustness, this is a great source of information, notably due to the availability of thorough methodologies and code repositories.

B. Other fields

1) Medical: In a research experimentation with positive-negative pneumonia–classification cases in patients given x-ray photos, [17] outlined a procedure similar to the objectives , with a model that (i) has small(er) size due to *knowledge distillation*; (ii) general understanding with high accuracy due to being trained with a larger and more general dataset, (iii) avoiding the (a) **vanishing gradient**, (b) **unnecessarily rapid complexity growth**, and (c) **individual-neuron dependency** issues of traditional *Convolutional Neural Networks* (CNNs), a specific type of NN used for image-classification tasks in the field of CV. In the methodology, the authors describe how all but the last hidden layers of the vanilla DenseNet model are frozen prior to fine tuning to a specific task. Besides reducing training needs, it avoids overfitting by maintaining better connections and stronger background prior to training and inference.

2) Logistics: In an optimization paper on one- and two-dimensional camera-interpreted machine-readable mediums, [18] demonstrated how the use of ML can consistently reduce the needs from human intervention or lack of information reading in situations where information matrices are partially hidden, skewed, or altered in some way such such that (i) accurate reading is more consistent among different tasks and (ii) reduces inference time and error rates.

3) Robotics: In a comparison between a rudimentary evolutionary and a state-of-the-art (SOTA) *PPO* reinforcement learning (RL) ML model show that although for a much greater size, the better improved model has shown to have a much faster training performance despite significant additional resource requirements for a much better result in a shorter time [19]. Particularly for pre-trained models, they are extremely robust at what they do, and given that they have a strong meta-understanding of the general task, they are able to teach a smaller model in a much better way than any other model. The subfield of robotics, additionally, also includes autonomous driving vehicles, which by itself uses both RL and supervised-learning (SL) algorithms. Although significantly different, their ideas are similar, and both are going to be utilized on this research, notably for auto-hyperparameter tuning.

C. *Applications to this paper*

The three analyzed fields all greatly benefit from reductions in inference energy, time, and accuracy, without the overwhelming need of additional data for training, usually when additional training data (or occasions for RL applications) is typically too costly. The solutions for these problems were highlighted in their respective papers. Each has their own strength, but the idea in this paper is to analyze how different methods combine within each other—to be further explained in the methodology section.

III. Methodology

A. *What is the problem with standard methods?*

1) Principle of least effort: Traditional AI approaches, like standard *Stochastic Gradient Descent* (SGD), simply move in the direction of the gradient, aiming to find a point of minimum energy. The issue is that with many parameters, as mentioned earlier, nearby local minima often aren't optimal in the parameter space, and hence, a model has to (i) repeat this process multiple times to make sure its specific points are even able to reach local minima, and (ii) initialize at multiple points to make sure of the full capturing of the loss function. Standard SGD would follow the same as (3).

2) Lack of meta-robustness: Modern behavioral human minds think the way they do because of tens to hundreds of thousands of years of evolution and adaptation to our environment [20]. Although DL algorithms can simulate thousands to millions of years of evolution given enough computing power, it is clear that without proper guidance to help a model achieve human goals, no computer can achieve human-like performance.

3) Static model overparametrization: A current issue in DL is that of *scaling laws*, which empirically show how different model sizes, architectures, and training dataset sizes respond to distinct data inputs [21]. [22] provide a major change to standard *static* models—that which have for the longest time been considered standard: *adaptive* models. As much as model depth or width weakly correlate to model performance, "performance depends strongly on scale" [21].

B. *Improved inference performance given improved training paradigms*

1) Reasoning: Evaluations show that, over the course of a model's lifecycle, approximately 10–20% [23] to 35% [24] of its energy usage purely consists of training needs—with rest being solely used for inference up to the point where it is fully replaced by a newer model. It becomes clear, hence, that improving model inference performance is helpful as it leads to (i) improved results; (ii) better end-user satisfaction, and (iii) improved model turnover time (similar how employing better workers in a company leads to fewer future worker substitutions in a given time frame).

2) Approach: Rather than reducing training requirements, the presented rationale is to increase it. Instead of inefficiency, higher energy usages come as a result of higher robustness used for model preparation. The hypothesis is that by having a more robust way of reaching optimal parameters, a model will either (i) achieve a better final stable accuracy with similar number of parameters, or (ii) achieve similar metrics as traditional models with fewer parameters. In one hand, better accuracy leads to direct improvements without significant changes in the architecture, and in the other, smaller models lead to lower power draw. It is thought that, should these methods be implemented in a large—off-device—inference scale, the first alternative will be preferred. However, for small—on-device—inference, the second will be preferred.

C. *Course of action*

Several aforementioned hindrances to an ideal CV model have been identified. This section highlights the main points that will be utilized to counteract those points.

1) Sharpness-Aware Minimization: Many examples one may encounter themselves with when searching the internet depict parameter spaces with clean, rounded, edges. Although it may be a misfortune for them not to be, [25], demonstrate how a minimization technique proposed as *Sharpness-Aware Minimization* (SAM), later expanded through Adaptive-SAM (ASAM) [26], gets rid of jagged corners (refer to Fig. 4 for a visualization). In this case, the principle of least effort (previously deemed as an innacuracy for regular models) has a higher chance of truly approaching a global minimum.

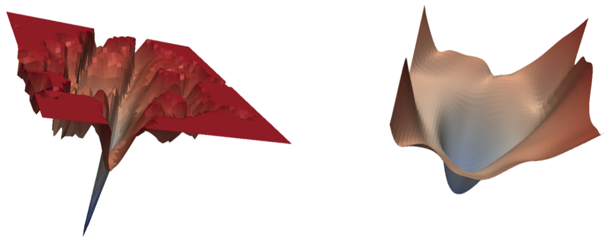

Fig. 4. Comparison of parameter spaces: The original jagged parameter space (left) exhibits sharp and irregular contours, while the application of Sharpness-Aware Minimization (right) produces a smoother landscape, facilitating convergence toward a global minimum [25].

2) Robustness techniques: In order to combat the typical lack of meta-robustness, two regularization approaches will be at play: the usage of *Stochastic gradient Langevin dynamics* (SGLD) for the optimization technique, as well as *data augmentation* (DA) of the dataset in question. SGLD helps, following the general idea previously provided at (3), now with an additional layer of added *Gaussian* noise. DA adds random crops, hue shifts in the color, flips, and so forth. The two techniques add to the dataset noise, seeking robustness future model inference.

3) Decreasing model size: An adaptive-size model and random neuron dropout (ND) are the two techniques aiding the model from becoming too large or unnecessarily reliant on specific sections or neurons of the NN. While adaptive models have been previously discussed, ND has not. The process works by, during training, arbitrarily deactivating neurons. In turn, the model understand to distill knowledge among all of its neurons.

D. Data preparation

All analyses will be performed equally on all models, where the following are held constant among all models:

1) Learning rate: $\eta = 0.001$
2) Activation function: ReLu[1]
3) Loss function: Cross-entropy
4) GPU: NVIDIA A100 80GB[2]
5) Dataset: ImageNet and CIFAR-{10, 100}[3]

A couple parameters are relative to each configuration—following a general guide—such as the *model stop*, which will depend on the average initial learning rate on that specific configuration.

All utilized code samples, datasets, additional remarks, further updates, and news are available in section V.

E. Data analysis

All data gotten from the models (final accuracy, scalar loss, epochs, power usage, etc.) can and will be used in order to determine to what extent the employment of different techniques were helpful, not helpful, or if they had no effect, and also to which extent they altered the final inference predictions of a model.

IV. Conclusion

A. Further explorations

There are multiple considerations that have not been taking during this round of experimentation. The primary reason arises by the fact that CIFAR-{10, 100} is not a particularly diverse or complex dataset. The following explorations, although may be implemented in the same dataset, may not lead to significant improvements in performance given the truly greater increase in computational needs. In larger datasets, however, the implementation of the following could lead to improvements in most, if not all, of the aforementioned measured performance metrics.

1) Combatting superlinear algorithmic performance growth through model modularization: When inputting images to multimodal models of vertical text, rather than trying to flip the image (given much of the labeled data tend to in accordance with some basic assumptions that, for example, a user will only input data that makes sense to be read), the model will either (i) hallucinate into text that does not exist based on previous context or cues, or (ii) admit the text is not readable. Assumptions can also logically take place, such as in self-driving cars, which (hopefully) will not be looking at greatly rotated images. However, particularly in business-to-customer (B2C) interactions where coherence in the utilization of an interface is not always enforceable, having separate models with the sole job of making the raw data more easy to be interpreted by a pure classification model can greatly leverage both the performance of the classification and the reduction in computational needs due to the parsing of diverse data into ones that are consistent. The pipeline, or modular, architecture was not explored in this paper given both the complexity of the analysis dataset and the assumption is that the performance evaluations came after some level automated or human revision over the training data. Still, data augmentation was put in place in the assumption that revisions still left a margin of error comparable to that of the added variability.

Additionally, similar to the vertical information flow of the pipeline architecture, *Mixture of Experts* (MoE) assimilates to an automated horizontal flow counterpart. The process is done during the training phase in a process analogous to that of *unsupervised learning* (UL), where distinct inputs require distinct inference procedures, all of which that are clustered without the need of any labeled data. The hypothesis put forth is that MAML's quick adaptation, coupled with SGLD's high observability, have the potential to greatly improve MoE's expert selections. Better selection means each expert has to be less general to areas the automatic clustering process selected, leading to smaller, more efficient, models. Additionally, the paradigm has an astounding capability to reduce model utilization given that fractions of the whole model are to be called, instead of the total inference architecture.

2) Combatting linearly proportional model growth: Floating point constraints

Knowledge distillation: Models can get really large very quickly, yet this is not always beneficial, let alone efficient. Teacher-student in ML is a paradigm which makes a large model "teach" a smaller "student" model the intricacies of a task. This process can lead to substantially smaller models despite possessing similar performance levels.

3) Additional variability: Hyperparameters were chosen mainly on literary review and state-of-the-art models, not on empirical testing.

V. Resources

A. Code samples

All code utilized in this paper are available in this GitHub repo.[4]

[1] *Rectified Linear Unit*, a popular NN activation function. $\mathrm{ReLU}(\mathbf{z}) := \max(0, \mathbf{z})$.

[2] Card will be run on `vast.ai`, using the same provider for all runs for consistency.

[3] Common image classification datasets consisting of thousands of images for both training and testing.

[4] Not yet available as of yet.

B. *Further reading*

1) Interactive MAML [27]
2) Neural networks [28]

References

[1] S. J. Russell and P. Norvig, *Artificial Intelligence: A Modern Approach*, fourth edition ed., ser. Pearson Series in Artificial Intelligence. Hoboken, NJ: Pearson, 2021.

[2] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser, and I. Polosukhin, “Attention Is All You Need,” 2017.

[3] OpenAI, “Introducing ChatGPT,” https://openai.com/index/chatgpt/, Nov. 2022.

[4] L. K. Gronmeyer, B. W. Ruffin, M. A. Lybanon, S. E. Pierce, Jr., and P. L. Neely, “An Overview of Optical Character Recognition (OCR) Technology and Techniques,” Naval Ocean Research and Development Activity (NORDA), NSTL Station, Mississippi, Technical Report ADA131341, Jun. 1978.

[5] E. Goldberg, “Statistical Machine,” Germany Patent US1 838 389A, Dec., 1931.

[6] R. Fisher, S. Perkins, A. Walker, and E. Wolfart, “Hough Transform,” https://homepages.inf.ed.ac.uk/rbf/HIPR2/hough.htm, 2003.

[7] P. V. C. Hough, “Method and means for recognizing complex patterns,” US Patent US3 069 654A, Dec., 1962.

[8] Y. LeCun, B. Boser, J. S. Denker, D. Henderson, R. E. Howard, W. Hubbard, and L. D. Jackel, “Backpropagation Applied to Handwritten Zip Code Recognition,” *Neural Computation*, vol. 1, no. 4, pp. 541–551, Dec. 1989.

[9] A. Kay, “Tesseract: An Open-Source Optical Character Recognition Engine,” https://www.linuxjournal.com/article/9676, Jul. 2007.

[10] D. Kahneman, *Thinking, Fast and Slow*, ser. Penguin Psychology. London: Penguin Books, 2012.

[11] D. M. Hawkins, “The Problem of Overfitting,” *American Chemical Society*, vol. 44, no. 1, pp. 1–12, Dec. 2003.

[12] Y. Nhi, “Understanding the Cost Function in Linear Regression for Machine Learning Beginners,” May 2023.

[13] C. E. Shannon, “A Mathematical Theory of Communication,” *The Bell System Technical Journal*, vol. 27, pp. 379–423, 623–656, Oct. 1948.

[14] ThirdOrderLogic, “Pyplot overfitting,” https://commons.wikimedia.org/wiki/File:Pyplot_overfitting.png, Jul. 2024.

[15] MartinThoma, “3d-gradient-cos.svg,” Aug. 2018.

[16] Papers With Code, “Image Classification on CIFAR-100,” https://paperswithcode.com/sota/image-classification-on-cifar-100, 2024.

[17] M. Bundea and G. M. Danciu, “Pneumonia Image Classification Using DenseNet Architecture,” *Information*, vol. 15, no. 10, p. 611, Oct. 2024.

[18] J. Chen, N. Dai, X. Hu, and Y. Yuan, “A Lightweight Barcode Detection Algorithm Based on Deep Learning,” *Applied Sciences*, vol. 14, no. 22, p. 10417, Nov. 2024.

[19] T. Jean, “Reinforcement Learning - My Algorithm vs State of the Art,” Nov. 2024.

[20] C. W. Marean, M. Bar-Matthews, J. Bernatchez, E. Fisher, P. Goldberg, A. I. R. Herries, Z. Jacobs, A. Jerardino, P. Karkanas, T. Minichillo, P. J. Nilssen, E. Thompson, I. Watts, and H. M. Williams, “Early human use of marine resources and pigment in South Africa during the Middle Pleistocene,” *Nature*, vol. 449, no. 7164, pp. 905–908, Oct. 2007.

[21] J. Kaplan, S. McCandlish, T. Henighan, T. B. Brown, B. Chess, R. Child, S. Gray, A. Radford, J. Wu, and D. Amodei, “Scaling Laws for Neural Language Models,” Jan. 2020.

[22] S. Anagnostidis, G. Bachmann, I. Schlag, and T. Hofmann, “Navigating Scaling Laws: Compute Optimality in Adaptive Model Training,” May 2024.

[23] C. Douwes and R. Serizel, “From Computation to Consumption: Exploring the Compute-Energy Link for Training and Testing Neural Networks for SED Systems,” Sep. 2024.

[24] C.-J. Wu, R. Raghavendra, U. Gupta, B. Acun, N. Ardalani, K. Maeng, G. Chang, F. A. Behram, J. Huang, C. Bai, M. Gschwind, A. Gupta, M. Ott, A. Melnikov, S. Candido, D. Brooks, G. Chauhan, B. Lee, H.-H. S. Lee, B. Akyildiz, M. Balandat, J. Spisak, R. Jain, M. Rabbat, and K. Hazelwood, “Sustainable AI: Environmental Implications, Challenges and Opportunities,” Jan. 2022.

[25] P. Foret, A. Kleiner, H. Mobahi, and B. Neyshabur, “Sharpness-Aware Minimization for Efficiently Improving Generalization,” Apr. 2021.

[26] J. Kwon, J. Kim, H. Park, and I. K. Choi, “ASAM: Adaptive Sharpness-Aware Minimization for Scale-Invariant Learning of Deep Neural Networks,” Jun. 2021.

[27] Luis Müller, Max Ploner, and Thomas Goerttler, “An Interactive Introduction to Model-Agnostic Meta-Learning,” https://interactive-maml.github.io/, Oct. 2021.

[28] 3Blue1Brown, “Neural Networks,” https://www.3blue1brown.com/topics/neural-networks, Aug. 2024.