An Information-Theoretic Investigation of Nested Learning

David Goh
University of Toronto

December 13, 2025

1 Introduction

Deep learning training can be interpreted as a hierarchy of nested optimization problems, each operating on distinct timescales. At the lowest level, parameters W are updated frequently based on the immediate input x_t, while higher-level components aggregate these updates more slowly, effectively compressing the history of past inputs into concise summaries. Each nested layer thus acts as an associative memory, encoding and propagating information that shapes the dynamics of faster, lower-level updates. Key notation: W (weights), x_t (input at time t), $g_t = \nabla L(W_t; x_t)$ (gradient), m_t (momentum), η (learning rate).

2 Summary: Nested Learning Paper

2.1 From Gradient Descent to Momentum

Standard gradient descent performs a single-level optimization over weights:

$$W_{t+1} = W_t - \eta g_t \tag{1}$$

Here, updates rely solely on the instantaneous gradient, with no explicit mechanism for capturing or compressing past information.

Adding momentum introduces a second, slower optimization layer:

$$W_{t+1} = W_t + m_{t+1} \tag{2}$$
$$m_{t+1} = m_t - \eta g_t \tag{3}$$

Momentum can itself be interpreted as solving a proximal objective:

$$m_{t+1} = \text{argmin}_m \langle m, g_t \rangle + \frac{1}{2\eta}\|m - m_t\|^2 \tag{4}$$

Insight: The momentum term m functions as an associative memory, compressing the history of past gradients into a smoothed signal that guides the weight updates W_{t+1}. This nested structure delineates two distinct gradient flows: rapid updates at the weight level and slower, memory-driven updates at the momentum level. Recognizing this separation clarifies how higher-order gradient information can be systematically captured and leveraged within the optimizer.

2.2 Deeper Nested Optimizers

Extending beyond momentum, higher-order optimizers such as Adam or RMSProp introduce additional internal states (e.g., adaptive second-moment estimates) that operate on slower timescales than m_t. Denoting these higher-level states as $\mathbf{v}_t$, we can write:

$$W_{t+1} = W_t + m_{t+1} \tag{5}$$
$$m_{t+1} = m_t - \eta \frac{g_t}{\sqrt{\mathbf{v}_t + \epsilon}} \tag{6}$$
$$\mathbf{v}_{t+1} = \beta \mathbf{v}_t + (1-\beta) g_t^2 \tag{7}$$

Each new nested state $\mathbf{v}_t$ acts as an associative memory of past gradient magnitudes, biasing the effective step size and implicitly shaping the evolution of representations. This hierarchy can be viewed as a multi-timescale memory network in which each layer compresses different aspects of the gradient trajectory.

Extensions enable deeper nesting:

Preconditioned momentum: $m_{t+1} = \alpha m_t - \eta P_t g_t$ maps gradients to structured values.

Delta-rule objective: $\min_m ||m g_t - P_t||^2$ improves memory capacity.

Deep momentum (DMGD): Replacing linear m with an MLP enables nonlinear gradient compression across timescales.

2.3 Self-Modifying Networks

Nested optimization extends to architecture. Linear attention's update $M_{t+1} = M_t + v_t k_t^T$ is equivalent to one gradient step on $\min_M \langle M k_t, v_t \rangle + ||M - M_t||^2$. Self-modifying layers generalize this:

$$\min_W ||W x_t - g_t||^2 \quad (X) \tag{8}$$

$$W_{t+1} = W_t (I - x_t x_t^T) - \eta g_t x_t^T \quad (Y) \tag{9}$$

The $(I - x_t x_t^T)$ term enables input-dependent updates, preserving relational structure and compressing representation information.

2.4 Continuum Memory

Traditional feedforward layers operate at a single timescale. The Continuum Memory System (CMS) employs multiple MLPs $\theta^{(\ell)}$ updating at frequencies f_ℓ, compressing knowledge from different timescales. Classifiers benefit from multi-rate layer updates.

3 Connection to Information Bottleneck Theory

Tishby's Information Bottleneck (IB) framework characterizes the tradeoff between representation compression $I(T;X)$ and predictive relevance $I(T;Y)$. We hypothesize that deeper nested optimizers induce stronger gradient compression, effectively accelerating the movement of internal representations T along the information plane.

Specifically, if T^l denotes the representation at layer l, the nesting depth modulates how T^l evolves:

$$T^l_{t+1} = T^l_t + f_{\text{nest}}(T^l_t, g^l_t, m^l_t, \mathbf{v}^l_t) \tag{10}$$

where f_{nest} encodes the hierarchical memory updates of the nested optimizer. Empirically, we expect that increasing nesting depth:

1. Compresses $I(T^l;X)$ more rapidly by filtering out gradient noise.
2. Preserves or enhances $I(T^l;Y)$, since predictive features are reinforced through aggregated memory.
3. Alters the trajectory on the information plane, potentially generating steeper compression phases.

This perspective provides a concrete mechanism linking optimizer architecture to representation learning dynamics, bridging nested optimization and the IB principle in a testable framework.

4 Novel Empirical Investigation: IB Dynamics in Nested Optimizers

To validate the "associative memory" and "multi-timescale" claims of the Nested Learning framework, we designed a novel experiment comparing the Information Bottleneck (IB) trajectories of standard optimizers against Deep Momentum Gradient Descent (DMGD).

4.1 Experimental Methodology

We implemented a highly bottlenecked MLP architecture designed to force aggressive representation compression: $784 \to 12 \to 10 \to 8 \to 6 \to 4$. We utilized *tanh* activations at every hidden layer to ensure bounded activations for stable Mutual Information (MI) estimation. The network was trained on MNIST for 3999 epochs using four optimization regimes: SGD, GDM, AdamW, and DMGD. We tracked the evolution of the information plane ($I(T;X)$ vs. $I(T;Y)$) across all five hidden layers using a binning-based MI estimator.

4.2 Results: The Nested Compression Phase

Our results reveal a qualitative shift in learning dynamics unique to the deeply nested optimizer, summarized in Figure 1.

1. The Failure of Traditional Compression: In SGD, GDM, and AdamW, we did not observe a significant compression phase. As seen in the AdamW trajectories, $I(T;X)$ (complexity) rose asymptotically across all layers, plateauing above 10 bits within the first 300 epochs and remaining steady until $t = 3999$. While $I(T;Y)$ (sufficiency) maximized quickly, these networks failed to "forget" irrelevant input noise in the terminal phase.

2. DMGD's Unique Compression Phase: DMGD exhibited a distinct two-phase trajectory in deep layers (L_4, L_5).

- **Phase 1 (Expansion):** $I(T;X)$ increases until approximately Epoch 150 as the model captures features.
- **Phase 2 (Compression):** Post-Epoch 150, $I(T;X)$ in the penultimate layer enters a sustained decrease, settling near 7 bits. Most strikingly, the final hidden layer (L_5) approaches an $I(T;X)$ of 0 bits asymptotically throughout the 3999 epochs.

3. Generalization and Stability: While DMGD reached a similar terminal accuracy to AdamW, it maintained a significantly tighter gap between training and validation accuracy. We observed high-frequency fluctuations in $I(T;Y)$ for the penultimate layer of DMGD between Epochs 500 and 3900; however, as the final layer accuracy and $I(T;Y)$ remained stable, we hypothesize this represents internal "dynamic re-encoding" facilitated by the nested MLP optimizer.

4.3 Conclusion

This experiment provides empirical evidence that deep nesting in optimizers is a sufficient condition to trigger a physical compression phase in deep representations. While standard adaptive methods like AdamW effectively manage step sizes, the deeply nested structure of DMGD successfully filters descriptive noise ($I(T;X)$) from predictive signals ($I(T;Y)$) in the terminal training phase, validating the paper's theory of optimizers as multi-timescale memory systems.

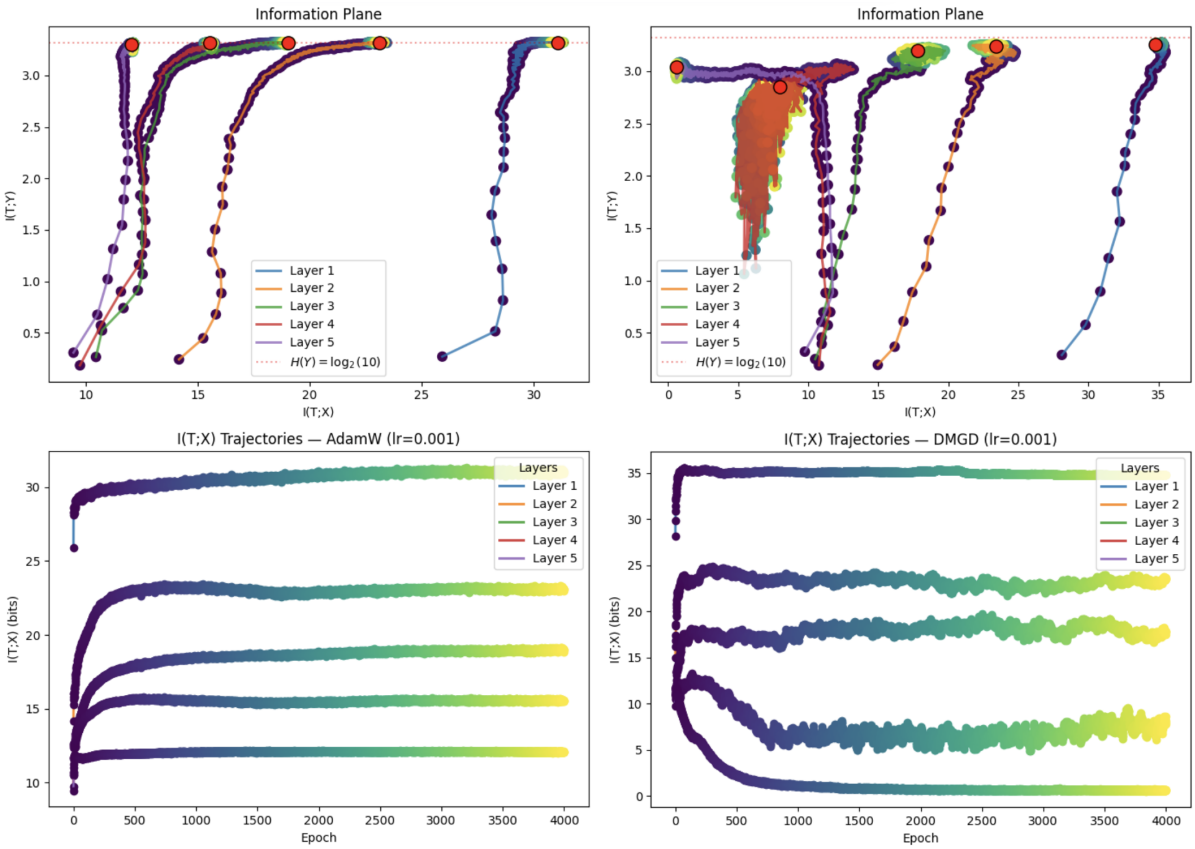

Figure 1: Empirical Validation of Nested Learning: Information Bottleneck Trajectories. The figure compares the dynamics of the AdamW (Standard Nested) and DMGD (Deep Nested) optimizers. Note the distinct compression phase (sustained $I(T;X)$ decrease) observed only in the DMGD trajectories after Epoch 150 (bottom-right panel), confirming that deeper nesting effectively filters descriptive noise from representations.