From Matrix-Based to Dynamical Deep Learning: Synchronization and Energy

Yasusi Kanada
Dasyn.com
yasusi@kanadas.com

Abstract — Deep learning has historically developed around discrete, matrix-based representations that align naturally with GPU-oriented, globally synchronized computation. While this paradigm has enabled remarkable computational performance, the power required for large-scale training and inference has become a significant practical and societal constraint as model size and deployment scale continue to grow. In recent years, continuous-time and dynamical formulations such as Neural Ordinary Differential Equations, diffusion models, and state-space models have gained increasing attention. This paper argues that this representational shift has implications beyond modeling style. First, it challenges the implicit reliance on global synchronization inherent in matrix-centric computation, opening possibilities for more asynchronous or locally coordinated execution. Second, it suggests new opportunities for improving energy efficiency by reducing synchronization and data movement overhead. Third, this transition admits a limited but informative structural analogy with the historical development of quantum mechanics, in which matrix mechanics and wave mechanics emerged as distinct yet complementary representational frameworks. Without claiming mathematical equivalence, this analogy is proposed as a conceptual perspective for understanding an ongoing transition in the foundations of deep learning.

Keywords — *Deep learning; Matrix-based computation; Continuous-time models; Neural Ordinary Differential Equations (Neural ODE); Diffusion models; State space models; Energy efficiency; GPU architectures; Dynamical systems; Representation learning; Analog computation; Computational sustainability.*

I. INTRODUCTION

Deep learning has developed predominantly on the assumption of matrix-based computation. From early feedforward networks to modern architectures such as convolutional neural networks and Transformers [22], the core computational abstraction has remained largely unchanged: models are constructed as discrete layers, each implemented primarily through matrix or tensor multiplications. This abstraction has provided a clear mathematical framework and has supported rapid progress over the past decade [13][8]. It is worth noting that the core learning principle underlying neural networks—gradient-based optimization [18][19] and its stochastic variants rooted in stochastic approximation theory [18][3]—was originally developed in the context of continuous optimization and stochastic processes, prior to the widespread adoption of layer-wise, matrix-centric implementations. Later developments, such as information-geometric formulations of learning [1], further deepened this perspective within neural network models.

The success of this paradigm is inseparable from the evolution of GPU architectures. General matrix–matrix multiplication (GEMM) has been optimized to an extraordinary degree, becoming the fundamental operation around which modern accelerators, software frameworks, and programming models are organized. As a result, deep learning systems have achieved unprecedented levels of computational throughput by aligning model structure with the strengths of GPU hardware.

At the same time, this tight coupling has also revealed growing limitations. High performance is achieved by aligning model structure with globally synchronized, batch-oriented execution and large-scale data movement, a computational organization that, while effective, leads to substantial energy consumption at scale [12][11]. As models continue to scale and are increasingly deployed in real-world systems, the power required for training and inference has emerged as a critical constraint. The issue of energy efficiency is no longer a purely engineering concern, but one that intersects with economic cost, infrastructure capacity, and environmental impact.

In parallel with these challenges, recent years have witnessed the rapid emergence of alternative formulations of deep learning based on continuous-time and dynamical representations, including Neural Ordinary Differential Equations (Neural ODE) [5], stochastic differential equation–based generative models [4][21], and state-space models [9][6]. These approaches differ fundamentally from the traditional layer-by-layer matrix multiplication paradigm and suggest new ways of organizing computation.

The purpose of this paper is to reconsider the current state of deep learning by making explicit the representational assumptions that underlie models, computational organization, and hardware execution. Focusing on the transition from matrix-centric formulations to continuous-time and dynamical representations, we examine how representation influences synchronization, scalability, and energy use in modern systems.

This paper argues that the ongoing shift in deep learning from matrix-centric representations toward continuous-time and dynamical formulations has several important implications. First, this transition challenges the implicit reliance on globally synchronized computation that has accompanied matrix-based models and GPU-oriented implementations, opening the possibility of more asynchronous or locally coordinated computational structures. Second, this representational shift suggests new opportunities for improving energy efficiency, as reduced global synchronization and localized state evolution may alleviate data movement and power consumption bottlenecks inherent in large-scale matrix computation. Third, this development admits a limited but informative structural

analogy with historical transitions in quantum mechanics, where changes in representational frameworks reshaped theoretical understanding without immediate one-to-one mathematical correspondence.

II. Discrete, Matrix-Based Deep Learning: Strengths and Limits

Modern deep learning architectures are predominantly built upon discrete, matrix-based representations [8]. Widely used models such as residual networks (ResNet) [10] and Transformer-based architectures [22] can be viewed, at an abstract level, as compositions of layers, each of which is implemented primarily through matrix or tensor multiplications followed by element-wise nonlinearities. This formulation has proven remarkably successful, enabling both expressive modeling and efficient implementation.

The dominance of this paradigm is closely tied to the evolution of GPU architectures. Matrix multiplication has been optimized to an extraordinary degree on modern accelerators, achieving near-peak hardware utilization under favorable conditions. Large-scale general GEMM serves as the computational backbone of contemporary deep learning frameworks, allowing massive parallelism and high arithmetic throughput.

However, this efficiency is achieved under specific assumptions. First, performance depends strongly on batch size and sequence length. High utilization of GPU resources requires sufficiently large matrices so that parallel execution units can be fully occupied. In scenarios involving small batch sizes, short sequences, or real-time and streaming inference, computational efficiency can degrade significantly. This dependency introduces practical constraints for applications that demand low latency or incremental processing.

Second, although matrix multiplication is known to exhibit high data reuse in theory, the extent to which weights and intermediate representations can be effectively reused is structurally limited in deep learning workloads. As model size grows, parameters and activations often exceed on-chip memory capacity, forcing frequent transfers between memory hierarchies. Moreover, successive layers typically involve different weight matrices and transformed representations, restricting reuse across layers. As a result, data movement increasingly dominates computation, giving rise to well-known memory bottlenecks that cannot be resolved by faster arithmetic units alone.

Third, scaling model size leads to rapid growth in computational cost, memory consumption, and communication overhead. As models become deeper and wider, the number of parameters and intermediate activations increases accordingly, requiring more arithmetic operations and larger memory footprints. During training, forward propagation, backward propagation, and parameter updates must be repeatedly executed, and intermediate states must be stored or recomputed, further amplifying resource demands. In distributed settings, these effects are compounded by the need to synchronize parameters and gradients across devices, leading to increased communication overhead. Together, these factors indicate that continued scaling imposes growing structural burdens on computation and memory systems.

Taken together, these factors suggest that the matrix-based paradigm, while extraordinarily effective, is approaching structural limits. This observation resonates with broader reflections in the machine learning community, including the argument that progress driven primarily by scale and computation eventually encounters diminishing returns. In this sense, the current "matrix-based era" of deep learning may be nearing a point where alternative representations and computational paradigms warrant serious consideration.

III. The Emergence of Continuous-Time Deep Learning

In contrast to the discrete, layer-wise formulation that has dominated deep learning, recent years have seen the rapid emergence of models based on continuous-time and dynamical representations. Notably, this shift resonates with earlier views of learning as a dynamical and stochastic process. Gradient-based optimization and its stochastic variants were originally formulated in the context of continuous optimization and stochastic approximation [18], and were later given a geometric interpretation within neural network learning through information-geometric formulations [1]. More recent work has further revisited stochastic gradient descent as a stochastic dynamical system with nontrivial long-term behavior in deep networks [4]. These approaches depart from the assumption that computation must be organized as a sequence of distinct matrix-based layers, and instead describe learning and inference as the evolution of states over continuous time.

A. Neural Ordinary Differential Equations

The introduction of Neural Ordinary Differential Equations (Neural ODEs) [5] marked a conceptual shift in how deep neural networks can be understood. Neural ODEs can be interpreted as the continuous-time limit of residual networks, in which the depth of the network tends to infinity while the change induced by each layer becomes infinitesimal. In this formulation, the notion of discrete layers effectively disappears, and network behavior is governed by an ordinary differential equation parameterized by a neural network.

A key consequence of this perspective is that parameters are reused continuously over time, rather than being tied to individual layers [5]. This contrasts sharply with conventional deep networks, where each layer introduces a new set of parameters and intermediate representations. By framing computation as a dynamical system, Neural ODEs suggest alternative ways of organizing model capacity, memory usage, and temporal evolution.

B. Neural Stochastic Differential Equations and Diffusion Models

Building on this continuous-time viewpoint, stochastic differential equation (SDE)–based models [4][21] have become central to modern generative modeling. Diffusion models, which have achieved state-of-the-art performance in image, audio, and text generation [21], are naturally described as continuous-time stochastic processes that progressively transform noise into structured data.

In this setting, the essence of generation is not a sequence of deterministic transformations, but rather the controlled evolution of a probability distribution over time. Although these models are often implemented using discrete time steps for practical reasons, their theoretical foundation lies in continuous stochastic dynamics. This represents a further departure from traditional layer-based interpretations and reinforces the view that generative modeling is increasingly grounded in continuous-time processes.

C. State Space Models and Long-Sequence Modeling

Another important development is the resurgence of state space models (SSMs) for sequence modeling, exemplified by architectures such as S4 [9] and Mamba [6]. These models move away from both attention mechanisms and conventional convolutional formulations, instead leveraging the stability and expressive power of continuous-time linear dynamical systems.

By modeling sequences through latent states that evolve according to structured dynamics, SSM-based approaches offer favorable scaling properties for long sequences and emphasize local state updates rather than global interactions. While matrix operations remain part of their implementation, the underlying representational framework is fundamentally dynamical, rooted in continuous-time system theory rather than being organized around layer-wise compositions of large GEMMs.

D. Continuous Dynamics in Broader Learning Frameworks

Beyond these specific model classes, continuous-time dynamics have also appeared in a wide range of learning paradigms. Physics-informed neural networks [16] explicitly encode differential equations into the learning process, treating neural networks as function approximators for continuous physical systems. Similarly, certain approaches in implicit neural representations [20], meta-learning [7], and optimization-based learning frameworks [3] interpret adaptation and inference as trajectories in continuous parameter or state spaces.

E. Toward Non–Matrix-Centric Representations

Taken together, these developments indicate a broader trend: the representational foundations of deep learning are no longer exclusively tied to discrete, matrix-based layer constructions. Instead, an increasing number of models are naturally described in terms of continuous-time evolution, dynamical systems, and state transitions. While matrix operations remain central at the level of implementation, the conceptual shift toward continuous dynamics suggests an expanding space of representations that are not fundamentally defined by matrix multiplication alone.

This growing diversity of representational frameworks sets the stage for reconsidering both the theoretical and practical assumptions underlying modern deep learning, particularly in relation to scalability, energy efficiency, and computational organization.

IV. Implications of the Representational Transition

The transition from matrix-centric representations to dynamical and continuous-time formulations in deep learning is not merely a change in modeling technique. Rather, it carries a set of broader implications concerning how computation is organized, how resources are consumed, and how theoretical frameworks are conceptualized. This section examines three such implications: the role of global synchronization in computation, its consequences for energy efficiency and scalability, and a limited structural analogy with historical conceptual transitions in quantum mechanics.

A. From Global Synchronization to Local Dynamics

Matrix-based representations implicitly favor a computational model built around global synchronization. In conventional layer-based neural networks, neurons within a given layer are typically evaluated synchronously: the outputs of all units in a layer are computed together before computation proceeds to the next layer. This layer-wise synchronization is central to how such models are both specified and executed in practice.

The mathematical formalism underlying this mode of computation is the matrix. Although matrices as abstract mathematical objects do not explicitly encode synchronization, they are commonly interpreted as representing simultaneous operations over collections of elements. Writing a matrix multiplication conceptually groups many operations into a single computational step, an interpretation that naturally aligns with synchronous execution.

This interpretation, in turn, matches the design of synchronous digital circuits. Modern hardware executes matrix operations by coordinating large numbers of arithmetic units under a shared clock, ensuring lockstep processing across layers and batches. The effectiveness of matrix-centric deep learning is therefore closely tied to hardware architectures optimized for globally synchronized computation.

Importantly, synchronization is not an explicit concept in most mathematical formulations of learning or computation. Nevertheless, many existing theories rely implicitly on assumptions of global consistency and ordered updates, even in pure mathematics, where theorem proving and formal reasoning typically presuppose a well-defined sequence of evaluation steps. Such assumptions are largely unproblematic in human reasoning and in centrally controlled, synchronous computation. However, they become explicit constraints when computation is distributed, asynchronous, or subject to formal verification. In these settings, removing global synchronization exposes ambiguities and inconsistencies that were previously hidden, revealing limitations in existing theoretical frameworks and motivating the development of new mathematical tools capable of reasoning about learning dynamics without implicit global order [14].

From this viewpoint, global synchronization should be understood not as an inherent requirement of neural computation, but as a consequence of matrix-centric modeling and its associated computational interpretation. The assumption that computation proceeds through globally ordered, layer-wise updates arises naturally when learning is formulated in terms of discrete matrix operations executed in lockstep. However, this organizational structure reflects historical and practical design choices rather than a fundamental property of learning dynamics themselves. Recognizing this distinction is essential for understanding how alternative representational frameworks can

support different modes of computation, including asynchronous, decentralized, or locally coordinated dynamics.

A similar principle can be observed in biological computation. Neural processing in the brain unfolds through continuous-time dynamics and predominantly local interactions, without relying on explicit matrix multiplications or strict global synchronization. While biological systems differ fundamentally from artificial neural networks in their physical realization and learning mechanisms, their existence demonstrates that highly efficient computation can be achieved without globally synchronized, layer-wise execution. This observation reinforces the view that global synchronization is a contingent feature of particular representational and hardware choices, rather than a universal requirement for effective computation.

B. Synchronization and Energy Consumption

The limitations discussed in Section II are often described in terms of hardware efficiency, memory bandwidth, or system-scale optimization. In this section, we revisit these issues from a different perspective. Rather than treating energy consumption and scalability as purely engineering challenges, we interpret them as consequences of representational and organizational assumptions embedded in matrix-centric computation, particularly its reliance on global synchronization.

The implications of reduced reliance on global synchronization extend beyond conceptual clarity to practical concerns of energy consumption and scalability. In large-scale matrix-based systems, a significant portion of energy is expended not on arithmetic operations themselves, but on data movement and coordination. Global synchronization often amplifies these costs by requiring frequent communication, memory transfers, and idle waiting across processing units.

As model sizes grow, these overheads become increasingly dominant. The need to maintain synchronized execution across large batches and deep layer stacks exacerbates memory bandwidth demands and contributes to rising power consumption in training and inference. From this perspective, energy inefficiency is not merely a byproduct of large models, but is closely tied to the organizational assumptions embedded in matrix-centric computation.

Dynamical representations suggest alternative organizational principles. Local state updates, parameter reuse over time, and continuous evolution can reduce the frequency and scope of global coordination. In such settings, computation may be structured around localized interactions and incremental updates, potentially alleviating some of the data movement and synchronization overheads that dominate energy consumption in current systems [2][17].

It is important to emphasize that energy efficiency is not the primary objective of adopting dynamical representations. Rather, any potential reductions in power consumption should be viewed as secondary consequences of a broader representational shift. Whether and to what extent these advantages can be realized in practice depends on hardware design, numerical methods, and system-level considerations. Nonetheless, the transition away from strict global synchronization opens a plausible pathway toward more energy-efficient and scalable learning systems.

C. A Structural Analogy with Quantum Mechanics

The historical development of quantum mechanics provides a useful reference point for situating the representational transition discussed in this paper, in which multiple mathematical frameworks coexist to describe the same underlying phenomena. In the mid-1920s, quantum theory emerged through two seemingly distinct formulations: matrix mechanics, developed by Heisenberg, and wave mechanics, introduced by Schrödinger. Matrix mechanics described physical observables using discrete algebraic structures, emphasizing operator relations and non-commutative matrices. Wave mechanics, by contrast, formulated quantum phenomena in terms of continuous wave functions evolving over space and time.

Despite their markedly different mathematical forms and conceptual intuitions, matrix mechanics and wave mechanics were soon understood as complementary descriptions of the same physical phenomena. Each emphasized different structural aspects of quantum theory, offering distinct perspectives on quantum behavior. This relationship was later clarified—most notably through the work of von Neumann [15]—by showing that the two formulations are mathematically equivalent representations of a single underlying theory. Their coexistence therefore reflected not merely an alternative choice of notation, nor a simple replacement of one formalism by another, but a deeper conceptual reorganization of how physical systems could be described at a time when classical intuitions about dynamics, measurement, and representation were being fundamentally revised. More broadly, this episode illustrates how shifts in representational frameworks can reshape theoretical understanding without necessitating the immediate abandonment of established formalisms.

A similar, though not mathematically equivalent, situation can be observed in contemporary deep learning. Matrix-based models and dynamical formulations represent different ways of organizing and interpreting computation. Neither framework fully subsumes the other, and both continue to coexist in practice. The analogy lies in the recognition that shifts in representation can reshape conceptual understanding without requiring immediate unification or equivalence.

It is crucial to stress that this analogy is structural and historical rather than formal. This paper does not claim any mathematical correspondence between the equations of quantum mechanics and learning dynamics. Instead, the analogy serves as a conceptual aid for interpreting the current diversification of representational frameworks in deep learning. By viewing the emergence of dynamical models as part of a broader transition in how computation is represented, it becomes possible to situate recent developments within a larger pattern of theoretical evolution.

V. IMPLICATIONS FOR ENERGY-EFFICIENT HARDWARE

Matrix-based deep learning architectures implicitly rely on globally synchronized execution. In layer-based models, all activations within a layer must be computed and aligned before the next transformation can proceed. These layer-wise synchronization barriers induce concentrated data movement,

idle waiting, and bursts of memory access, leading to high energy density and significant power consumption in modern GPU-based accelerators, as observed in large-scale accelerator studies such as TPU and GPU deployments [12].

By contrast, dynamical representations do not require strict layer-wise global synchronization as a principle. State evolution can, at least conceptually, proceed through asynchronous or locally coordinated updates, allowing different components of the system to evolve without centralized coordination. Although practical implementations often reintroduce discretization and partial synchronization, the representational framework itself does not mandate global barriers.

From an energy perspective, this distinction is significant. A substantial fraction of energy consumption in large-scale systems arises not from arithmetic operations alone, but from synchronization overhead, data movement, and waiting induced by global coordination, as extensively analyzed in hardware-level studies [11]. Relaxing global synchronization therefore opens the possibility of reducing these overheads, even when the total number of parameters or arithmetic operations remains comparable.

Importantly, any potential reduction in energy consumption should be understood as a consequence of altered synchronization structure rather than as an intrinsic property of continuous-time models. The transition from matrix-based to dynamical representations does not guarantee energy efficiency by itself, but it expands the design space in which alternative trade-offs between synchronization, data movement, and power consumption can be explored.

From this viewpoint, the evolution of deep learning representations and the pursuit of energy-efficient hardware appear to be aligned through a shared shift away from globally synchronized computation toward more flexible, locally coordinated execution. Recognizing this alignment may be essential for sustaining progress in deep learning under increasingly stringent energy and resource constraints.

The implications discussed in this section do not arise from a reduction in model size or parameter count. Instead, they stem from differences in computational organization, particularly the degree to which global synchronization is required during execution. From this perspective, energy efficiency is not treated as a purely engineering concern, but as a consequence of representational assumptions embedded in matrix-centric computation.

Removing global synchronization also exposes assumptions that are usually left implicit in mathematical formulations of learning dynamics. While such assumptions are largely unproblematic under synchronous execution, studies of asynchronous and decentralized optimization have shown that they become explicit constraints when synchronization is relaxed, requiring new theoretical tools to reason about consistency, convergence, and stability [14].

VI. Concluding Remarks: A Position, Not A Prescription

The preceding sections have argued that contemporary deep learning is undergoing a shift in its dominant representational assumptions. Discrete, matrix-based formulations—long aligned with GPU-oriented, globally synchronized computation—are increasingly complemented by continuous-time and dynamical perspectives. This paper has approached this development not as a proposal of new models, but as a conceptual reexamination of how representation, computation, and constraint interact.

A central theme of this work is that many practical challenges commonly framed as engineering issues—such as scalability, energy consumption, and system-level efficiency—are deeply connected to representational and organizational assumptions embedded in matrix-centric computation. In particular, the reliance on global synchronization emerges not as a fundamental requirement of learning itself, but as a consequence of how computation has been structured around discrete matrix operations.

From this perspective, continuous-time and dynamical formulations are not presented as replacements for matrix-based models, nor as guaranteed solutions to efficiency concerns. Rather, they expand the space of possible computational organizations, including those that relax strict global synchronization and emphasize local or incremental state evolution. Whether such possibilities translate into practical advantages depends on many factors, including algorithm design, hardware support, and system integration.

More broadly, this work suggests that deep learning may not admit a single, essential mathematical representation. Matrix-based and dynamical formulations emphasize different aspects of learning—algebraic composition on the one hand, and trajectories, stability, and evolution on the other. Their coexistence should therefore be understood not as a temporary inconsistency, but as an expression of the field's increasing conceptual richness.

In this sense, the contribution of this paper is not prescriptive. It does not advocate a specific architecture, training method, or hardware platform. Instead, it articulates a position: that ongoing developments in deep learning are best understood by making explicit the representational assumptions that underlie computation, synchronization, and energy use. By clarifying these assumptions, we hope to provide a framework for interpreting current trends and for motivating future work that aligns expressive power with sustainable computation.

Acknowledgments

The author acknowledges the use of ChatGPT (version 5.2, OpenAI) in the preparation of this manuscript. Responsibility for the content rests with the author.

References

[1] H. Amari, "Natural gradient works efficiently in learning," *Neural Computation,* Vol. 10, No. 2, pp. 251–276, 1998.

[2] G. Bertsekas and J. Tsitsiklis, "Parallel and Distributed Computation: Numerical Methods," Englewood Cliffs, NJ, USA: Prentice-Hall, 1989.

[3] L. Bottou, F. E. Curtis, and J. Nocedal, "Optimization methods for large-scale machine learning," *SIAM Rev.*, Vol. 60, No. 2, pp. 223–311, 2018.

[4] P. Chaudhari and S. Soatto, "Stochastic gradient descent performs variational inference, converges to limit cycles for deep networks," *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, 2018, pp. 6456–6465.

[5] R. T. Q. Chen, Y. Rubanova, J. Bettencourt, and D. Duvenaud, "Neural ordinary differential equations," *Proc. Adv. Neural Inf. Process. Syst. (NeurIPS)*, 2018, pp. 6571–6583.

[6] T. Dao, D. Fu, S. Ermon, A. Rudra, and C. Ré, "Mamba: Linear-time sequence modeling with selective state spaces," *Proc. Int. Conf. Learn. Represent. (ICLR)*, 2024.

[7] C. Finn, P. Abbeel, and S. Levine, "Model-agnostic meta-learning for fast adaptation of deep networks," *Proc. Int. Conf. Mach. Learn. (ICML)*, 2017.

[8] I. Goodfellow, Y. Bengio, and A. Courville, "Deep Learning," Cambridge, MA, USA: MIT Press, 2016.

[9] A. Gu, K. Goel, and C. Ré, "Efficiently modeling long sequences with structured state spaces," *Proc. Int. Conf. Learn. Represent. (ICLR)*, 2022.

[10] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR),* Las Vegas, NV, USA, 2016, pp. 770–778.

[11] M. Horowitz, "Computing's energy problem (and what we can do about it)," *IEEE ISSCC*, 2014.

[12] N. P. Jouppi *et al.*, "In-Datacenter Performance Analysis of a Tensor Processing Unit," *Proc. 44th Int. Symp. Computer Architecture (ISCA)*, 2017.

[13] Y. LeCun, Y. Bengio, and G. Hinton, "Deep learning," *Nature*, Vol. 521, No. 7553, pp. 436–444, May 2015.

[14] X. Lian, Y. Huang, Y. Li, and J. Liu, "Asynchronous parallel stochastic gradient for nonconvex optimization," *Proc. Adv. Neural Inf. Process. Syst. (NeurIPS)*, 2015, pp. 2737–2745.

[15] J. von Neumann, "Mathematical Foundations of Quantum Mechanics," Princeton, NJ, USA: Princeton Univ. Press, 1955.

[16] M. Raissi, P. Perdikaris, and G. E. Karniadakis, "Physics-informed neural networks: A deep learning framework for solving forward and inverse problems involving nonlinear partial differential equations," *J. Comput. Phys.*, Vol. 378, pp. 686–707, 2019.

[17] B. Recht, C. Ré, S. Wright, and F. Niu, "HOGWILD!: A lock-free approach to parallelizing stochastic gradient descent," *Proc. Adv. Neural Inf. Process. Syst. (NeurIPS)*, 2011, pp. 693–701.

[18] H. Robbins and S. Monro, "A stochastic approximation method," *Annals of Mathematical Statistics*, Vol. 22, No. 3, pp. 400–407, 1951.

[19] S. Ruder, "An overview of gradient descent optimization algorithms," arXiv:1609.04747, 2016.

[20] V. Sitzmann, J. N. P. Martel, A. W. Bergman, D. B. Lindell, and G. Wetzstein, "Implicit neural representations with periodic activation functions," *Proc. Adv. Neural Inf. Process. Syst. (NeurIPS)*, 2020.

[21] Y. Song, J. Sohl-Dickstein, D. P. Kingma, et al., "Score-based generative modeling through stochastic differential equations," *Proc. Int. Conf. Learn. Represent. (ICLR),* 2021.

[22] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser, and I. Polosukhin, "Attention is all you need," *Proc. Adv. Neural Inf. Process. Syst. (NeurIPS)*, 2017, pp. 5998–6008.