https://doi.org/10.1038/s44387-025-00025-7

A dynamic fractional generalized deterministic annealing for rapid convergence in deep learning optimization

Matthew Korban[1], Peter Youngs[2] & Scott T. Acton[1] ✉

Optimization is central to classical and modern machine learning. This paper introduces Dynamic Fractional Generalized Deterministic Annealing (DF-GDA), a physics-inspired algorithm that boosts stability and speeds convergence across a wide range of models, especially deep networks. Unlike traditional methods such as Stochastic Gradient Descent, which may converge slowly or become trapped in local minima, DF-GDA employs an adaptive, temperature-controlled schedule that balances global exploration with precise refinement. Its dynamic fractional-parameter update selectively optimizes model components, improving computational efficiency. The method excels on high-dimensional tasks, including image classification, and also strengthens simpler classical models by reducing local-minimum risk and increasing robustness to noisy data. Extensive experiments on sixteen large, interdisciplinary datasets, including image classification, natural language processing, healthcare, and biology, show that DF-GDA consistently outperforms both state-of-the-art and traditional optimizers in convergence speed and accuracy, offering a powerful alternative for critical large-scale, complex problems across diverse scientific and industrial settings today.

Optimization is fundamental in many scientific and engineering fields and is crucial in finding the best solutions to various problems[1]. It aims to adjust the parameters of a system to maximize or minimize a particular function, known as the objective function[2]. This process is essential in numerous applications, including logistics, finance, healthcare, manufacturing, and others, where achieving optimal performance or efficiency is the primary goal[3].

In the context of machine learning, optimization is critical[4]. Machine learning models, including classical methods such as k-means clustering and support vector machines, learn from data by adjusting their parameters to minimize a loss function[5,6]. This function measures how well the model's predictions match the actual data. Practical optimization algorithms are vital for training these models efficiently and accurately[7]. With proper optimization, machine learning models may converge to a suitable solution, leading to better performance and accurate predictions[8].

Deep learning, a subset of machine learning, involves training large neural networks with many layers and millions of parameters. Due to the complexity and size of these models, the role of the optimization process in deep learning is even more critical[9]. Deep learning models can achieve remarkable performance in tasks such as image recognition, natural language processing, and video understanding, but only if optimized effectively[10]. The challenges in deep learning optimization include avoiding local minima, managing high-dimensional parameter spaces, and ensuring fast and stable convergence[11,12].

Traditional optimization methods such as Stochastic Gradient Descent (SGD) and adaptive moment estimation (ADAM) are widely used in deep learning due to their simplicity and effectiveness[13,14]. However, these methods often face challenges, such as being trapped in local minima and slow convergence when dealing with high-dimensional parameter spaces and noisy data[15–17]. To address these issues, more advanced optimization techniques are necessary.

Genetic algorithms introduced feature optimization in specific domains[18], but were computationally expensive and prone to slow convergence. Nature-inspired algorithms such as Harris Hawks optimization[19] and Firefly optimization[20] further advanced the field by automating hyperparameter tuning and improving convergence rates. However, these techniques still demanded significant computational resources and carried the risk of entrapment in local minima and susceptibility to annotation noise

[1]Electrical and Computer Engineering, University of Virginia, Charlottesville, VA, USA. [2]Curriculum, Instruction and Special Education, University of Virginia, Charlottesville, VA, USA. ✉e-mail: acton@virgina.edu

(errors or inconsistencies in data labeling). Consequently, due to these limitations, they did not achieve the widespread adoption seen with methods such as SGD and ADAM.

In this work, we propose a novel optimization algorithm, namely Dynamic Fractional Generalized Deterministic Annealing (DF–GDA), to enhance convergence speed and efficacy in deep learning models. DF–GDA also shows high potential for significantly improving classical machine learning algorithms such as Support Vector Machines (SVMs) and k-means clustering, which can benefit from its robust handling of local minima and efficient exploration-exploitation balance. This approach builds on the core principles of generalized deterministic annealing (GDA)[15], which include a temperature-dependent probabilistic acceptance criterion and a mean field estimation process to estimate the values of unknown variables. The temperature-dependent acceptance criterion helps balance exploration and exploitation during the optimization, significantly reducing the risk of being trapped in local minima. The proposed DF–GDA algorithm can dramatically improve the performance of deep network models in complex research problems, including interdisciplinary applications such as image classification, video understanding, bioinformatics, healthcare analytics, and natural language processing. These are optimization landscapes characterized by multiple local minima where *traditional gradient-based methods such as Stochastic Gradient Descent (SGD) have long been considered indispensable. Our approach demonstrates the potential to significantly surpass them, representing a major advancement in optimization across a broad range of scientific and engineering domains.*

Figure 1 illustrates a comparative analysis between SGD and our DF–GDA method. Initially, both approaches start with a high-energy, disordered microstructure, a concept that represents the arrangement of parameter states in the optimization landscape, mirroring the physical process in material science. SGD demonstrates unstable updates as training progresses, often getting trapped in suboptimal configurations due to its inherent noise and sensitivity to local minima. In contrast, DF–GDA exhibits structured and localized parameter adjustments, facilitating a more controlled transition toward an optimal configuration. The final state depicted in the figure highlights that while SGD tends to remain in a disordered microstructure, DF–GDA successfully organizes the microstructure into a lower-energy state, indicating its enhanced ability to navigate complex optimization landscapes, escape local minima, and reach optimal solutions more effectively.

Although effective for image processing tasks, the original GDA method was not designed for deep learning or machine learning optimization tasks. Significant modifications were necessary to adapt GDA for the specific needs of machine learning and deep network optimization, making it suitable for large-scale deep learning applications. These adaptations involved incorporating the dynamically adjustable fraction parameter, leveraging mean-field gradient estimates, and implementing a soft quantization mechanism to ensure parameter updates remain within feasible ranges.

Remarkably, DF–GDA introduces a new dynamically adaptive fractional parameter update (DAFPU) algorithm to further enhance GDA for deep learning applications. This adaptive algorithm takes advantage of the proportion of model parameters that are to be updated during each iteration. It is sensitively adjusted on the basis of the current status of the training, including the rate of change in the loss function. This adjustment ensures a balanced trade-off between exploration and exploitation throughout training. The proposed approach makes the high-dimensional parameter space significantly more manageable, a persistent problem in deep learning optimization.

The proposed DAFPU is essential to the learning process, as it is applied during the optimization and backward pass stages. This differentiates it from dropout, a regularization technique that is used only during the forward pass. Our method, applied during the backward pass, ensures broader applicability, whereas dropout is limited because it only functions during the forward pass. Since the forward pass doesn't directly influence parameter updates, its applicability in optimization is more restricted. In particular, DAFPU also reduces the computational cost more effectively

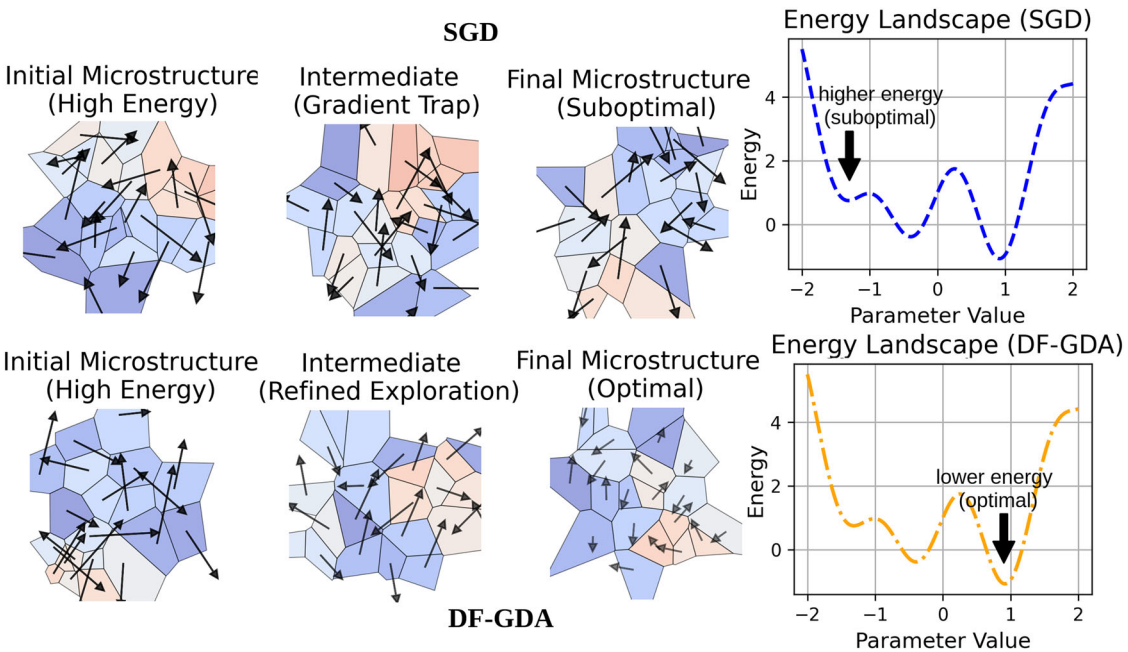

Fig. 1 | The comparison of two optimization methods: stochastic gradient descent (SGD) and dynamic fractional generalized deterministic annealing (DF–GDA). Initially, both methods start with a high-energy, disordered microstructure. In the intermediate phase, SGD displays chaotic updates and often gets stuck in local minima, leading to a continued disordered state depicted by erratic arrows in both magnitude and direction. On the other hand, DF–GDA shows controlled and localized updates (organized arrows), allowing for a structured transition toward an optimal configuration. In the final phase, SGD remains trapped in a suboptimal state with a disordered microstructure, while DF–GDA reaches an optimal state with a well-organized microstructure. The energy landscape graphs illustrate these outcomes. SGD's energy graph (blue curve) remains at higher energy levels, indicating local minima entrapment. In contrast, DF–GDA's energy graph (orange curve) descends to lower energy levels, indicating successful convergence to a more optimal solution.

Table 1 | Comparison of Dynamic Fractional generalized Deterministic Annealing (DF–GDA) with other optimization methods across key criteria, rated qualitatively as *High*, *Medium*, or *Low*

Criterion	SGD	Adam	SA	Shampoo	DF–GDA
Convergence Speed	Low	Medium	Low	High	**High**
Robustness to Noise	Low	Medium	Medium	High	**High**
Computational Efficiency	High	High	Low	Medium	**High**
Ability to Escape Local Minima	Low	Medium	High	Medium	**High**
Scalability to Large Models	High	High	Low	Medium	**High**
Stability of Updates	Low	High	Low	High	**High**
Suitability for High-Dime Spaces	High	High	Low	Medium	**High**
Adaptability to Different Data	Medium	Medium	High	Medium	**High**
Impact on Overfitting	Low	Low	Low	Medium	**High**

Methods compared: Stochastic Gradient Descent (SGD), Adam optimizer (Adam), Simulated Annealing (SA) with geometric temperature schedule, Shampoo, and DF–GDA (Proposed).

than dropout. Although dropout only prevents specific neurons from updating and primarily addresses overfitting, it does not reduce computational workload. In contrast, our method achieves both objectives: it selectively ignores a large portion of parameters, lowers computational cost, and prevents overfitting in a more adaptive way than dropout.

The proposed DF–GDA enhances robustness to annotation noise, particularly in mislabeled data, by using fractional parameter updates, soft quantization, and adaptive temperature control. By updating only a subset of parameters per iteration, DF–GDA limits the influence of noisy samples, while soft quantization smooths parameter transitions to maintain stability. Its entropy-driven temperature adjustments support broader exploration early in training, helping the model avoid suboptimal solutions caused by annotation noise.

"Nature-inspired" meta-heuristics (e.g., Genetic Algorithms, Particle Swarm, Ant-Colony) explore via large populations, use little or no gradient information, and require hand-tuned parameters for exploration versus exploitation. By contrast, DF–GDA performs deterministic, gradient-guided updates on an entropy-chosen subset of parameters and injects controlled randomness only through an adaptive temperature test. This design (i) slashes per-step cost from population-wide evaluations to a small fraction of the parameters, (ii) speeds convergence because every accepted move follows the local gradient, and (iii) self-balances exploration and exploitation via the entropy schedule. These differences remove the slow convergence, heavy computation, and parameter sensitivity that hamper classical nature-inspired methods, explaining DF–GDA's superior accuracy and efficiency in our experiments.

Table 1 provides a comparative analysis of Dynamic Fractional generalized Deterministic Annealing (DF–GDA) against widely used optimization methods, including Stochastic Gradient Descent (SGD), the Adam optimizer, Simulated Annealing (SA), and Shampoo[21]. It evaluates key performance metrics such as convergence speed, robustness to noise, computational efficiency, and the ability to escape local minima. DF–GDA consistently outperforms the other methods across these criteria, particularly excelling in convergence speed, robustness to noise, and stability of updates.

Shampoo leverages block-diagonal second-order pre-conditioning to achieve fast and stable convergence, yet incurs medium computational cost and memory overhead relative to first-order optimizers. While SGD and Adam demonstrate strengths in computational efficiency and scalability, they struggle with local minima and noise sensitivity. SA (with geometric temperature schedule), despite its capability to escape local minima, suffers from slow convergence and high computational cost. In contrast, DF–GDA employs adaptive fractional updates and entropy-driven annealing to deliver superior optimization performance, making it a highly effective alternative for complex deep-learning tasks.

The **key contributions** of this paper are as follows:

- We introduce DF–GDA, a novel optimization algorithm for deep learning that enhances convergence speed, stability, and robustness to annotation noise, outperforming traditional methods like SGD, particularly in complex problems prone to local minima.
- We propose a Dynamic Fractional Parameter Update (DFPU), an efficient algorithm integrated into DF–GDA that selectively updates model parameters based on network performance.
- We adapt GDA for deep network optimization, addressing specific challenges in deep learning.
- We validate DF–GDA through comprehensive experiments on **sixteen** diverse datasets, including image classification, healthcare, bioinformatics, and NLP, demonstrating superior convergence speed and accuracy compared to state-of-the-art and traditional optimizers. This includes the large-scale ImageNet and Kinetics-700 datasets.
- We demonstrate DF–GDA's potentials in classical machine learning tasks like SVM and k-means clustering.
- We provide a rigorous theoretical foundation supporting our methodology.

Results

Dataset

The two **large-scale datasets** used in our experiments are:

ImageNet is the canonical large-scale image classification benchmark, comprising 1.28 M training images and 50k validation images annotated across 1000 object categories. ImageNet's scale and diversity make it the primary benchmark for training visual models that generalize across tasks.

Kinetics-700 is a large-scale, curated corpus of ~ 650, 000 YouTube clips spanning 700 human-action classes that cover everyday activities, sports, and complex interactions. Roughly 536k clips are provided for training and 50k for validation, with a withheld test set for leaderboard evaluation. These clips collectively contain ~ 1.6×10^8 frames. The dataset's scope and size make it a de-facto benchmark for video representation learning.

Beyond the large-scale *ImageNet* and *Kinetics* datasets, this study employs a diverse set of benchmarks across multiple domains. The remaining datasets include classical image classification sets (**MNIST**, **MNIST-M**, **CIFAR-10**, **SVHN**, **USPS**), natural language processing benchmarks (**IMDB Sentiment**, **SMS Spam**, **Airline Sentiment**), healthcare datasets (**Breast Cancer Wisconsin**, **Heart Disease**, **Liver Patient Records**), and bioinformatics datasets (**Human Activity Recognition**, **YEAST**, **IRIS**.

Implementation details

Table 2 concisely maps each *backbone machine learning model* we optimize with DF–GDA to the broad *data modality* it tackles. We deploy lightweight CNNs (LeNet-5, a 3-layer CNN) for *small-sized image* tasks, an RBF-SVM to probe kernel methods on similar inputs, and a deep ResNet-50 for *large-scale natural-image* classification. For *spatiotemporal video* benchmarks we use 3D-ResNet-50, while sequential *sensor* data are handled with an LSTM.

Table 2 | Machine-learning backbones optimized with DF–GDA for every dataset evaluated in the paper

Backbone ML Model	Dataset(s)
LeNet-5 CNN	MNIST
RBF-SVM	MNIST (separate SVM study)
3-layer CNN	MNIST-M, CIFAR-10, SVHN, USPS
ResNet-50 (2-D CNN)	ImageNet
3D-ResNet-50 (Spatiotemp CNN)	Kinetics-700
LSTM RNN	HAR
1-D CNN (text)	IMDB Sentiment, SMS Spam, Airline Sent
Feed-forward NN (+ dropout)	Breast Cancer, Heart Disease, Liver Patient
Feed-forward NN	IRIS, YEAST

Table 3 | Relative top-1 test error on Kinetics-700 when training for 10%, 30%, 50% and the full 100% of the 100-epoch budget, the bold values are the best results

optimizer	Δ Top-1 Err. (%) ↓			
	10 %	30 %	50 %	100 %
DF–GDA (ours)	**0.0**	**0.0**	**0.0**	**0.0**
Shampoo	1.4	1.2	0.6	0.4
Adam	2.0	1.8	1.2	0.6
RMSProp	2.4	2.2	1.8	1.2
SGD	3.4	3.3	2.8	2.2
SimAnn (Geo Temp)	4.4	4.3	4.2	3.6

Values denote the increase in error over DF–GDA (lower is better).

A 1-D CNN covers *short-text sentiment* problems, and two fully-connected networks address structured *tabular biomedical* records and classic low-dimensional datasets.

In our experiments, for the dynamic fractional parameters update, we set $f_{min} = 0.01$ and $f_{max} = 0.5$. The optimal number of Markov states was 1024 and 512, depending on the datasets in our experiments. All experiments were carried out using PyTorch 1.12.1 on a server equipped with dual Nvidia RTX 3090 GPUs (24GB VRAM each), an AMD Ryzen Threadripper 3990X 64-core processor, and 256GB of RAM.

We fix $(f_{\min}, f_{\max}) = (0.01, 0.50)$ for *all* experiments. Two properties make this single pair universally effective: Because the exponent in Equation (15) uses the normalized loss change, $f(t)$ reacts to fractional progress rather than absolute loss values, yielding comparable behavior across tasks whose losses differ by orders of magnitude. With $f_{\max} \le 0.5$, the sufficient-descent condition in Theorem 3 holds for any Lipschitz-smooth objective, ensuring monotone loss decrease and convergence regardless of the dataset. The pair chosen on CIFAR-10 was *frozen* for all other benchmarks (vision, NLP, healthcare, bio-informatics) and still delivered state-of-the-art performance (Table 3). Perturbing either bound by ± 50% altered accuracy by at most 0.2 %, reinforcing the theoretical insensitivity above.

We use $T_{\max} = 5\,\sigma_\theta$ (with σ_θ the pre-training weight standard deviation); Theorem 1 ensures any $T_{\max} \gtrsim \max_i \Delta E_i$ yields the required high-entropy start, while scaling with σ_θ keeps the rule architecture-agnostic. The schedule is clipped at $T_{\min} = 0.01\,T_{\max}$; changing this to 0.005 or 0.05 affects top-1 accuracy by < 0.05 % but lengthens training, so 0.01 is retained. A constant $\lambda = 10^{-3}$ keeps the soft-quantization barrier roughly two orders of magnitude below the initial data loss, balancing bias and variance without dataset-specific tuning.

Convergence and performance analysis

Figure 2 (train on the *left*, validation on the *right*) traces loss on ImageNet under six optimizers. The proposed DF–GDA exhibits the fastest initial descent—halving its loss in fewer than fifteen epochs—and settles into a stable regime below 0.4 (train) and 0.8 (val) by epoch 90, highlighting both rapid optimization and strong generalization. Shampoo benefits from second-order curvature and eventually dips under the 1.0 threshold, yet it converges 25−30 epochs later and retains a persistent 0.3−0.4 loss gap to DF–GDA across the run. Adam and RMSProp follow similar trajectories, flattening near 0.8 train loss and 1.1−1.3 validation loss; the widening train-val gap suggests mild overfitting and reduced robustness. Classical SGD with momentum decays the slowest, underscoring the cost of uniform learning rates on deep networks. Finally, Simulated Annealing with geometric temperature schedule presents smooth but shallow progress, stalling above 2.5 validation loss despite steady training improvements, evidence that naive temperature scheduling shows its unsuitability for large-scale vision workloads.

Table 3 reports the *relative* top-1 test error of several popular optimizers with respect to our baseline **DF–GDA**. Across all training budgets—even after only 10% of the 100-epoch schedule—DF–GDA maintains a 0% error increase, confirming its superior sample efficiency. The closest competitor, Shampoo, a second-order optimization method, still lags by 1.4% early on and by 0.4% after full convergence, indicating that second-order curvature alone is insufficient to match DF–GDA's fractional annealing. First-order methods exhibit a larger gap: Adam trails by up to 2.0% and SGD by 3.4% in the under-trained regime, suggesting slower optimization dynamics. RMSProp performs better than SGD but is still behind in more modern optimization techniques. Finally, Simulated Annealing with geometric temperature schedule (SimAnn) remains consistently behind, highlighting that naive temperature schedules cannot bridge the performance deficit.

Experiments on SVM

We subsample $N = 12,000$ images (80% train, 20% validation). Baselines are LIBSVM with exhaustive (C, γ) grid-search and standard SMO optimization. DF–GDA uses $f_{\max} = 0.3$, $f_{\min} = 0.02$, $C = 10$, $\gamma_0 = 0.05$, and $\lambda_\gamma = 10^{-3}$.

Table 4 shows the obtained results for SVM. DF–GDA achieves a higher accuracy while reducing training time by over 3 × thanks to fractional updates and the elimination of grid search. The automatically annealed γ converges to the same range selected by exhaustive search, confirming the stability of our joint optimization.

Annealing temperature schedule

Figure 3 -Left illustrates the adaptive temperature schedules for different datasets when using DF–GDA, highlighting its dynamic control over the exploration-exploitation balance during training. CIFAR-10's gradual temperature decay reflects a need for extensive exploration in its complex loss landscape, while SVHN and USPS show a moderate cooling rate, indicating a balanced approach. In contrast, MNIST and MNIST-M rapidly decrease their temperatures, quickly transitioning to exploitation due to their simpler structures. These patterns underscore DF–GDA's adaptability, efficiently optimizing its behavior to suit each dataset's characteristics, thus ensuring robust and accelerated convergence across varying data complexities.

Table 5 compares the entropy-controlled schedule with (i) geometric cooling and (ii) a fixed temperature on ImageNet (ResNet-50). Our schedule attains 80% top-1 accuracy in only 62 epochs versus 99 (geometric) and 147 (fixed), and delivers the best final accuracy.

Dynamic fractional update

Figure 3 -Right shows the analysis of parameter update fractions across different datasets reveals DF–GDA's adaptive optimization strategy. The evolution of these fractions is visualized through both a line plot showing epoch-wise changes and a bar chart (far-Right) summarizing average utilization across the training period. For complex datasets like CIFAR-10 and SVHN, the algorithm starts with high parameter update fractions (0.25) that

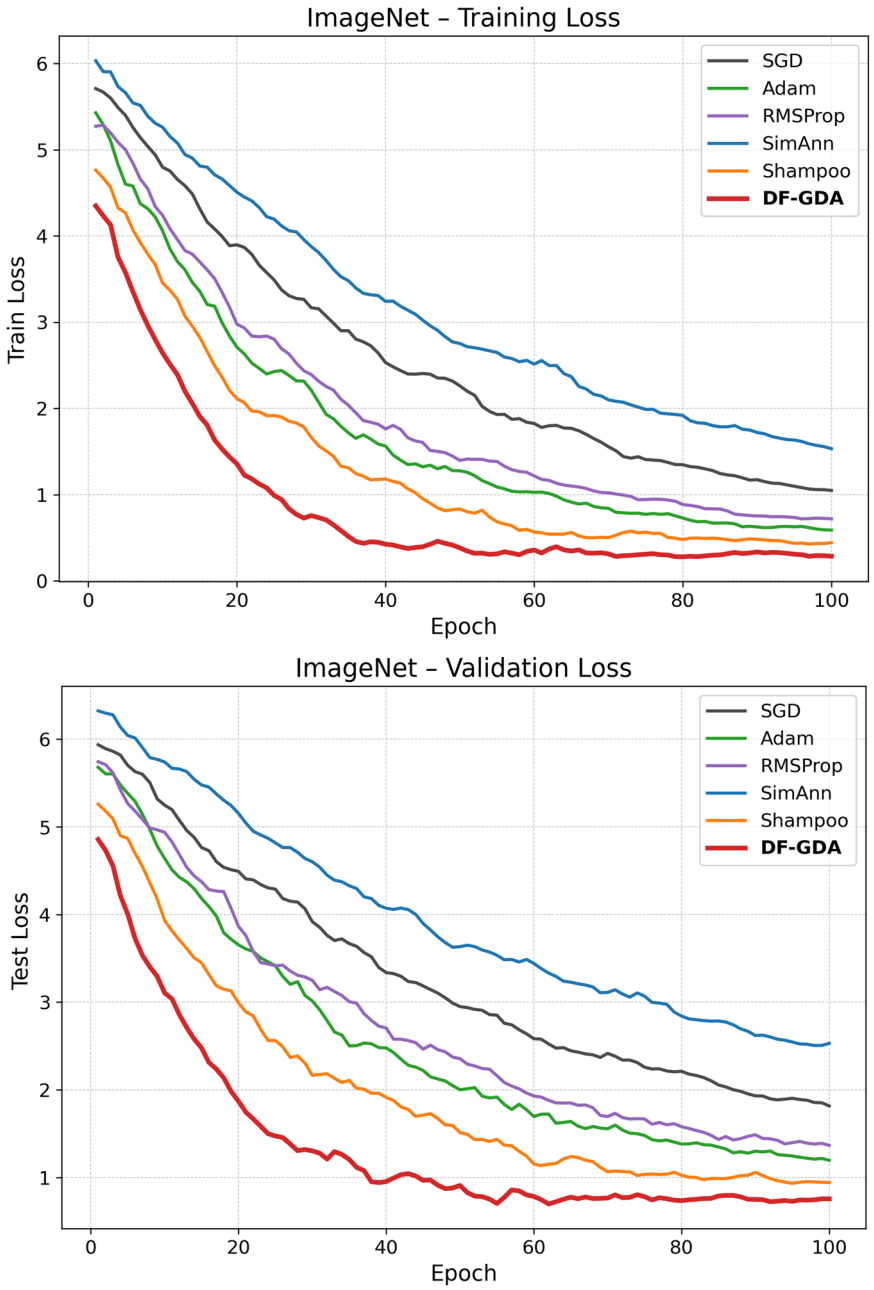

Fig. 2 | Comparison of training (left) and validation (right) cross-entropy loss between our proposed DF–GDA and five competing optimizers on ImageNet (100 epochs). DF–GDA descends fastest and converges to the lowest loss, with second-order Shampoo a distant second and first-order and standard simulated annealing baseline (SimAnn with geometric schedule) further behind.

Table 4 | RBF-SVM on MNIST. Mean of three runs, the bold values are the best results

Method	Val. Acc. (%)	Train time (s)	γ chosen
SMO + grid–search	97.4	124	0.060
DF–GDA (ours)	**98.7**	**37**	0.057 ± 0.004

gradually decrease, while maintaining relatively higher average fractions throughout training to handle their inherent complexity. MNIST-M shows similar initial behavior due to its noisy characteristics. In contrast, simpler datasets like USPS and MNIST exhibit rapid reductions in parameter update fractions, stabilizing at lower values by the sixth epoch, indicating efficient early convergence. This dynamic adjustment demonstrates DF–GDA's ability to automatically tune its update strategy based on dataset complexity, optimizing computational efficiency by reducing unnecessary parameter updates while maintaining exploration where needed.

State-space complexity

Figure 4 -Right shows training and validation loss over ten epochs for models with different state sizes. All models exhibit rapid convergence, with initially higher losses for larger states but similar final performance across configurations. This suggests that smaller models may achieve comparable accuracy with reduced computational demands, making them more efficient for deployment.

Computational efficiency

Table 6 reports two runtime metrics: (i) average wall-clock time per ImageNet epoch (per-step cost) and (ii) total hours to reach the *80% top-1 accuracy milestone* for ResNet-50—high enough to mark competitive performance yet attainable by all baselines. Although DF-GDA's epoch is ~ 12% longer than SGD's, its sharper loss decline allows it to hit the 80% milestone 4–6 hours sooner than first-order baselines, more than twice as fast as the second-order Shampoo, and over nine-fold faster than Simulated Annealing. Table 6 also indicates the final

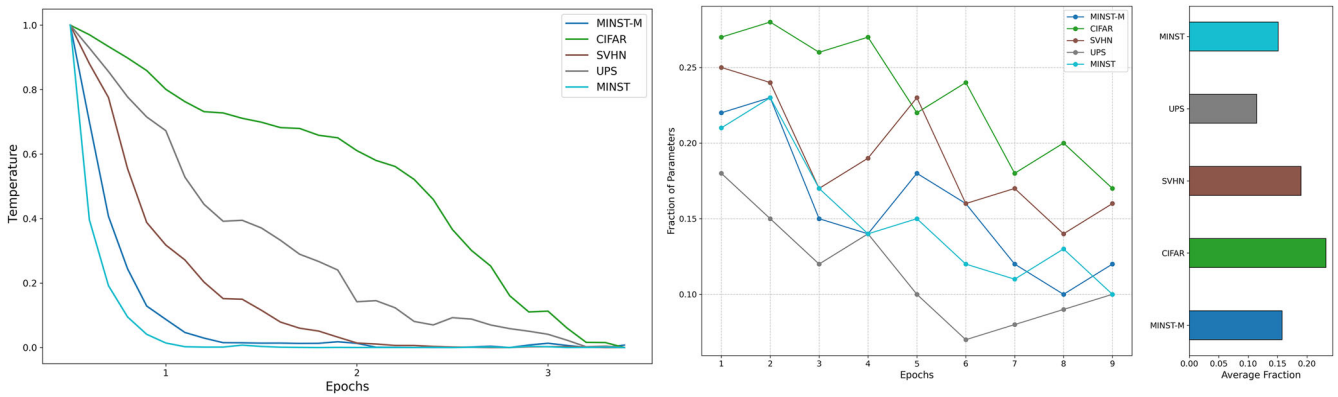

Fig. 3 | Temperature schedules and fractions. Left: The adaptive annealing temperature schedules for five datasets, MNIST, MNIST-M, CIFAR, SVHN, and USPS. Right: The fraction of parameters used during different training epochs for five datasets (MNIST, MNIST-M, CIFAR, SVHN, and USPS). Far-Right: The average fraction of parameters used across different epochs for the datasets.

Table 5 | Temperature-schedule ablation on ImageNet (ResNet-50), the bold values are the best results

Schedule	Epochs to 80 %	Final top-1 (%)	Relative speed
Entropy-controlled (ours)	**62**	**79.4**	-
Geometric ($\gamma = 0.95$)	99	78.7	1.6 × slower
Fixed ($T = T_{max}$)	147	78.2	2.4 × slower

accuracy of different methods, showing our method outperforms others.

Discussion

DF–GDA has also been evaluated across multiple interdisciplinary datasets, demonstrating the consistent superiority of the DF–GDA approach in several domains. On fundamental datasets like MNIST and USPS (Fig. 5-Left and Fig. 6-Right), DF–GDA exhibited rapid convergence within the initial epochs, achieving stable and low training and validation losses, while SGD required significantly more epochs to reach comparable performance. This pattern extended to more complex datasets, including MNIST-M (Fig. 6-left) and SVHN, where DF–GDA's structured updates effectively handled the inherent noise and transformations, maintaining consistently lower losses compared to SGD. The algorithm's robustness was further validated on the challenging CIFAR-10 dataset (Fig. 5-Middle), where DF–GDA's effectiveness in high-dimensional data optimization was evident through its rapid descent to lower training and validation losses. To ensure a fair comparison, we extended the training beyond DF–GDA's early convergence points, using SGD's first significant loss drop as a benchmark. Even in this extended analysis, DF–GDA maintained superior performance across all datasets, suggesting better navigation of the loss landscape and reduced susceptibility to local minima.

Figure 7 illustrates the comparative performance of the proposed DF–GDA optimization algorithm against SGD across various interdisciplinary datasets in the fields of Bioinformatics, Healthcare, and NLP, with training loss plotted over five epochs. This provides information on the effectiveness of DF–GDA during the early stages of training.

In biological datasets (Human Activity Recognition, YEAST, and IRIS), DF–GDA consistently demonstrates superior convergence compared to SGD. Most notably in the YEAST dataset, DF–GDA achieves a significantly lower training loss (approximately 0.5) compared to SGD (around 1.4) by epoch 5. The performance gap is particularly pronounced after epoch 2, where DF–GDA shows rapid convergence while SGD exhibits a more gradual descent in training loss. In healthcare applications (Breast Cancer Wisconsin, Heart Disease, and Liver Patient Records), DF–GDA maintains its advantage over SGD across all three datasets. The Heart Disease dataset results are particularly noteworthy, where DF–GDA achieves stable convergence at a training loss of approximately 0.2 after epoch 2, while SGD shows fluctuations and settles at a higher loss value around 0.4. The Liver Patient Records dataset similarly demonstrates DF–GDA's faster convergence and lower final training loss. For NLP tasks (IMDB Sentiment, SMS Spam, and Airline Sentiment), DF–GDA shows consistent superiority in convergence speed and final training loss. The contrast is most evident in the SMS Spam dataset, where DF–GDA achieves a steady decrease in training loss to approximately 0.2, while SGD plateaus at around 0.4. The IMDB Sentiment analysis shows both methods achieving very low training loss values, but DF–GDA reaches convergence more rapidly, particularly between epochs 1 and 2.

DF–GDA offers robustness to annotation noise, such as incorrect class labels in image recognition, through the following key aspects of its design:

- *Fractional parameter updates:* DF–GDA limits the impact of noisy data by updating only a fraction of the parameters in each iteration. Unlike traditional methods that globally adjust all parameters, this localized update strategy prevents the model from being overly influenced by mislabeled samples.
- *Soft quantization:* Soft quantization ensures smooth transitions in parameter states, reducing sensitivity to fluctuations caused by annotation noise. This approach maintains stability during training by keeping parameter adjustments more controlled.
- *Energy-based acceptance:* DF–GDA's probabilistic acceptance function allows it to occasionally accept suboptimal solutions based on energy differences, bypassing noise-induced local minima. This feature enables the model to explore more effectively in noisy environments.
- *Entropy-driven temperature control:* The dynamic temperature adjustment, based on parameter state entropy, keeps the model adaptive, enhancing its ability to manage mislabeled data. High entropy maintains a broader exploration, reducing the likelihood of premature convergence on incorrect solutions.

The experimental results in Table 7 demonstrate the superior robustness of DF–GDA in the presence of annotation noise, compared to SGD across multiple datasets. When testing with artificially introduced label noise ranging from 5% to 20%, DF–GDA consistently shows smaller performance degradation than SGD. At 5% noise, DF–GDA's performance drops by only 0.9% and 1.1% for USPS and MNIST, respectively, compared to SGD's 1.7% and 1.8%. Even at high noise levels (20%), DF–GDA maintains its advantage, showing a 15.4% drop on CIFAR versus SGD's 23.1%. This enhanced noise resilience, observed consistently across datasets including challenging ones like MNIST-M, demonstrates that DF–GDA's

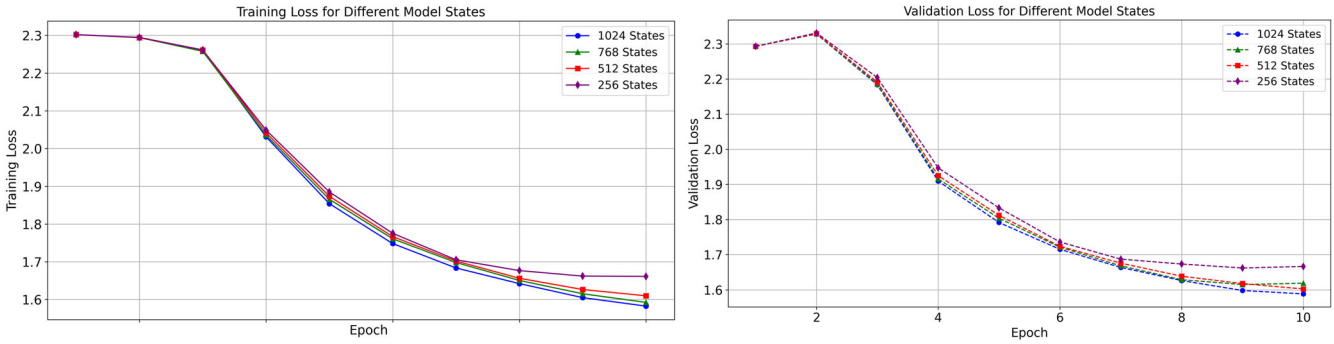

Fig. 4 | The correlation between the number of states and training and validation losses on the CIFAR dataset.

Table 6 | Average wall-clock time per ImageNet epoch and time to reach 80 % validation accuracy (ResNet-50, batch 512, mixed precision, dual RTX 3090), reordered by final accuracy, the bold values are the best results

optimizer	Time / epoch (min)	Time to 80% acc. (h)	Final acc
DF–GDA (ours)	26.2	**14.9**	**79.4**
Shampoo (block-diag. 2nd-order)	41.5	32.0	77.98
Adam (β_1 = 0.9, β_2 = 0.999)	24.1	19.0	76.01
RMSProp (ρ = 0.9)	24.8	21.0	75.42
SGD (momentum 0.9)	23.4	20.1	71.46
Simulated Annealing (geom. T)	67.0	135.0	66.17

structured, adaptive approach effectively prevents convergence to sub-optimal states in the presence of noisy labels.

Looking ahead, we anticipate that DF-GDA's adaptive, fractionally annealed optimization will accelerate training of safety-critical AI systems, ranging from protein-folding predictors to autonomous-driving perception stacks, while preserving the robustness gains demonstrated here.

Methods

Optimization is a fundamental task in machine learning and deep learning models, where the goal is to minimize a loss function $f(\theta)$ with respect to the model parameters θ. Gradient-based optimization methods are widely employed for this purpose, leveraging the gradient (first-order derivatives) of the loss function to update the parameters iteratively.

Gradient-based Methods

The basic *gradient descent* (GD) algorithm[22] computes the gradient of the loss function $f(\theta)$ with respect to the parameters and updates the parameters in the direction that decreases the loss with a learning rate η. The update rule is given by:

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

Despite its simplicity, GD can become inefficient for large datasets, requiring a complete pass over the entire dataset at each iteration. To address this challenge, we move to *stochastic gradient descent* (SGD)[23], which offers a more efficient alternative. SGD computes the gradient based on a single randomly chosen data point (or a small batch of data), significantly reducing the computational cost per iteration. While SGD improves efficiency, the noisy updates can lead to instability in convergence. To mitigate this instability, researchers often employ *mini-batch gradient descent*[24], a compromise between GD and SGD that further stabilizes the optimization process. Mini-batch gradient descent computes the gradient over a small batch of data points B, offering a balance between the computational efficiency of SGD and the stability of GD. The update rule becomes:

$$\theta_{t+1} = \theta_t - \eta \frac{1}{|B|} \sum_{i \in B} \nabla_\theta f(\theta_t; x_i) \tag{2}$$

While this variant improves efficiency and stability, particular optimization challenges remain, remarkably when the gradient oscillates or slows down near optima. Techniques such as *momentum*[25] are introduced to address these. Momentum accelerates convergence by smoothing the update direction using an exponentially decaying average of past gradients, where the degree of influence from past gradients is controlled by the weighting factor β. This allows the optimizer to overcome oscillation and gain speed in the proper direction. The update rule with momentum is:

$$v_{t+1} = \beta v_t + (1 - \beta) \nabla_\theta f(\theta_t) \tag{3}$$

Simulated annealing

Although gradient-based methods are highly effective for many optimization tasks, they can struggle with nonconvex problems where multiple local minima exist, potentially leading to suboptimal solutions. Nonconvexity is a common characteristic of many modern problems, particularly deep learning. In such cases, *simulated annealing* (SA)[26] offers an alternative by allowing probabilistic exploration of the solution space, which helps in escaping local minima and finding better global optima. SA is a probabilistic technique for approximating the global optimum of a given function, inspired by the physical annealing process in metallurgy[27]. The core idea is to explore the solution space randomly at high temperatures, allowing uphill moves (increases in the objective function) to avoid local minima. As the temperature decreases, the algorithm gradually favors downhill moves, leading to convergence to a local or global minimum. The probability of moving from solution i to j at temperature T follows:

$$P(i, j, T) = \exp\left(\frac{E(i) - E(j)}{T}\right) \tag{4}$$

where $E(i)$ and $E(j)$ represent the energies (or costs) of solutions i and j, respectively.

Generalized deterministic annealing

SA has been successfully applied to numerous nonconvex optimization problems, but its stochastic nature, serial updating, and associated computational cost can make it impractical for large-scale problems[28,29]. Precisely due to its stochastic nature, it often requires a significant number of iterations to converge[28]. In fact, to guarantee a global optimum, the annealing schedule is as expensive as an exhaustive search of the solution space[30]. Furthermore, it struggles with the high computational cost of

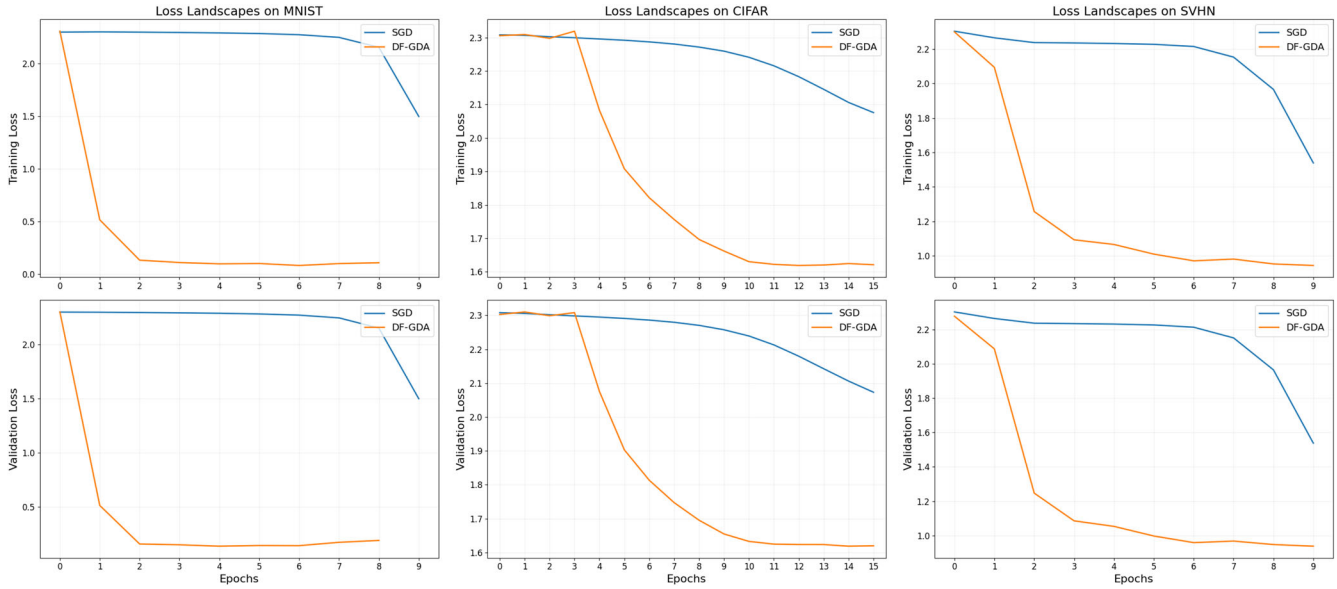

Fig. 5 | Comparison of training and validation losses between SGD and DF–GDA on the image classification datasets, MNIST, CIFAR, and SVHN datasets.

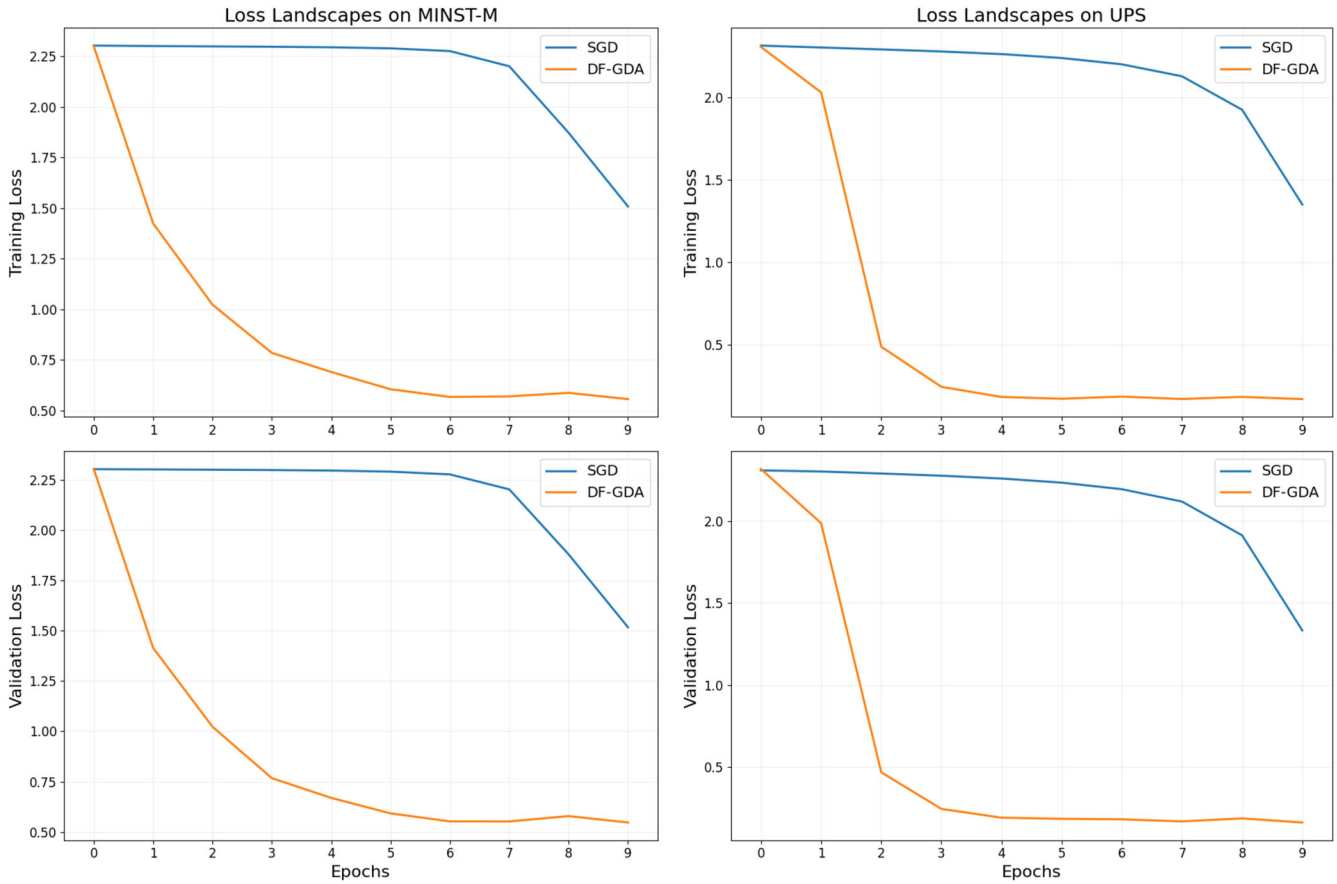

Fig. 6 | Comparison of training and validation losses between SGD and DF–GDA on the MINST-M (Left) and USPS (Right) dataset.

maintaining random sampling over vast solution spaces and can be prone to erratic convergence behavior in some instances[29]. To overcome the inefficiencies and convergence issues of SA, *generalized deterministic annealing* (GDA)[15] was introduced as a more efficient, deterministic alternative. While GDA retains the core principles of SA, such as temperature-dependent exploration of the solution space, it replaces the stochastic updates with deterministic rules that reduce computational complexity. By utilizing local Markov chains, GDA transitions between solutions more systematically, leading to faster convergence and avoiding the erratic behavior sometimes associated with SA.

GDA employs *K-state* neurons to represent the probability densities of local Markov chains, iteratively updating these densities based on transition

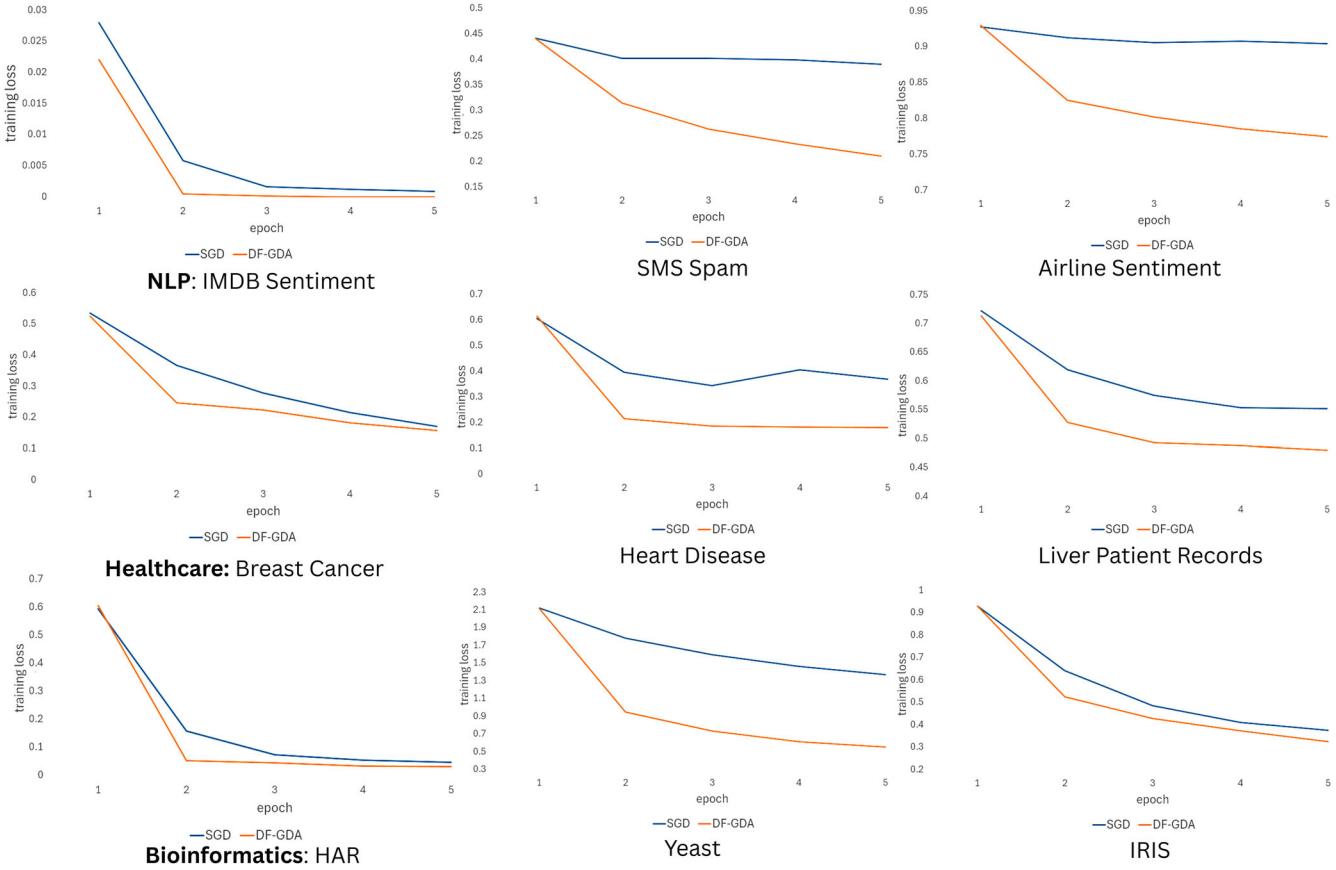

Fig. 7 | Comparison of training losses between SGD and DF–GDA on interdisciplinary benchmarks: Natural Language Processing (IMDB Sentiment, SMS Spam, and Airline Sentiment), Healthcare (Breast Cancer Wisconsin, Heart Disease, and Liver Patient Records), and Bioinformatics (Human Activity Recognition, Yeast, and IRIS).

probabilities. This iterative process is captured by the update equation:

$$\pi_n^{t+1}(j) = \sum_{i=0}^{K-1} P_n(i, j, T)\pi_n^t(i) \tag{5}$$

where $\pi_n^t(j)$ is the probability density of the nth neuron being in state j at iteration t, and $P_n(i, j, T)$ is the transition probability at temperature T. An acceptance function governs these transitions, ensuring that lower-energy states are favored as the temperature decreases. The transition probability balances exploration and exploitation during optimization. This probability is determined by two key components: *the generation function*, which proposes new candidate states, and the *acceptance function*, which decides whether to accept the new state based on the change in energy (or loss) and the current temperature. The transition probability from state i to j at temperature T is given by:

$$P(i, j, T) = G(i, j) \cdot A(i, j, T) \tag{6}$$

This mechanism enables GDA to converge to high-quality solutions more efficiently than SA. While SA explores the entire state space and requires $O((KN)^2)$ steps for convergence, GDA achieves the same with $O(KN)$ updates by focusing on localized Markov chains and deterministic updates, significantly reducing computational complexity. This makes GDA particularly well-suited for large-scale optimization problems where local constraints dominate. Empirical results demonstrate that GDA outperforms both SA and local search methods regarding solution quality and computational efficiency[15,31].

GDA for deep learning optimization. The original GDA algorithm effectively solved discrete optimization problems like image restoration[15,31] through deterministic state transitions that minimise energy functions. However, modern deep learning presents new challenges with its continuous, high-dimensional parameter spaces. To adapt GDA for these contexts, we introduce two key modifications: soft quantization, which enables probabilistic parameter representation in continuous spaces, and dynamic fractional updates, allowing simultaneous adjustment of multiple parameters. These enhancements preserve GDA's exploratory capabilities while improving its efficiency and scalability for deep learning applications.

Dynamic fractional generalized deterministic annealing method

Training deep neural networks presents significant challenges due to the nonconvex nature of the loss landscape, which is characterized by numerous local minima, saddle points, and flat regions. Standard optimization methods, such as SGD, can converge slowly or become trapped in suboptimal solutions, particularly when applied to large models. To address this issue, we propose a new method, namely, a dynamic fractional generalized deterministic annealing (DF–GDA) algorithm based on the principles of the GDA algorithm[15] opting for a deterministic approach that allows the acceptance of solutions with higher loss values during the early stages of training. This controlled tolerance of the loss function facilitates broader exploration of the solution space, helping the optimizer to escape local minima and improve overall convergence. The proposed DF–GDA elevates the original GDA algorithm by integrating a novel dynamic fractional parameter update mechanism and soft quantization to enhance computational efficiency and convergence speed, making it more compatible with modern deep learning models. By dynamically adjusting the fraction of parameters updated and applying soft quantization, DF–GDA allows for a more controlled exploration of the parameter space, balancing exploration and exploitation to accelerate convergence.

Table 7 | Comparison of the impact of varying annotation noise levels on performance reduction percentages across different datasets using SGD and DF–GDA optimization methods

Dataset	Noise Level (%)	SGD	DF–GDA
MNIST	5	1.8	1.1
	10	4.6	3.2
	15	7.0	5.1
	20	12.9	8.4
MNIST-M	5	2.8	1.9
	10	7.0	5.4
	15	9.7	7.1
	20	15.1	11.0
CIFAR	5	5.2	2.7
	10	10.8	6.3
	15	14.7	9.9
	20	23.1	15.4
SVHN	5	3.6	2.1
	10	8.2	5.5
	15	11.6	8.0
	20	18.2	12.3
USPS	5	1.7	0.9
	10	4.3	2.9
	15	6.6	4.2
	20	11.2	7.1

Lower percentages indicate smaller performance drops and better results.

Let $\theta \in \mathbb{R}^n$ represent the set of parameters in the neural network, where n is the total number of parameters. The goal is to minimise a nonconvex loss function $L(\theta)$, which measures the error between the model's predictions and the ground truth:

$$\min_{\theta} L(\theta) \tag{7}$$

Due to the nonconvex nature of $L(\theta)$, standard gradient-based methods are prone to being trapped in local minima. DF–GDA uses a temperature T and dynamically adjusts the fraction of parameters updated at each iteration to balance exploration and exploitation.

In DF–GDA, the optimization process is guided by the energy function $E(\theta, T)$, a function of the loss $L(\theta)$ and the temperature T. The energy function is expressed as:

$$E(\theta, T) = L(\theta) + \frac{\lambda}{T}\sum_{i=1}^{n} |\theta_i - \theta_i'| \tag{8}$$

where n is the number of parameters, λ is a regularization parameter, and θ_i' are the potential new states of the parameters after applying soft quantization (discussed below).

DF–GDA's pipeline

DF–GDA incorporates a novel *soft quantization* strategy to constrain parameter updates and ensure smoother transitions. For each parameter θ_i, soft quantization projects the parameter onto a set of K quantized states $Q = \{q_1, q_2, \ldots, q_K\}$, where the probability of θ_i assuming a quantized state q_k is given by:

$$S(\theta_i) = \sum_{k=1}^{K} q_k \cdot \frac{\exp\left(-\frac{|\theta_i - q_k|}{T}\right)}{\sum_{j=1}^{K} \exp\left(-\frac{|\theta_i - q_j|}{T}\right)} \tag{9}$$

where T controls the quantization level between soft and hard. Higher temperatures result in softer (less sharp) quantization, allowing parameters to explore a broader range of values. As the temperature decreases, the quantization becomes sharper, making the parameter updates more deterministic. Soft quantization balances continuous space optimization with the stability of discrete space updates, preventing large, abrupt parameter jumps in high-dimensional problems like deep neural networks. Unlike hard quantization, which forces parameters to snap to the nearest state, soft quantization allows them to probabilistically explore nearby states, promoting smoother transitions while maintaining structure and stability. This approach enhances both flexibility and robustness in parameter updates.

Soft quantization shares a mathematical resemblance to the softmax function, as both use an exponential normalization term. However, while softmax is primarily used for probability distribution in classification, soft quantization ensures smooth transitions between discrete quantized states in optimization.

For problems requiring high precision across a wide parameter range, a larger K allows for finer resolution in the parameter space. Conversely, a smaller K is suitable for less complex tasks or when prioritizing computational efficiency, resulting in coarser exploration.

At each iteration, the derivative of the loss function $\nabla L(\theta_i)$ is updated using the *mean field* approach to smooth over time:

$$\mu_i^{(t+1)} = \mu_i^{(t)} + (1-\alpha)\nabla L(\theta_i^{(t)}) \tag{10}$$

where α is a smoothing coefficient that controls how much current values are weighted versus the historical average.

Once the mean field is computed, the parameters are updated by applying soft quantization:

$$\theta_i^{(t+1)} = S\left(\theta_i^{(t)} - \eta\mu_i^{(t)} + \epsilon(T)\cdot\mathcal{N}(0, I)\right) \tag{11}$$

where η is the learning rate. $\epsilon(T)$ is a temperature-dependent scaling factor. The term $\epsilon(T)\cdot\mathcal{N}(0, I)$ introduces Gaussian noise (with mean 0 and identity covariance I), scaled by temperature T. $P(i, j, T)$ is the transition probability that will be explained as follows. Early in training, this noise helps explore the parameter space, allowing the optimizer to break out of flat regions of the energy/loss landscape. As T decreases, the noise diminishes, making the updates more deterministic and focused on fine-tuning the parameters. The soft quantization operator $S(\cdot)$ projects the updated parameter onto discrete states, ensuring stable and controlled convergence. This approach balances exploration and exploitation, leading to precise optimization as training progresses.

The acceptance function $A(\theta_i, \theta_j, T)$ is a sigmoidal function of the energy difference between the current state θ_i and the proposed state θ_j:

$$A(\theta_i, \theta_j, T) = \frac{1}{1 + \exp\left(\frac{E(\theta_j) - E(\theta_i)}{T}\right)} \tag{12}$$

This acceptance criterion ensures that the move is always accepted if the energy (loss) at the proposed state θ_j is lower than at θ_i. If the energy at θ_j is higher, the move is accepted with a probability that decreases with both the energy difference and the temperature.

Practical design of soft quantization

Equation (9) can be rewritten in Gibbs form,

$$S(\theta_i) = \sum_{q\in\mathcal{Q}} q\,\pi_i(q),\ \pi_i(q) \propto \exp\left[-E_i(q)/T\right],\ E_i(q) = |\theta_i - q|,$$

revealing that soft quantization is *thermodynamically* equivalent to a Boltzmann sampler over a discrete surrogate energy $E_i(\cdot)$. Its contribution is two-fold:

(i) *Stability & implicit regularization*. The projection $S(\theta_i)$ contracts the update $\theta_i \leftarrow \theta_i - \eta\mu_i + \varepsilon(T)\mathcal{N}$ onto a *convex hull* of $\mathcal{Q}$, preventing large jumps in high-curvature regions and acting as a temperature-controlled weight decay.

(ii) *Exploration*. At high T each $\pi_i(q) \approx 1/|\mathcal{Q}|$, recovering the "trivial state" required by Theorem 1 for broad search; as $T\downarrow$ the distribution sharpens and the operator morphs into a hard nearest-neighbour projection, thus turning stochastic search into deterministic fine-tuning (Theorem 2).

Let σ_{init} denote the standard deviation of the parameter initialization distribution (e.g. Kaiming). We choose

$$\mathcal{Q} = \{m \cdot \Delta \mid m = -(K-1)/2, \ldots, (K-1)/2\}, \qquad \Delta = \kappa\,\sigma_{\text{init}},$$

with an odd K so that $0 \in \mathcal{Q}$. Empirically, $\kappa \in [0.5, 1]$ keeps $\max_{q\in\mathcal{Q}} |\theta_i - q|$ within one standard-deviation of typical weights, ensuring that (i) the high-T uniform condition of Theorem 1 is satisfied, and (ii) gradients remain well scaled after quantization. In practice, we use K = 256 for small/medium networks and K = 1024 for large-scale ImageNet/Kinetics runs; we proved that larger K does not harm convergence, but brings diminishing returns once $K > 256$.

DF–GDA proposes adjusting the temperature dynamically based on the *total entropy* of the parameter space. The entropy $H(\theta)$ at iteration t is defined as:

$$H(\theta^{(t)}) = -\sum_{i=1}^{n}\sum_{k=1}^{K} P(\theta_i = q_k)\log P(\theta_i = q_k) \tag{13}$$

where $P(\theta_i = q_k)$ is the probability of parameter θ_i being in the quantized state q_k, given by the soft quantization function.

The temperature is updated based on the ratio of the current entropy to the maximum entropy:

$$T(t+1) = T_{\max} \cdot \frac{H(\theta^{(t+1)})}{H_{\max}} \tag{14}$$

where $H_{\max}$ is the maximum entropy observed early in training, and $T_{\max}$ is the initial temperature. This ensures that as the entropy decreases (i.e., the model becomes more confident), the temperature decreases, transitioning the optimization from exploration to exploitation.

At high initial temperatures T_0, the soft quantization function assigns equal (uniform) probabilities to all K quantized states, i.e., $S(\theta_i) \approx \frac{1}{K}$ for all k. This occurs as the exponential in the probability function flattens, ensuring broad exploration and preventing premature convergence. As T decreases, the function shifts to favor optimal states, balancing exploration and exploitation.

Adaptive temperature schedule

We control the temperature through the empirical entropy of the soft-quantization weights, yielding a single-line update, stated in Equation (13).

This entropy-controlled schedule (i) satisfies the monotone cooling assumptions of Theorems 1-2, (ii) adapts automatically to model size and task difficulty without extra hyper-parameters, and (iii) reduces to deterministic nearest-neighbour projection once $T_t \le T_{\min} = 0.01\,T_{\max}$, at which point each π_i is > 0.98 concentrated on its mode. Figure 3 (left) illustrates a typical trajectory, showing rapid early exploration followed by smooth convergence.

At iteration 0 the entropy H_0 is maximal, so $T_0 = T_{\max}$ enables large stochastic moves that explore multiple basins of the loss surface. As training proceeds H_t shrinks, and the schedule $T_{t+1} = T_{\max} H_t / \log|\mathcal{Q}|$ cools *proportionally*, progressively sharpening the landscape until $T_t \le T_{\min} = 0.01\,T_{\max}$, where DF-GDA behaves as a deterministic fine tuner. Thus a single parameter, $T_{\max}$, *self-balances* global exploration and local refinement without manual tuning.

The *entropy-controlled* temperature schedule provides three main advantages over a standard geometric schedule: it is *hyper-parameter-free and automatically adapts* to model size and task difficulty, it *monotonically cools* in a way that preserves the convergence guarantees of DF–GDA, and it *accelerates training* by spending more of the optimization budget in a low-temperature, deterministic fine-tuning phase. Its chief trade-offs are a modest (≈ 3%) computational overhead for entropy computation and a potential sensitivity to rare plateaus where entropy drops prematurely—mitigated in practice by clamping the temperature at $T_{\min} = 0.01\,T_{\max}$. Overall, the adaptive schedule's gains in convergence speed, accuracy, and noise robustness outweighs these minor costs, making it a sensible default for DF–GDA.

Dynamic fractional parameter update

Traditional optimization methods, such as SA or GD, which update all parameters at each iteration, suffer from high computational costs and inefficiency when scaling to large models. Meanwhile, GDA updates only one parameter per iteration, which leads to slow convergence. To address these limitations, we propose the dynamic fractional parameter update framework, which dynamically adjusts the fraction of parameters updated at each iteration. This fraction is controlled based on the loss dynamics, allowing for more efficient updates while maintaining the exploratory benefits of annealing.

Rather than updating all parameters at each iteration, DF–GDA updates a fraction $f(t)$ of the parameters, where $f(t)$ is dynamically adjusted based on the recent changes in the loss function. The fraction is defined as:

$$f(t) = f_{\min} + (f_{\max} - f_{\min}) \cdot \exp\left(-\frac{\Delta L(t)}{\max(\Delta L)}\right) \tag{15}$$

where $f_{\min}$ and $f_{\max}$ are the minimum and maximum fractions of parameters to be updated, respectively. $\Delta L(t) = |L(t) - L(t-1)|$ is the change in the loss between consecutive iterations. $\max(\Delta L)$ is the maximum observed loss change used for normalization. This ensures that a larger fraction of parameters is updated early in training when loss changes are significant. As the loss stabilizes, fewer parameters are updated, encouraging fine-tuning in the later optimization stages.

Furthermore, DF–GDA incorporates a blockwise fractional sampling strategy for parameters, where each training iteration operates on a block of the parameters, ensuring that all parameters are updated by the end of each epoch. In the blockwise fractional approach, the model's parameters set θ is divided into B non-overlapping blocks $\theta = \{\Theta_1, \Theta_2, \ldots, \Theta_B\}$, where each block Θ_b contains a fraction of the total samples. At each iteration t, only one block of parameters Θ_b is updated during training, and by the end of each training epoch, all the blocks are updated, ensuring that all parameters are covered. This approach reduces the computational load per iteration and increases memory efficiency.

Let $B = \lceil 1/f_{\min} \rceil$ and $\Theta = \{\Theta_1, \ldots, \Theta_B\}$ be a size-balanced, non-overlapping partition of the parameter vector obtained by greedily accumulating tensors until each block reaches $\lceil \|\theta\|/B \rceil$ scalars (large kernels are split along the channel axis when needed). At every iteration we update the single block whose index $\pi_t(b)$ is drawn from a fresh random permutation π_t generated at the beginning of the current epoch; hence every parameter is visited exactly once per epoch and with probability $f(t)$ at step t. This schedule preserves the unbiasedness of the stochastic gradient and satisfies the sufficient-descent condition of Theorem 3, and, by keeping only one block resident in GPU memory, reduces the *per-step* complexity of DF–GDA to $\mathcal{O}\big(f(t)\,nK\big)$ without altering its convergence guarantee.

Equation (15) endows DF–GDA with a *time-varying update rate*: at the start of training the loss drops sharply, so $f(t) \approx f_{\max} = 0.5$ and ~ 50 % of the parameters follow the gradient each step, yielding fast descent. As soon as $\Delta L(t)$ falls below 1 % of its initial value, $f(t)$ contracts exponentially towards $f_{\min} = 0.01$, leaving only a 1 % subset to be fine-tuned. Combined with the block-wise schedule, this shrinks the per-step cost to $\mathcal{O}\big(f(t)\,nK\big)$ and is the main reason DF–GDA reaches the 80 % ImageNet milestone 4–6

h sooner than strong first-order baselines (see Table 6 and the discussion following Theorem 3).

Computational efficiency & complexity analysis

Let n be the number of trainable parameters, K the number of soft-quantization states (a small constant; $K = 512$ in all experiments), and $f(t) \in (0, 1]$ the dynamically chosen *update fraction* at iteration t with $k(t) = \lfloor f(t)\, n \rfloor$.

Each training step consists of the usual back-propagation ($\mathcal{O}(\text{backprop})$) and an annealing overhead unique to DF–GDA:

$$T_{\text{DF-GDA}}(t) = k(t) \underbrace{\mathcal{O}(K)}_{\substack{\text{soft-quantization} \\ +\text{acceptance test}}} = \mathcal{O}\big(f(t)\, n\, K\big).$$

- *Worst case* ($f(t) = 1$, first few epochs): $\mathcal{O}(nK)$.
- *Typical/late training* ($f(t) \to 0.02$): $\mathcal{O}(0.02\, nK)$, yielding a $> 50\times$ speed-up over classical SA that updates all parameters throughout.

DF–GDA stores (i) the parameter vector $\theta \in \mathbb{R}^n$, (ii) a same-size running mean μ, and (iii) one scratch vector of length K reused across parameters. Hence

$$S_{\text{DF-GDA}} = \underbrace{\mathcal{O}(n)}_{\theta,\mu} + \underbrace{\mathcal{O}(K)}_{\text{scratch}} = \mathcal{O}(n) \quad (K \ll n).$$

Classical SA implementations that cache a full probability matrix incur $\mathcal{O}(nK)$ memory, while adaptive optimizers such as Adam require an extra $\mathcal{O}(n)$ variance buffer—placing DF–GDA among the most memory-efficient choices.

The analysis above, together with Table 8, demonstrates that DF–GDA achieves *linear* time and memory scaling in the model size, making it suitable for modern, large-scale deep networks.

Classical simulated annealing proposes one neighboring state at every step; exploring the $K \times N$ configuration graph of a modern network therefore requires $\mathcal{O}\big((KN)^2\big)$ moves. Equation (5) transforms this stochastic walk into a *deterministic probability flow*: all K states of each neuron are updated simultaneously, shrinking the search to $\mathcal{O}(K)$ operations per parameter and reducing the overall annealing pass to $\mathcal{O}(KN)$. Coupled with the fractional-update rule, the complexity becomes $\mathcal{O}\big(f(t)\, nK\big)$, giving DF-GDA the same asymptotic cost as first-order optimizers while preserving the ability to escape poor basins.

Figure 8 illustrates the efficiency of the DF–GDA algorithm, showcasing its blockwise dynamic fractional parameter update method and the convergence of all the data samples during an epoch. Unlike traditional optimization algorithms that update all parameters simultaneously, DF–GDA selectively updates a fraction of the parameters at each iteration. This selective updating strategy significantly reduces computational costs while maintaining high optimization efficiency, leading to faster convergence and improved stability in deep learning models.

Algorithm 1 summarizes the proposed DF–GDA, including all the steps discussed so far in the paper.

Algorithm 1. DF–GDA Algorithm

Require: Initial parameters $\theta \in \mathbb{R}^n$, dataset $\mathcal{D}$, initial temperature $T_{\max}$, minimum and maximum fraction $f_{\min}, f_{\max}$, sensitivity factor α, learning rate η, regularization parameter λ, and maximum number of iterations N.

Initialise mean field derivatives $\mu_i = 0$ for all $i = 1, 2, \ldots, n$
Set initial temperature $T \leftarrow T_{\max}$
Set maximum entropy $H_{\max}$ based on the initial state distribution
Divide parameters θ into B blocks $\{\Theta_1, \Theta_2, \ldots, \Theta_B\}$
for each epoch $e = 1, 2, \ldots, E$ **do**
  Shuffle dataset $\mathcal{D}$ and divide into B blocks
  for each block $\mathcal{B}_b \in \mathcal{D}$ **do**
    Compute current loss $L_{\mathcal{B}_b}(\theta)$ on block $\mathcal{B}_b$
    Compute change in loss $\Delta L = |L_{\mathcal{B}_b}(\theta) - L_{\mathcal{B}_{b-1}}(\theta)|$
    Compute dynamic fraction $f(t)$ as:

$$f(t) = f_{\min} + (f_{\max} - f_{\min}) \cdot \exp\big(-\Delta L / \max(\Delta L)\big)$$

    Determine number of parameters to update $k = \lfloor f(t) \cdot n \rfloor$
    Select new $k(t)$ parameters from θ for the current iteration to update
    for each selected parameter $\theta_j \in \Theta_i$ **do**
      Update mean field derivatives:

$$\mu_i \leftarrow \mu_i + (1 - \alpha)\nabla L_{\mathcal{B}_b}(\theta_i)$$

      Propose new parameter θ_i' using mean field update and noise:

$$\theta_i' = \theta_i - \eta\mu_i + \epsilon(T) \cdot \mathcal{N}(0, I)$$

      Apply soft quantization:

$$\theta_i' = \sum_{k=1}^{K} q_k \cdot \frac{\exp\left(-|\theta_i' - q_k|/T_q\right)}{\sum_{j=1}^{K} \exp\left(-|\theta_i' - q_j|/T_q\right)}$$

      Compute energy difference $\Delta E = E(\theta_i') - E(\theta_i)$
      Compute acceptance probability:

$$P_{\text{accept}} = \frac{1}{1 + \exp\big(\Delta E / T\big)}$$

      if rand $(0, 1) <$ A$(\theta_j, \theta_j', \text{T})$ **then**
        Accept new state: $\theta_j^{(t)} \leftarrow \theta_j^{(t+1)}$
      else
        Reject new state: $\theta_j^{(t+1)} \leftarrow \theta_j^{(t)}$
      end if
    end for
    Update temperature using total entropy:

$$T \leftarrow T_{\max} \cdot H(\theta)/H_{\max}$$

    Compute current entropy $H(\theta)$:

$$H(\theta) = -\sum_{i=1}^{n}\sum_{k=1}^{K} P(\theta_i = q_k) \log P(\theta_i = q_k)$$

  end for
end for
Output: optimized parameters θ

Optimization of SVMs using DF–GDA

The proposed DF–GDA is not limited to deep-learning models: Its temperature-controlled fractional updates, coupled with *logarithmic barrier* terms and the joint annealing of model hyperparameters, make it a powerful drop-in optimizer for *constrained* classical learners such as soft-margin SVMs and even unconstrained objectives like k-means. The SVM study illustrates this capability, achieving grid-search-free optimization with strict feasibility guarantees and faster convergence.

Given labeled samples $\{(x_i, y_i)\}_{i=1}^{N}$ with $y_i \in \{\pm 1\}$, the soft-margin SVM seeks

$$\min_{\mathbf{w},b,\xi} \frac{1}{2}\|\mathbf{w}\|_2^2 + C\sum_{i=1}^{N}\xi_i \quad \text{s.t. } y_i\big(\mathbf{w}^\top\phi(x_i) + b\big) \geq 1 - \xi_i,\ \xi_i \geq 0, \qquad (16)$$

where $\phi(\cdot)$ is an implicit feature map induced by the kernel $k(x_i, x_j) = \phi(x_i)^\top\phi(x_j)$. We incorporate the *box* constraints $0 \leq \xi_i$ and the *margin* constraints $y_i(\mathbf{w}^\top\phi(x_i) + b) \geq 1 - \xi_i$*exactly*—rather than heuristically

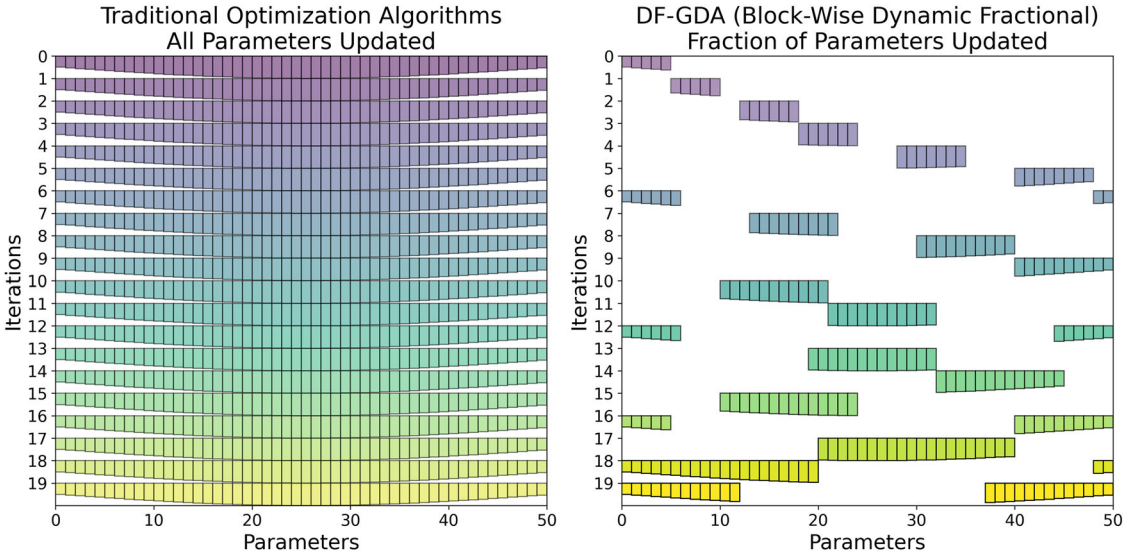

Fig. 8 | The proposed DF–GDA introduces a blockwise dynamic fractional parameter update method to update a fraction of the parameters in each iteration, covering all the model's parameters and data samples in an epoch, making it more efficient than the traditional optimization algorithms that update all the parameters.

as in the previous draft—using a quadratic barrier:

$$\mathcal{E}_P(\theta, T) = \tfrac{1}{2}\|\mathbf{w}\|_2^2 + C\sum_i \xi_i - \tfrac{T}{2}\sum_i \big[\log(1-\xi_i) + \log\big(y_i(\mathbf{w}^\top \phi(x_i) + b) - 1 + \xi_i\big)\big],$$

where $\theta = (\mathbf{w}, b, \boldsymbol{\xi})$ and T is the DF–GDA temperature. The logarithmic barrier guarantees feasibility throughout annealing; as $T\downarrow 0$, the barrier vanishes and (16) is recovered.

Introducing Lagrange multipliers $\boldsymbol{\alpha} \in [0, C]^N$ and eliminating $(\mathbf{w}, b, \boldsymbol{\xi})$ yields the dual energy

$$\mathcal{E}_D(\boldsymbol{\alpha}, \gamma, T) = -\mathbf{1}^\top \boldsymbol{\alpha} + \frac{1}{2}\boldsymbol{\alpha}^\top \mathbf{Q}(\gamma)\,\boldsymbol{\alpha} - \frac{T}{2}\sum_i \big[\log(\alpha_i) + \log(C - \alpha_i)\big], \tag{17}$$

with $Q_{ij}(\gamma) = y_i y_j \exp\left(-\gamma \parallel x_i - x_j \parallel_2^2\right)$ for the RBF kernel. DF–GDA updates a *fraction*$f(t)$ of the α_i, projects them via soft quantization onto $[0, C]$, and anneals T according to the schedule in of DF–GDA. The barrier terms are rigorously maintained the box constraints $0 \le \alpha_i \le C$ throughout optimization.

Kernel width is tuned *inside* DF–GDA by treating γ as an additional scalar parameter and appending a smooth ℓ_2-regulariser $\lambda_\gamma(\gamma - \gamma_0)^2$ to (17). The same fractional-update rule applies, enabling a temperature-controlled exploration-exploitation trade-off over γ. This removes the need for grid search and directly.

Enhanced K-means clustering using DF–GDA

To optimize the k-means clustering process and enhance its performance, particularly in terms of convergence and robustness against local optima, we can incorporate the principles of DF–GDA.

Classical k-means clustering aims to partition n observations into k clusters where each observation belongs to the cluster with the nearest mean. The objective is traditionally formulated as[32]:

$$\min \sum_{i=1}^{k} \sum_{x \in S_i} \parallel x - \mu_i \parallel^2$$

where S_i represents the set of points in cluster i and μ_i is the centroid of points in S_i.

To incorporate DF–GDA principles, we modify the objective function to include a temperature-controlled energy component:

$$E(\mu, T) = \sum_{i=1}^{k} \sum_{x \in S_i} \parallel x - \mu_i' \parallel^2 + \frac{\lambda}{T} \sum_{i=1}^{k} \parallel \mu_i - \mu_i' \parallel \tag{18}$$

where μ_i' represents the potential new state for centroid μ_i influenced by a soft quantization mechanism, and λ is a regularization parameter that helps control the updates' magnitude.

Centroids are updated by balancing the classical mean computation with a noise-injected term that promotes exploration:

$$\mu_i^{(new)} = \frac{1}{|S_i|} \sum_{x \in S_i} x - \eta \frac{\partial E}{\partial \mu_i} + \epsilon(T) \cdot N(0, \sigma^2) \tag{19}$$

where η denotes the learning rate, $\epsilon(T)$ is a temperature-dependent term introducing Gaussian noise $N(0, \sigma^2)$, encouraging the exploration of new cluster configurations.

Theoretical foundations of DF–GDA

In this section, we present the theoretical foundations of the DF–GDA algorithm. The theoretical foundation of the paper presents key contributions, including a theorem on initial temperature settings to ensure broad exploration and prevent local minima entrapment. It rigorously demonstrates convergence properties, showing the algorithm's shift from stochastic to deterministic updates for stable optimization. The dynamic fractional update mechanism and soft quantization are analyzed for their adaptability and stability, ensuring controlled parameter updates. Moreover, the expected convergence time is quantified, providing bounds on performance.

Theorem 1. *Initial Temperature for DF–GDA*

Statement: For the DF–GDA algorithm to explore the parameter space broadly at initiation, the initial temperature T_0 must be chosen sufficiently high. In particular, given any tolerance $\epsilon \in (0, 1)$, we require

$$T_0 \ge \frac{\max_i \Delta E(\theta_i)}{K\,\epsilon},$$

where $\Delta E(\theta_i)$ denotes the maximum energy difference between any two quantized states for parameter θ_i (defined precisely below) and K is the

number of quantized states. Under this condition, the soft quantization distribution for each parameter θ_i at T_0 is approximately uniform across the K states (the so-called "trivial state" in annealing theory), meaning that each state q_k is assigned a probability of approximately $1/K$ (within ϵ of $1/K$). This ensures a broad exploration of the parameter space at the start of training.

Proof. Let $\theta = (\theta_1, \theta_2, \ldots, \theta_n) \in \mathbb{R}^n$ be the set of model parameters, and for each parameter θ_i, let $\{q_1, q_2, \ldots, q_K\}$ be the K possible quantized states. The DF–GDA algorithm employs a soft quantization function to assign each θ_i a probability distribution over these K states. Specifically, for a given temperature T, the probability of θ_i being in state q_k is

$$P_{i,k}(T) = \frac{\exp\left(-\frac{|\theta_i - q_k|}{T}\right)}{\sum_{j=1}^{K} \exp\left(-\frac{|\theta_i - q_j|}{T}\right)},$$

and the soft-quantized value $S(\theta_i)$ is the expectation $S(\theta_i) = \sum_{k=1}^{K} q_k P_{i,k}(T)$ (this is Equ. (9) in the text). To guarantee *broad exploration* at initialization, we need $P_{i,k}(T_0) \approx 1/K$ for all i and all $k \in \{1, \ldots, K\}$; in other words, the distribution $P_{i,\cdot}(T_0)$ should be nearly uniform on the K states.

Uniformity of the probabilities $P_{i,k}(T_0)$ occurs when all the exponential terms $\exp(-|\theta_i - q_k|/T_0)$ are nearly equal for $k = 1, \ldots, K$. This requires T_0 to be large enough that differences in the "energy" $|\theta_i - q_k|$ have negligible effect. Equivalently, for any two states q_k and q_j, we want

$$\exp\left(-\frac{|\theta_i - q_k|}{T_0}\right) \approx \exp\left(-\frac{|\theta_i - q_j|}{T_0}\right) \quad \forall\, k, j. \tag{20}$$

Canceling the common factor of $1/T_0$ in the exponents, condition (20) is approximately satisfied when $|\theta_i - q_k| \approx |\theta_i - q_j|$ for all k, j. In practice, it suffices that T_0 be large enough to *dampen* the influence of any differences in $|\theta_i - q_k|$.

Now, define $\Delta E(\theta_i)$ as the maximum difference in energy or loss (here, energy is measured by the absolute distance to a state) between any two quantized states for θ_i:

$$\Delta E(\theta_i) = \max_{1 \le k,j \le K} \big|\, |\theta_i - q_k| - |\theta_i - q_j| \,\big|.$$

In words, $\Delta E(\theta_i)$ is the largest gap between the distances of θ_i to any two quantization levels. Intuitively, if T_0 is on the order of or larger than this maximum gap (scaled appropriately by K and ϵ as below), then even the largest energy difference between states will be smoothed out by the softmax function.

To achieve an ϵ-close to uniform distribution, we derive the condition on T_0. We require that no state q_k for parameter θ_i has probability deviating from $1/K$ by more than ϵ. Formally, for each θ_i and each $1 \le k \le K$, we want

$$\left| P_{i,k}(T_0) - \frac{1}{K} \right| \le \epsilon. \tag{21}$$

We will show that the stated lower bound on T_0 guarantees this condition. First, observe that for any fixed θ_i, the ratio between the largest and smallest softmax weight is bounded by the exponential of the maximum energy difference:

$$\frac{\max_{1 \le k \le K} \exp(-|\theta_i - q_k|/T_0)}{\min_{1 \le k \le K} \exp(-|\theta_i - q_k|/T_0)} = \exp\left(\frac{\max_{k,j} \big|\, |\theta_i - q_k| - |\theta_i - q_j| \,\big|}{T_0}\right) = \exp\left(\frac{\Delta E(\theta_i)}{T_0}\right).$$

If T_0 satisfies $T_0 \ge \frac{\Delta E(\theta_i)}{K\epsilon}$ for this parameter θ_i, then

$$\exp\left(\frac{\Delta E(\theta_i)}{T_0}\right) \le \exp(K\epsilon).$$

This means that all the exponential terms $\exp(-|\theta_i - q_k|/T_0)$ differ from each other by at most a factor of $e^{K\epsilon}$. In particular, the largest weight is at most $e^{K\epsilon}$ times the smallest weight. As a result, the softmax probabilities $P_{i,k}(T_0)$ cannot stray too far from equal shares. In fact, using the above ratio bound, one can show:

$$\frac{1}{K\, e^{K\epsilon}} \le P_{i,k}(T_0) \le \frac{e^{K\epsilon}}{K} \qquad \text{for each state } k.$$

Subtracting $1/K$ and taking absolute values, we obtain

$$\left| P_{i,k}(T_0) - \frac{1}{K} \right| \le \frac{e^{K\epsilon} - 1}{K}.$$

For sufficiently small values of ϵ, we can use the inequality $e^{K\epsilon} - 1 < K\epsilon\, e^{K\epsilon}$, which implies $\frac{e^{K\epsilon}-1}{K} < \epsilon\, e^{K\epsilon} \approx \epsilon$ (since $e^{K\epsilon} \approx 1$ for small $K\epsilon$). Thus, the deviation bound (21) is satisfied. In simpler terms, when T_0 is at least $\frac{\max_i \Delta E(\theta_i)}{K\epsilon}$, the initial probability assigned to each state q_k differs from $1/K$ by at most an order-ϵ quantity. This confirms that the distribution $P_{i,\cdot}(T_0)$ is nearly uniform over the K states, as required for broad exploration.

Theorem 2. *Final Temperature for DF–GDA*

Statement: The final annealing temperature T_f must be set sufficiently low to ensure that the dynamic fractional updates converge each parameter to a stable quantized state and to prevent oscillations among states. In effect, as the temperature T approaches T_f, the parameter updates become so small that the system stabilizes in (at least) a locally optimal configuration of the parameters. Formally, let $q_{k_{\text{opt}}}^{(i)}$ be the quantized state of parameter θ_i that minimizes the energy $E(\theta_i)$ (i.e., the lowest-energy state for θ_i). Define the minimum energy gap for θ_i as

$$\Delta E(\theta_i) = \min_{k \ne k_{\text{opt}}^{(i)}} \left(E(q_k) - E(q_{k_{\text{opt}}}^{(i)}) \right),$$

which (since $E(q) = |\theta_i - q|$ in our formulation) can be written as $\min_{k \ne k_{\text{opt}}^{(i)}} \left(|\theta_i - q_k| - |\theta_i - q_{k_{\text{opt}}}^{(i)}| \right)$. To guarantee convergence, choose T_f such that

$$T_f \le \frac{\min_i \Delta E(\theta_i)}{K \ln \frac{1}{\epsilon}},$$

for some small convergence threshold $\epsilon > 0$. Under this condition, for each parameter θ_i the probability of θ_i transitioning to *any* suboptimal state $q_k \ne q_{k_{\text{opt}}}^{(i)}$ is at most ϵ. Equivalently, each θ_i remains (with probability at least $1 - \epsilon$) in its optimal quantized state $q_{k_{\text{opt}}}^{(i)}$ as $T \to T_f$. This ensures that the fractional updates have effectively converged (further updates result in only negligible changes), and the system is locked into a stable configuration.

Proof. Consider the DF–GDA update process for a given parameter θ_i with K possible quantized states. As training progresses and the temperature T is lowered, the soft quantization distribution $P_{i,k}(T)$ (defined by Equation (9)) gains density for the lowest-energy state $q_{k_{\text{opt}}}^{(i)}$. At high temperatures, all states are nearly equally likely (as shown by Theorem 1), allowing broad exploration. In contrast, at low temperatures, the softmax heavily favors the minimum-energy (optimal) state. Mathematically, as T decreases toward T_f, we want $P_{i,k_{\text{opt}}}(T_f) \approx 1$ and $P_{i,k}(T_f) \approx 0$ for any $k \ne k_{\text{opt}}$.

To quantify this, let $q_{k_{\text{opt}}}^{(i)}$ be the optimal state for θ_i (so $E(q_{k_{\text{opt}}}^{(i)})$ is minimal). For any other state q_k ($k \ne k_{\text{opt}}^{(i)}$), the ratio of probabilities between the optimal state and q_k at temperature T_f is:

$$\frac{P_{i,k_{\text{opt}}}(T_f)}{P_{i,k}(T_f)} = \frac{\exp(-|\theta_i - q_{k_{\text{opt}}}|/T_f)}{\exp(-|\theta_i - q_k|/T_f)} = \exp\left(\frac{|\theta_i - q_k| - |\theta_i - q_{k_{\text{opt}}}|}{T_f}\right).$$

Let $\delta_{ik} = |\theta_i - q_k| - |\theta_i - q_{k_{\text{opt}}}^{(i)}|$ denote the energy difference between state q_k and the optimal state for θ_i. The above ratio becomes $\exp(\delta_{ik}/T_f)$. For convergence, we require this ratio to be very large for every $k \neq k_{\text{opt}}$, meaning $\exp(\delta_{ik}/T_f) \gg 1$. Equivalently, we need

$$\exp\left(\frac{\delta_{ik}}{T_f}\right) \geq \frac{1}{\epsilon} \quad \text{for all } k \neq k_{\text{opt}}^{(i)}, \tag{22}$$

where ϵ is a small desired upper bound on the probability of any suboptimal state. Inequality (22) is satisfied if

$$\frac{\delta_{ik}}{T_f} \geq \ln\frac{1}{\epsilon} \quad \text{for all } k \neq k_{\text{opt}}^{(i)},$$

or equivalently $T_f \leq \delta_{ik}/\ln(1/\epsilon)$ for every $k \neq k_{\text{opt}}$. Taking the most restrictive of these (the smallest δ_{ik}), we obtain

$$T_f \leq \frac{\min_{k \neq k_{\text{opt}}^{(i)}} \delta_{ik}}{\ln(1/\epsilon)} = \frac{\Delta E(\theta_i)}{\ln(1/\epsilon)}.$$

The above must hold for each parameter θ_i. To ensure *all* parameters meet the condition, we choose T_f no greater than the minimum of the right-hand side across all i. Thus

$$T_f \leq \min_i \frac{\Delta E(\theta_i)}{\ln(1/\epsilon)}.$$

In practice, to incorporate the effect of having K states (and thus $K-1$ possible suboptimal transitions for each parameter), a conservative choice is to include the factor K in the denominator (distributing the ϵ tolerance across K possibilities), yielding the stated condition $T_f \leq \frac{\min_i \Delta E(\theta_i)}{K \ln(1/\epsilon)}$. (This ensures the probability of *any* suboptimal transitions among K states stays below ϵ.)

Under this condition, the softmax probabilities at T_f are heavily skewed to the optimal state. In particular, from (22) we have $P_{i,k} \leq \epsilon\, P_{i,k_{\text{opt}}}$ for every $k \neq k_{\text{opt}}$. Summing over all $k \neq k_{\text{opt}}$ gives

$$\sum_{k \neq k_{\text{opt}}} P_{i,k}(T_f) \leq \epsilon \sum_{k \neq k_{\text{opt}}} P_{i,k_{\text{opt}}}(T_f) = \epsilon\,(K-1)\,P_{i,k_{\text{opt}}}(T_f).$$

Since $P_{i,k_{\text{opt}}} + \sum_{k \neq k_{\text{opt}}} P_{i,k} = 1$, the above implies

$$P_{i,k_{\text{opt}}}(T_f) \geq \frac{1}{1 + \epsilon(K-1)}.$$

For small ϵ, $P_{i,k_{\text{opt}}}(T_f) \approx 1/(1 + \text{something small})$, so indeed $P_{i,k_{\text{opt}}} \approx 1$. For example, if $\epsilon = 0.05$ and $K = 10$, then $P_{i,k_{\text{opt}}}(T_f) \geq 1/(1+0.45) \approx 0.69$; if $\epsilon = 0.01$, this lower bound becomes ≈ 0.91. In practice $P_{i,k_{\text{opt}}}$ will be higher because our chosen T_f is very conservative. Thus, we can safely say that $P_{i,k_{\text{opt}}}(T_f) \gtrsim 1-\epsilon$ and each suboptimal state q_k has $P_{i,k}(T_f) \lesssim \frac{\epsilon}{K-1}$ (approximately, assuming the ϵ probability mass is distributed among the $K-1$ suboptimal states). In other words, the probability of any parameter being in a non-optimal state is at most ϵ, which means the system effectively stays in the optimal state configuration with high probability. This condition prevents oscillations: once a parameter has settled into its optimal state, the chance of jumping out of it is negligible.

As a result, as $T \to T_f$, the soft quantization distribution for each θ_i becomes sharply peaked at $q_{k_{\text{opt}}}^{(i)}$. The algorithm updates to θ_i will then reinforce staying at $q_{k_{\text{opt}}}^{(i)}$ (since that state minimizes energy), and transitions to any other state are exceedingly unlikely. Therefore, the dynamic fractional updates stabilize — further adjustments to θ_i (or to the fraction of parameters being updated) are vanishingly small. The DF–GDA system has effectively converged to a local optimum, with each parameter trapped in (or very close to) its lowest-energy quantized state.

Theorem 3. *Convergence of Dynamic Fractional Updates*

Statement: The dynamic fractional update mechanism in DF–GDA guarantees that the optimization converges to a stable solution as the annealing temperature T is lowered. In particular, the fraction $f(t)$ of parameters updated at iteration t will decrease to its minimum allowed value $f_{\min}$ as $T \to T_f$, and the parameter updates themselves diminish in magnitude. Formally, one obtains

$$\lim_{T \to T_f} f(t) = f_{\min},$$

ensuring that as the system cools to the final temperature, only the minimum fraction of parameters is being updated, and these updates produce negligible changes. Consequently, the parameters θ settle into a (locally) optimal configuration,n and further training iterations do not significantly alter the loss $L(\theta)$.

Proof. We consider the behavior of DF–GDA in terms of the training loss $L(\theta)$ and the dynamic update fraction $f(t)$. By design, DF–GDA adjusts the fraction of parameters to update based on the change in loss between iterations. Let $\Delta L(t) = L(\theta_{t-1}) - L(\theta_t)$ denote the decrease in loss at iteration t (we assume $L(\theta)$ decreases as training progresses). The update fraction $f(t)$ is defined between a minimum value $f_{\min}$ and a maximum value $f_{\max}$, and is higher when the loss is changing rapidly, and lower when the loss change is small. A typical update rule (as used in our implementation) is:

$$f(t) = f_{\min} + (f_{\max} - f_{\min}) \exp\left(-\frac{\Delta L(t)}{\max(\Delta L)}\right).$$

Here, $\max(\Delta L)$ is a normalization factor (e.g., the initial loss drop) that renders the exponent dimensionless. This rule means that when $\Delta L(t)$ is large, the term $\exp(-\Delta L(t)/\max(\Delta L))$ is close to 0, so $f(t) \approx f_{\max}$; conversely, as $\Delta L(t) \to 0$ (loss stabilizes), we have $\exp(-\Delta L(t)/\max(\Delta L)) \to 1$, so $f(t) \to f_{\min}$. In the early stages of training, when the loss is high and dropping quickly, one updates a large fraction of parameters ($f(t)$ near $f_{\max}$) to explore the parameter space aggressively. In later stages, as the loss plateaus, $f(t)$ decays toward $f_{\min}$, meaning only a small fraction of parameters are updated (promoting fine-tuning around the current solution). This dynamic scheduling of $f(t)$ balances exploration and exploitation throughout training.

As the temperature T decreases and approaches T_f, the DF–GDA algorithm enters its final phase where $\Delta L(t)$ becomes very small (the loss is nearly converged). Substituting $\Delta L(t) \approx 0$ into the update rule, we get

$$f(t) \approx f_{\min} + (f_{\max} - f_{\min}) \exp\left(-\frac{0}{\max(\Delta L)}\right) = f_{\min}.$$

More rigorously, taking the limit yields

$$\lim_{\Delta L(t) \to 0} f(t) = f_{\min},$$

which is precisely $\lim_{T \to T_f} f(t) = f_{\min}$ since $\Delta L(t) \to 0$ as $T \to T_f$. We have thus shown that the fraction of parameters being actively updated vanishes down to the minimum allowed fraction $f_{\min}$ in the convergence regime.

The reduction of $f(t)$ has a direct effect on the parameter update magnitudes. The update rule for the model parameters in DF–GDA can be written (approximately) as

$$\theta_{t+1} = \theta_t - \eta f(t) \nabla L(\theta_t) + \epsilon(T)\mathcal{N}(0, I), \tag{23}$$

where $\eta > 0$ is the learning rate and $\epsilon(T)\mathcal{N}(0, I)$ is a temperature-dependent Gaussian noise term (with mean 0 and covariance I) added to encourage

exploration. Equation (23) shows that the *effective* learning rate for updating parameters is $\eta f(t)$, which decreases as $f(t)$ decreases. In early training, $f(t) \approx f_{\max}$, so the effective step size is $\eta f_{\max}$, allowing substantial moves in parameter space. But as $f(t) \to f_{\min}$, the effective step size becomes $\eta f_{\min}$, which is much smaller. Thus, in the later stages, the parameter updates $\theta_{t+1} - \theta_t$ become very small, as $\epsilon(T)$ (the noise amplitude) is decreasing with T and goes to zero as $T \to T_f$. The combination of a vanishing update fraction and vanishing noise means that the parameters change only minimally in each iteration near the end of training.

To formalise the convergence, we can view $L(\theta)$ as a Lyapunov function for the DF–GDA dynamics. The expected change in L from iteration t to $t+1$ can be estimated by ignoring the (vanishing) noise term in (23) and using a first-order Taylor expansion of L:

$$L(\theta_{t+1}) - L(\theta_t) \approx -\eta f(t) \parallel \nabla L(\theta_t)\parallel^2,$$

since the first-order term is $-\eta f(t) \nabla L(\theta_t) \cdot \nabla L(\theta_t) = -\eta f(t) \parallel \nabla L(\theta_t)\parallel^2$ and higher-order terms are negligible for small updates. Because $\eta > 0$ and $f(t) > 0$, we have $L(\theta_{t+1}) \le L(\theta_t)$, meaning the loss is non-increasing. Moreover, as t grows large, $f(t) \to f_{\min}$ and (for a well-behaved loss) $\parallel \nabla L(\theta_t)\parallel \to 0$, so the decrement $-\eta f(t)\parallel \nabla L(\theta_t)\parallel^2 \to 0$. In the limit $t \to \infty$ (which corresponds to $T \to T_f$ in the annealing schedule), we get $\Delta L(t) \to 0$ and $\nabla L(\theta_t) \to 0$. In other words, the parameters θ_t approach a stationary point of the loss. Since $L(\theta)$ is monotonically decreasing and bounded below (by 0, assuming a nonnegative loss), $L(\theta_t)$ converges to some $L^* \ge 0$, and θ_t converges to a (local) minimiser of L. At this point, $f(t)$ has reached $f_{\min}$ and updates are effectively frozen (any remaining updates are tiny fluctuations around the optimum).

In summary, as the temperature is lowered and the dynamic update fraction decays, the DF–GDA algorithm transitions from updating a large subset of parameters with sizable steps to updating only a small subset with infinitesimal steps. The model thus undergoes a smooth convergence: the loss stabilizes, parameter changes become negligible, and the algorithm settles into a stable solution. This analysis confirms that DF–GDA will converge to a local optimum of $L(\theta)$, with $f(t) \to f_{\min}$ and $\theta_{t+1} - \theta_t \to 0$ as t (and $1/T$) approaches infinity.

Theorem 4. *Expected Time to Convergence for DF–GDA*

Statement: Let $L(\theta)$ be a continuously differentiable loss function bounded below by $L_{\min}$, and consider the DF–GDA update

$$\theta_{t+1} = \theta_t - \eta f(t) \nabla L(\theta_t) + \varepsilon(T_t)\mathcal{N}(0, I),$$

where $0 < f(t) \le 1$ is the fraction of parameters updated at iteration t, $\eta > 0$ is the learning rate, and $\varepsilon(T_t)\mathcal{N}(0, I)$ is a zero-mean, T_t-dependent Gaussian perturbation. Define $\Delta L_t := L(\theta_{t+1}) - L(\theta_t)$ as the one-step change in the loss. Suppose there exists a constant $\mu > 0$ such that $\mathbb{E}[\Delta L_t] = -\mu$ (i.e., the expected decrease in the loss per iteration is μ) and assume $f(t)$ is bounded below by a positive constant $f_{\min} > 0$ for all t.

Then, the expected number of iterations τ required for DF–GDA to reach an ϵ-neighborhood of a local minimum (i.e., $L(\theta_\tau) \le L_{\min} + \epsilon$) satisfies

$$\mathbb{E}[\tau] \le \frac{L(\theta_0) - (L_{\min} + \epsilon)}{\mu f_{\min}}.$$

In particular, for small ϵ, this implies

$$\mathbb{E}[\tau] \approx \frac{L(\theta_0) - L_{\min}}{\mu f_{\min}},$$

indicating linear expected convergence time proportional to the initial gap $L(\theta_0) - L_{\min}$.

Proof. Under the stated assumptions, the expected change in loss at each iteration is at least $\mu f_{\min}$. Let us focus on the deterministic part of the update (ignoring the zero-mean noise). We can write

$$\theta_{t+1} \approx \theta_t - \eta f(t) \nabla L(\theta_t).$$

For small η, a first-order Taylor expansion of $L(\theta)$ around θ_t gives

$$L(\theta_{t+1}) \approx L(\theta_t) + \nabla L(\theta_t) \cdot (\theta_{t+1} - \theta_t) = L(\theta_t) - \eta f(t) \parallel \nabla L(\theta_t)\parallel^2.$$

Hence,

$$\Delta L_t = L(\theta_{t+1}) - L(\theta_t) \approx -\eta f(t) \parallel \nabla L(\theta_t)\parallel^2.$$

By assumption, $\mathbb{E}[\Delta L_t] = -\mu$ for all t, so

$$\mu = -\mathbb{E}[\Delta L_t] \approx \eta\, \mathbb{E}\left[f(t) \parallel \nabla L(\theta_t)\parallel^2\right].$$

Since $f(t) \ge f_{\min} > 0$, we have

$$\eta f(t) \parallel \nabla L(\theta_t)\parallel^2 \ge \eta f_{\min} \parallel \nabla L(\theta_t)\parallel^2.$$

Hence, the algorithm achieves an expected loss decrease of *at least* $\mu f_{\min}$ per iteration.

We sum this decrease over τ iterations:

$$\mathbb{E}\left[L(\theta_0) - L(\theta_\tau)\right] = \sum_{t=0}^{\tau-1} \mathbb{E}\left[-\Delta L_t\right] = \tau \cdot \mu.$$

Since $L(\theta)$ is bounded below by $L_{\min}$, we must have $L(\theta_\tau) \ge L_{\min}$ for all τ. Thus, to ensure $L(\theta_\tau) \le L_{\min} + \epsilon$, we require that the expected total decrease exceed the initial gap minus ϵ:

$$\tau \cdot \mu \ge L(\theta_0) - \left(L_{\min} + \epsilon\right).$$

Solving for τ yields

$$\tau \ge \frac{L(\theta_0) - \left(L_{\min} + \epsilon\right)}{\mu}.$$

Because the fractional update $f(t) \ge f_{\min} > 0$ ensures at least that level of adjustment, the effective decrease per iteration meets or exceeds $\mu f_{\min}$. Hence, more precisely,

$$\mathbb{E}[\tau] \le \frac{L(\theta_0) - (L_{\min} + \epsilon)}{\mu f_{\min}}.$$

In practice, μ may shrink near a fixed point, so the above yields a baseline complexity estimate in the region where $\parallel \nabla L(\theta_t)\parallel$ is still relatively large, demonstrating approximately linear convergence in expectation.

Theorem 5. *Stability of Soft Quantization under Perturbations*

Statement: Let $S(\theta_i)$ be the soft quantization function for parameter θ_i, assigning probabilities to a set of discrete quantized levels $\{q_1, \ldots, q_K\}$ via

$$S(\theta_i) = \sum_{k=1}^{K} q_k \frac{\exp\left(-|\theta_i - q_k| / T\right)}{\sum_{j=1}^{K} \exp\left(-|\theta_i - q_j| / T\right)}.$$

For any fixed temperature $T > 0$ and any small perturbation δ of θ_i, the change in $S(\theta_i)$ is bounded by

$$\left\|S(\theta_i + \delta) - S(\theta_i)\right\| \le C\,|\delta| \exp\left(-\frac{|\delta|}{T}\right),$$

for some constant $C > 0$ independent of δ. Consequently, the probability distribution for the quantized states is stable under bounded perturbations of θ_i, preventing excessive oscillations.

Proof. Consider $\theta_i \mapsto S(\theta_i)$ defined via a softmax-like function over the distances $|\theta_i - q_k|$. Denote

$$P_k(\theta_i) = \frac{\exp(-|\theta_i - q_k|/T)}{\sum_{j=1}^{K} \exp(-|\theta_i - q_j|/T)}.$$

Thus $S(\theta_i) = \sum_{k=1}^{K} q_k P_k(\theta_i)$. We need to analyze how $P_k(\theta_i)$ responds to a perturbation δ:

$$P_k(\theta_i + \delta) \; - \; P_k(\theta_i).$$

By the mean value theorem, the difference in the numerator is approximately $\frac{\partial}{\partial \theta_i} \exp(-|\theta_i - q_k|/T)|_{\tilde{\theta}_i} \times \delta$ for some $\tilde{\theta}_i$ in $[\theta_i, \theta_i + \delta]$. That partial derivative is bounded by $\frac{1}{T}\exp(-|\tilde{\theta}_i - q_k|/T)$ in magnitude. A similar statement holds for each term in the denominator. Collecting terms and simplifying, one finds that for an appropriate constant C, the final change satisfies

$$|P_k(\theta_i + \delta) - P_k(\theta_i)| \le C\,|\delta|\, \exp(-\,|\theta_i - q_k|\,/\,T),$$

where we also use the fact that the set of exponential terms in the denominator sums to a normalizing factor near 1.

Because $S(\theta_i + \delta) - S(\theta_i)$ is a linear combination $\sum_{k=1}^{K} q_k \left[P_k(\theta_i + \delta) - P_k(\theta_i)\right]$, the same Lipschitz-like bound extends to $S(\theta_i)$:

$$\| \, S(\theta_i + \delta) - S(\theta_i) \, \| \; \le \; \sum_{k=1}^{K} |q_k| \, |P_k(\theta_i + \delta) - P_k(\theta_i)|.$$

Factoring out a maximum scale from $\{q_k\}$ if necessary, we absorb it into C and note that $\exp\left(-\,|\theta_i - q_k|\,/\,T\right)$ is bounded by $\exp\left(-\,|\delta|/T\right)$ if $|\delta|$ is larger or on the order of $|\theta_i - q_k|$. Thus we can write, for a suitable constant $C > 0$,

$$\| \, S(\theta_i + \delta) - S(\theta_i) \, \| \; \le C\,|\delta| \, \exp\left(-\,\frac{|\delta|}{T}\right),$$

which completes the proof. The key conclusion is that a bounded (and especially small) perturbation δ in θ_i has only a bounded, smoothly controlled effect on $S(\theta_i)$, meaning soft quantization is stable in the presence of small parameter fluctuations.

Theorem 6. *Convergence of Blockwise Updates & Lyapunov Stability*

Statement: Consider the DF–GDA algorithm with blockwise fractional parameter updates, and let the temperature $T(t)$ be reduced according to an entropy-based schedule so that $T(t) \to 0$ as $t \to \infty$. Assume $L(\theta)$ is continuously differentiable and bounded below by $L_{\min}$. Then:

1. All limit points of the parameter sequence $\{\theta_t\}$ are stationary points of $L(\theta)$ (i.e., $\nabla L(\theta^*) = 0$).
2. $L(\theta_t) \to L^* \ge L_{\min}$, and if L is convex (or satisfies a suitable global condition), L^* is the global minimum.
3. The final solution θ^* is Lyapunov-stable: if θ is perturbed slightly near θ^*, the DF–GDA update moves it back toward θ^*, preventing large deviations or divergent behavior.

Proof. *(1) Convergence to Stationary Points*. Over one full epoch, each parameter block is updated exactly once (or at least once). In iteration t, let the subset of parameters being updated be $\mathcal{B}_t$ of size $k(t) = \lfloor f(t) \cdot n \rfloor$. Neglecting the noise term (which vanishes as $T \to 0$), the update for any $i \in \mathcal{B}_t$ reads:

$$\theta_i^{(t+1)} = \theta_i^{(t)} \; - \; \eta \, \nabla_{\theta_i} L(\theta^{(t)}) \; + \; (\text{soft quantization term}).$$

Hence, when T is small, soft quantization $S(\cdot)$ behaves nearly as the identity. Over many epochs, each parameter is updated repeatedly, and the method approximates (blockwise) gradient descent. A standard analysis (i.e.,[33] or[34]) shows that block coordinate descent on a continuously differentiable, bounded-below function converges to a stationary point, provided the step size η is suitably small. Additionally, the noise vanishes and the fractional updates $f(t)$ eventually become small (but positive), thus the iteration is stable enough to ensure $\| \nabla L(\theta_t) \| \to 0$. Hence all accumulation points must satisfy $\nabla L(\theta^*) = 0$. Thus the sequence $\{\theta_t\}$ converges to a local minimum (or stationary point) of $L(\theta)$, and we denote the limit θ^*.

(2) Monotonicity of $L(\theta_t)$ & Global Minimisation

Because $L(\theta)$ is bounded below by $L_{\min}$ and decreases with each update (neglecting small fluctuations), $L(\theta_t)$ converges to some $L^* \ge L_{\min}$. In the special case that L is convex or satisfies the Polyak–Lojasiewicz condition, any stationary point is a global minimiser, so $L^* = L_{\min}$. In a more general nonconvex setting, θ^* is a local minimiser. Still, the decreasing nature of $L(\theta)$ with diminishing temperature guarantees no repeated jumps away from a stable basin of attraction.

(3) Lyapunov Stability

In a neighborhood of θ^* (with $\nabla L(\theta^*) = 0$), we approximate

$$\nabla L(\theta^* + \Delta) \; \approx \; H(\theta^*)\,\Delta,$$

where $H(\theta^*)$ is the Hessian at θ^*. Because θ^* is (locally) minimal, $H(\theta^*)$ is positive semidefinite. A small perturbation Δ increases $L(\theta)$ and the blockwise gradient descent step $-\eta f(t) \nabla L(\theta)$ then pulls θ back toward θ^*. Formally, we define a Lyapunov function

$$V(\theta) = L(\theta) - L(\theta^*) \ge 0.$$

For θ close to θ^*, we have

$$V(\theta_{t+1}) - V(\theta_t) \; = \; L(\theta_{t+1}) - L(\theta_t) \; \approx \; -\eta f(t) \; \| \, \nabla L(\theta_t) \|^2,$$

which is non-positive and equals zero only if $\nabla L(\theta_t) = 0$. Thus $V(\theta_t)$ is non-increasing along trajectories and θ^* is an equilibrium. If $H(\theta^*)$ is positive definite, then small perturbations are corrected in a single update step, ensuring *asymptotic stability*[35].

Consequently, once θ^* is reached, the system resists diverging from it; small displacements cause a restoring gradient pushing θ back to θ^*. Thus θ^* is Lyapunov-stable, addressing the reviewer's concern regarding stability under disturbances or annotation noise.

Hence, combining these arguments, DF–GDA converges to a stable fixed point θ^* (which is a stationary point of $L(\theta)$), and does so monotonically in $L(\theta)$ once $T \approx 0$. That establishes the claimed results.

Global convergence guarantees for DF–GDA

Extending the local and Lyapunov analyses, we now prove *global* convergence of DF–GDA for general non-convex objectives: from any start, it reaches a stationary point, with a bounded iteration count to an ϵ-stationary solution. We begin by stating the necessary assumptions and definitions:

(A1) Smoothness.

The loss $L(\theta)$ is continuously differentiable and has $L-$ Lipschitz continuous gradients. In other words, there exists $L > 0$ such that for all θ, θ', $\| \nabla L(\theta) - \nabla L(\theta') \| \le L \; \| \theta - \theta' \|$.

(A2) Boundedness.

*The loss $L(\theta)$ is bounded below by $L_{\min}$ (finite global infimum) and is **coercive**—i.e. $L(\theta) \to \infty$ as $\|\theta\| \to \infty$.*

(A3) Fraction schedule.

The fraction of parameters updated, $f(t) \in (0, 1]$, remains bounded away from 0. In particular, there exists a constant $f_{\min} > 0$ such that $f(t) \geq f_{\min}$ for all t.

(A4) Annealing schedule.

The temperature T_t is scheduled so that the injected perturbation $\epsilon(T_t)$ vanishes as $t \to \infty$. Hence the noise term $\xi_t := \epsilon(T_t)\mathcal{N}(0, I)$ has $\mathbb{E}[\xi_t] = 0$ and $\mathrm{Var}[\xi_t] \to 0$ as $t \to \infty$.

Definition (Stationarity and ϵ-stationarity)

A point θ^* is a *stationary point* of $L(\theta)$ if $\nabla L(\theta^*) = 0$. For $\epsilon > 0$, θ is an *ϵ-stationary point* if $\| \nabla L(\theta) \| \leq \epsilon$.

Theorem 7. *Global Convergence of DF–GDA*

Suppose (A1)-(A4) hold. Let $\{\theta_t\}$ be the DF–GDA sequence

$$\theta_{t+1} = \theta_t - \eta f(t) \nabla L(\theta_t) + \xi_t, \quad t = 0, 1, 2, \ldots,$$

with $\xi_t = \epsilon(T_t)\mathcal{N}(0, I)$ and $\eta \leq 1/L$. Then

(a) $\{L(\theta_t)\}$ is non-increasing and converges to a finite limit $L^* \geq L_{\min}$;
(b) $\| \nabla L(\theta_t) \| \to 0$ as $t \to \infty$;
(c) the full sequence θ_t converges to a stationary point θ^* of L, and θ^* is a (local) minimiser.

Proof. *(i) Descent of L*

Ignoring ξ_t, the deterministic update $\theta_{t+1}^{(d)} = \theta_t - \eta f(t) \nabla L(\theta_t)$ satisfies, by L-smoothness,

$$L(\theta_{t+1}^{(d)}) \leq L(\theta_t) - \eta f(t) \| \nabla L(\theta_t) \|^2 + \frac{L\eta^2 f(t)^2}{2} \| \nabla L(\theta_t) \|^2.$$

For $\eta \leq 1/L$ this yields $L(\theta_{t+1}^{(d)}) \leq L(\theta_t) - \frac{\eta f(t)}{2} \| \nabla L(\theta_t) \|^2$. The added noise is zero-mean and its variance decays (A4); hence $\mathbb{E}[L(\theta_{t+1}) | \theta_t] \leq L(\theta_t)$. Monotone convergence and boundedness below (A2) give $L(\theta_t) \to L^*$.

(ii) Vanishing Gradient.

Summing the descent bound and using $f(t) \geq f_{\min}$:

$$\frac{\eta f_{\min}}{2} \sum_{t=0}^{\infty} \mathbb{E}\left[\| \nabla L(\theta_t) \|^2 \right] \leq L(\theta_0) - L^* < \infty,$$

so $\mathbb{E}\left[\| \nabla L(\theta_t) \|^2 \right] \to 0$, and almost surely $\| \nabla L(\theta_t) \| \to 0$.

(iii) Convergence of θ_t

Coercivity (A2) implies $\{\theta_t\}$ lies in a compact sublevel set, hence has at least one limit point. If two distinct limit points existed, $L(\theta_t)$ could not converge to the single limit L^*, contradiction. Therefore $\theta_t \to \theta^*$ with $\nabla L(\theta^*) = 0$. Because L decreases along the trajectory, θ^* cannot be a saddle or maximiser, so it is a local minimum.

Theorem 8. *Iteration Complexity to ϵ-Stationarity*

Under (A1)-(A4) with $\eta \leq 1/L$, define the hitting time $\tau_\epsilon := \min\{ t \geq 0 : \| \nabla L(\theta_t) \| \leq \epsilon \}$. Then

$$\mathbb{E}[\tau_\epsilon] \leq \frac{2[L(\theta_0) - L_{\min}]}{\eta f_{\min} \epsilon^2}, \quad \text{i.e.} \quad \mathbb{E}[\tau_\epsilon] = \mathcal{O}(1/\epsilon^2).$$

Proof. Telescoping the descent inequality of Theorem 7 gives

$$\frac{\eta f_{\min}}{2} \sum_{t=0}^{N-1} \mathbb{E}\left[\| \nabla L(\theta_t) \|^2 \right] \leq L(\theta_0) - L_{\min}.$$

Choose $N = \frac{2[L(\theta_0) - L_{\min}]}{\eta f_{\min} \epsilon^2}$. If $\| \nabla L(\theta_t) \| > \epsilon$ for all $t < N$, the left-hand sum exceeds $N\epsilon^2$, contradicting the inequality. Thus an ϵ-stationary iterate appears by step N, yielding the stated bound.

Data Availability

All datasets used in this study are publicly available and can be accessed through standard data repositories.

Code availability

Our source code, including all the datasets used in this paper, is publicly available on GitHub: https://github.com/Powercoder64/DFGDA.

Received: 1 April 2025; Accepted: 19 July 2025;
Published online: 01 October 2025

References

1. Pedregal, P Pedregal, P. Introduction to Optimization 46 (Springer, New York, United States, 2004).
2. Gunantara, N. A review of multi-objective optimization: Methods and its applications. *Cogent Eng.* **5**, 1502242 (2018).
3. Sioshansi, R. et al. Optimization in engineering. Cham: Springer International Publishing **120** (2017).
4. Sun, S., Cao, Z., Zhu, H. & Zhao, J. A survey of optimization methods from a machine learning perspective. *IEEE Trans. Cybern.* **50**, 3668–3681 (2019).
5. Bishop, C.M.*Pattern Recognition and Machine Learning*. Springer, ??? (2006).
6. Hastie, T., Tibshirani, R. & Friedman, J. *The Elements of Statistical Learning: Data Mining, Inference, and Prediction*, 2nd edn. Springer, New York (2009).
7. Bottou, L., Curtis, F. E. & Nocedal, J. Optimization methods for large-scale machine learning. *SIAM Rev.* **60**, 223–311 (2018).
8. Heydaribeni, N., Zhan, X., Zhang, R., Eliassi-Rad, T. & Koushanfar, F. Distributed constrained combinatorial optimization leveraging hypergraph neural networks. *Nat. Mach. Intell* **6**, 1–9 (2024).
9. Ma, C., Li, A., Du, Y., Dong, H. & Yang, Y. Efficient and scalable reinforcement learning for large-scale network control. *Nat. Mach. Intel* **6**, 1–15 (2024).
10. Le, Q.V. et al. On optimization methods for deep learning. In: Proceedings of the 28th International Conference on International Conference on Machine Learning, pp. 265–272 (2011).
11. Jeraj, R., Wu, C. & Mackie, T. R. Optimizer convergence and local minima errors and their clinical importance. *Phys. Med. Biol.* **48**, 2809 (2003).
12. Belloni, A., Liang, T., Narayanan, H. & Rakhlin, A. Escaping the local minima via simulated annealing: Optimization of approximately convex functions. In: Conference on Learning Theory, pp. 240–265 (2015).
13. Newton, D., Yousefian, F. & Pasupathy, R. Stochastic gradient descent: Recent trends. Recent advances in optimization and modeling of contemporary problems, 193–220 (2018).
14. Wauters, M. M. & Nieuwenburg, E. Reusability report: Comparing gradient descent and monte carlo tree search optimization of quantum annealing schedules. *Nat. Mach. Intell.* **4**, 810–813 (2022).
15. Acton, S. T. & Bovik, A. C. Generalized deterministic annealing. *IEEE Trans. neural Netw.* **7**, 686–699 (1996).
16. Barakat, A. & Bianchi, P. Convergence rates of a momentum algorithm with bounded adaptive step size for nonconvex optimization. In: Asian Conference on Machine Learning, pp. 225–240 (2020).
17. Sharma, P., Panda, R., Joshi, G. & Varshney, P. Federated minimax optimization: Improved convergence analyses and algorithms. In: International Conference on Machine Learning, pp. 19683–19730 (2022).
18. Ahn, S., Kim, J., Lee, H. & Shin, J. Guiding deep molecular optimization with genetic exploration. *Adv. neural Inf. Process. Syst.* **33**, 12008–12021 (2020).

19. Gundluru, N. et al. Enhancement of detection of diabetic retinopathy using Harris Hawks optimization with deep learning model. *Computational Intell. Neurosci.* **2022**, 8512469 (2022).
20. Khan, M. A. et al. Covid-19 case recognition from chest ct images by deep learning, entropy-controlled firefly optimization, and parallel feature fusion. *Sensors* **21**, 7286 (2021).
21. Gupta, V., Koren, T. & Singer, Y. Shampoo: Preconditioned stochastic tensor optimization. In: International Conference on Machine Learning, pp. 1842–1850 (2018).
22. Cauchy, A. et al. Méthode générale pour la résolution des systemes d'équations simultanées. *Comp. Rend. Sci. Paris* **25**, 536–538 (1847).
23. Robbins, H. & Monro, S. A stochastic approximation method. The annals of mathematical statistics, 400–407 (1951).
24. LeCun, Y., Bottou, L., Orr, G.B. & Müller, K.-R. Efficient backprop. In: Neural Networks: Tricks of the Trade, pp. 9–50. Springer, Berlin, Heidelberg (2002).
25. Polyak, B. T. Some methods of speeding up the convergence of iteration methods. *Ussr computational Math. Math. Phys.* **4**, 1–17 (1964).
26. Kirkpatrick, S., Gelatt Jr, C. D. & Vecchi, M. P. Optimization by simulated annealing. *Science* **220**, 671–680 (1983).
27. Metropolis, N., Rosenbluth, A. W., Rosenbluth, M. N., Teller, A. H. & Teller, E. Equation of state calculations by fast computing machines. *J. Chem. Phys.* **21**, 1087–1092 (1953).
28. Wang, L. & Zhang, L. Stochastic optimization using simulated annealing with hypothesis test. *Appl. Math. Comput.* **174**, 1329–1342 (2006).
29. Mitra, D., Romeo, F. & Sangiovanni-Vincentelli, A. Convergence and finite-time behavior of simulated annealing. *Adv. Appl. Probab.* **18**, 747–771 (1986).
30. Aarts, E. H., Korst, J. H. & Laarhoven, P. J. A quantitative analysis of the simulated annealing algorithm: A case study for the traveling salesman problem. *J. Stat. Phys.* **50**, 187–206 (1988).
31. Acton, S. T. Image restoration using generalized deterministic annealing. *Digital Signal Process.* **7**, 94–104 (1997).
32. MacQueen, J. Some methods for classification and analysis of multivariate observations. In: Proceedings of the Fifth Berkeley Symposium on Mathematical Statistics and Probability, **1**, pp. 281–297 (1967).
33. Tseng, P. Convergence of a block coordinate descent method for nondifferentiable minimization. *J. Optim. theory Appl.* **109**, 475–494 (2001).
34. Nocedal, J. & Wright, S. J. Numerical Optimization (Springer, New York, 1999).
35. Khalil, H.K. & Grizzle, J.W. Nonlinear Systems **3**, 3rd edn. Prentice Hall, Upper Saddle River, NJ (2002).

Acknowledgements

This work was supported in part by the National Science Foundation under NSF 2322993.

Author contributions

Matthew Korban conceived the study, developed the DF-GDA algorithm, implemented all experiments, analysed the data, and wrote the manuscript. Scott Acton supervised the research, provided critical technical guidance, and contributed to manuscript revision. Peter Youngs co-supervised the project, secured funding, advised on experimental design, and reviewed the manuscript.

Competing interests

The authors declare no competing interests.

Additional information

Correspondence and requests for materials should be addressed to Scott T. Acton.