HILBERT OPERATOR FOR PROGRESSIVE ENCODING (HOPE): A MATHEMATICAL FRAMEWORK FOR DECONSTRUCTING REPRESENTATIONS IN DEEP NETWORKS

Anonymous authors
Paper under double-blind review

ABSTRACT

Deep neural networks encode complex representations, but deconstructing this internal knowledge remains a challenge. Given the link between learning and compression, network compression offers a promising lens to analyze this knowledge. However, standard compression heuristics often suffer from scale symmetries and architectural biases. To resolve these, we introduce Hilbert Operator for Progressive Encoding (HOPE), a mathematical framework to gradually deconstruct the representations in trained network weights. HOPE shifts network compression from the discrete domain into a Hilbert space of continuous functions. By modeling individual neurons as rank-1 Hilbert-Schmidt operators, HOPE unifies pruning and neuron merging as low-rank subspace projection. Extending this formulation, HOPE introduces macro block eviction to encompass multi-layer structures like entire residual pathways under the same unified metric. This unified approach enables unbiased architectural decisions across layers with different types and sizes. HOPE is a data-free and hyperparameter-free framework. We present proof-of-concept experiments in model compression and fine-tuning to highlight the practical potential of our theory.

1 INTRODUCTION

While deep neural networks learn complex representations, deconstructing this knowledge from numerical weights remains challenging. In this work, we use model compression as a measurable proxy task to study these internal representations objectively. Given the fundamental link between compression and learning Rissanen (1978); Hinton & Van Camp (1993), which has recently been reinforced by demonstrating that LLMs are general-purpose compressors Delétang et al. (2023) and amortized algorithmic predictors Genewein et al. (2026), we believe compression provides a promising lens to study this issue objectively. Deconstructing opaque networks through capacity reduction has foundational roots Mozer & Smolensky (1988). Viewed through a modern information-theoretic lens, learning is essentially the systematic discarding of task-irrelevant **noise** to isolate generalizable **core** patterns Tishby et al. (1999); Shwartz-Ziv & Tishby (2017).

Consequently, we posit that *progressive compression* is an effective tool for achieving this separation. Because core invariants resist pruning significantly longer than malleable slack, iteratively reducing capacity with minimal distortion naturally peels away the periphery to expose the network's universal feature space He et al. (2026); Nguyen et al. (2026); Wang et al. (2025). This post-training deconstruction mirrors the network's learning dynamics: the "coarse-graining" dimensionality reduction that gradient descent originally used to build them Dandi et al. (2025). Indeed, theoretical analyses confirm that networks learn by incrementally adding effective units to model increasingly complex functions Zhang et al. (2025). While biologically inspired paradigms like Behrouz et al. (2026) achieve this core-slack segregation by expanding a network to consolidate memories, we demonstrate that this separation can be accomplished efficiently through data-free compression. Despite compression's promise, the opacity of deep networks poses hurdles for objectively identifying and reducing capacity. Simple heuristics such as magnitude-based pruning (e.g. norm of raw weights) often fail because these magnitudes are typically optimization artifacts rather than importance indicators Scholl et al.

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

(2021); Tanaka et al. (2020); Hooker et al. (2019), as also long observed in network sparsification LeCun et al. (1989); Hassibi & Storkey (1992); Frankle & Carbin (2019).

To address the shortcomings of magnitude based notions of capacity, we propose to transition from the physical parameter space to the function space, specifically the underlying transformation a neuron applies to its input. This perspective treats the entire neuron, rather than individual weight matrices, as the atomic unit of the network. Note that this transformation must be analyzed over the relevant data manifold rather than the entire input space. For instance, characterizing a neuron's behavior within a dog-versus-cat image classification task requires isolating the data manifold specific to those two classes, while discarding irrelevant classes and non-natural images. While one might attempt to approximate the input manifold to each neuron empirically by passing a finite dataset through the network, such finite-dimensional approximations tether the evaluation to specific samples. This reliance on explicit data sets can render the resulting architecture brittle to distribution shifts and disproportionately degrade performance on long-tail features Hooker et al. (2019). Furthermore, empirical approaches that rely on continuous activation matching or curvature approximations Molchanov et al. (2017); Luo et al. (2017) incur a severe computational penalty, as they require numerically evaluating neural activations across the dataset *repeatedly during every iteration* of the progressive model encoding.

To avoid both the drawbacks of weight-based parameterization and the pitfalls of empirical data dependence, we introduce the Hilbert Operator for Progressive Encoding (HOPE) framework. The core philosophy of HOPE is that evaluation and analysis of neurons must occur in the function space, the resulting compression must be executed discretely on the network's parameters. By lifting the parameter vectors and matrices of individual neurons into continuous functions, HOPE models each neuron as a rank-1 Hilbert-Schmidt operator. This abstraction unifies pruning and merging under a single theoretical paradigm: an optimal low-rank projection within a functional tensor space, where distances are measured by $\|\cdot\|_{\mathcal{H}}$. Once the optimal reductions (pruning or merging) are computed in this pure functional space, the framework projects the resulting continuous operators back onto the network parameters to execute the compression.

A key advantage of HOPE is that it operates entirely **data-free** for networks utilizing Batch Normalization (BN).[1] While prior research has successfully exploited BN moving averages to generate synthetic images for data-free knowledge distillation Yin et al. (2020); Micaelli & Storkey (2019), HOPE leverages this information in a different manner. Instead of generating synthetic spatial data or relying on massive real datasets to drive input signals, HOPE applies the Maximum Entropy principle directly to the empirical BN statistics embedded within the trained model checkpoint. This yields a continuous surrogate for the local input distribution at each neuron, allowing the framework to **analytically** evaluate the integrals required for Hilbert space norms $\|\cdot\|_{\mathcal{H}}$. This fully continuous, analytical perspective largely mitigates scaling symmetries inherent in raw parameters, and thus enables unbiased capacity measurements across heterogeneous layers without requiring a single real or synthetic data sample.

The primary value of HOPE lies in establishing a rigorous, **hyperparameter free** (and also data free in presence of BN) mathematical framework for the progressive encoding of trained deep networks. We present proof-of-concept applications for model compression and fine-tuning experiments to empirically validate these theoretical capabilities.

Paper Organization. We begin with a literature review in Section 2 and formally define a neuron within our framework in Section 3. To enable data-free evaluation, Section 4 constructs a surrogate distribution constrained by Batch Normalization (BN) statistics. In Section 5, we lift discrete neurons into continuous Hilbert-Schmidt operators. Section 6 then derives a cost functional $\mathcal{J}$ that captures the distortion induced by pruning or merging via subspace projection. Section 7 extends this projection metric to the macro level to evict larger residual blocks. Section 8 introduces a rate-distortion-inspired objective that balances the distortion cost $\mathcal{J}$ against the resulting reduction in parameter count; comprising the encoding process as a loop over greedy selection of compression actions. Finally, Section 9 provides a proof-of-concept evaluation of our framework for model compression and fine-tuning.

[1] This data-free approach relies on global BN statistics. For architectures lacking BN, HOPE easily adapts: it requires only a **one-time** calibration pass over a small data batch to capture the necessary pre-activation statistics.

2 Related Works

Pruning and Parameter-Space Methods. Network compression has historically relied on parameter-space pruning, evolving from early Taylor-expansion techniques LeCun et al. (1989); Hassibi & Storkey (1992) to modern unstructured Han et al. (2015); Frankle & Carbin (2019) and structured He et al. (2017); Wen et al. (2016) approaches. HOPE directly compares against structured baselines utilizing L_1-norms Li et al. (2017) and BN scaling Liu et al. (2017). To capture functional importance, methods ranging from early skeletonization via error derivatives Mozer & Smolensky (1988) to modern empirical activation tracking Molchanov et al. (2017; 2019); Singh & Alistarh (2020); Luo et al. (2017) rely on dataset passes; however, this introduces computational bottlenecks and brittleness to distribution shifts Hooker et al. (2019). Conversely, parameter-centric heuristics suffer from "scale symmetry" Blalock et al. (2020); Renda et al. (2020); Badrinarayanan et al. (2015); Dinh et al. (2017), where overparameterized many-to-one mappings Neyshabur et al. (2015) mean raw magnitudes often reflect optimization artifacts rather than true importance Scholl et al. (2021); Tanaka et al. (2020); Hooker et al. (2019). Although methods like Lee et al. (2021) can partially mitigate global scale symmetries, they remain bound to parameter-space magnitude heuristics, which are vulnerable to within-neuron scaling artifacts. By migrating evaluation to a continuous Hilbert space, HOPE circumvents both empirical data bottlenecks and parameter-space artifacts. This shift from aligns with the Platonic Representation Hypothesis Huh et al. (2024), which posits that diverse networks converge to a shared statistical model of reality, treating weight spaces as mere shadows.

Neuron Alignment and Model Merging. Some recent methods on merging networks include permutation invariances Entezari et al. (2022); Ainsworth et al. (2023), Optimal Transport Singh & Jaggi (2020), alignment strategies Tatro et al. (2020), or feature zipping Stoica et al. (2024). Other approaches perform data-free neuron merging by evaluating pairwise parameter similarities Srinivas & Babu (2015) or clustering weights via adaptive scalar hashing Yvinec et al. (2021). However, these methods typically rely on combinatorial matching or parameter averaging. HOPE advances this paradigm by unifying pruning and merging under a single continuous operation: optimal low-rank projection in a functional tensor space. This geometric approach mirrors using subspace embeddings to capture hierarchical and compositional representations Moreira et al. (2026). Merging acts as the inverse to "feature splitting" Bricken et al. (2023) or computation in superposition Hänni et al. (2024), where overparameterized models fragment concepts across correlated sub-features, a phenomenon rooted in foundational vector space arithmetic Mikolov et al. (2013); Pennington et al. (2014) and the Linear Representation Hypothesis Park et al. (2023); Engels et al. (2024). By projecting a rank-2 neuron pair into an optimal rank-1 Hilbert-Schmidt parent, HOPE reconsolidates this distributed knowledge, reclaiming capacity while preserving the network's underlying linear geometry.

Macro Architecture and Layer Pruning. To reduce network's depth, standard methods employ stochastic depth Huang et al. (2016), heuristic layer dropping Fan et al. (2020), Neural Architecture Search Zoph & Le (2017); Liu et al. (2019), or dynamic routing gates Veit & Belongie (2018); Wang et al. (2018). However, these approaches decouple macro-architectural decisions from granular feature selection, relying on separate optimization phases or custom hyperparameters. HOPE unifies these scales by formalizing block eviction as a macroscopic function subspace projection.

Data-Free Compression and Maximum Entropy Surrogates. Bypassing the original training data often involves inverting BN statistics to generate synthetic inputs Lopes et al. (2017); Cai et al. (2020); Nagel et al. (2019); Micaelli & Storkey (2019); Yin et al. (2020) or conserving discrete synaptic flow at initialization Tanaka et al. (2020). Rather than generating explicit samples, HOPE elevates these empirical statistics via the Maximum Entropy principle Jaynes (1957) to construct a continuous analytical surrogate, similar to some task-agnostic pruning of LLMs Ma et al. (2023). Paralleling theoretical analyses of infinite-width networks and Gaussian Processes Neal (1996); Lee et al. (2018); Jacot et al. (2018); Yang (2019), this framework allows infinite-dimensional integrals to be resolved without a single forward pass.

Parameter Plasticity, Continual Learning, and Representation Deconstruction. HOPE conceptually bridges progressive compression with transfer learning Yosinski et al. (2014); Hu et al. (2022); Houlsby et al. (2019) and the Stability-Plasticity dilemma of continual learning Grossberg (1987); Kirkpatrick et al. (2017); Zenke et al. (2017). Drawing parallels to Complementary Learning Systems in cognitive neuroscience McClelland et al. (1995); Kumaran et al. (2016), and aligning with the growing recognition across deep learning Kong et al. (2026), we hypothesize that, to allow

learning without forgetting, the learned representation must be segregated into a universal core of invariants and a peripheral slack of malleable volume. While previous frameworks attempt to protect foundational knowledge against learning drift, they rely on $\mathcal{O}(N^3)$ orthogonal matrix projections Saha et al. (2021); Zeng et al. (2019); Yang et al. (2025); HuggingFace Research Team et al. (2026) or dataset-dependent quadratic penalties Kirkpatrick et al. (2017). HOPE provides a data-free alternative by computing neuron capacity in $\mathcal{O}(N)$ time. HOPE's progressive pruning and merging peel away slack and expose core abstractions, mirroring how biological circuits to maintain network stability Andrei et al. (2023). This perspective is corroborated by recent literature utilizing capacity reduction to deconstruct hierarchies. For example, targeted parameter removal has been used to explain robustness He et al. (2026), uncover heavy-tailed synaptic backbones Nguyen et al. (2026), and demonstrate hierarchical learning through progressive feature compression Wang et al. (2025). Collectively, these works support the premise that HOPE's neuron capacity may efficiently identify foundations versus plastic slack, laying the groundwork for transfer learning and downstream algorithmic interpretability Bau et al. (2017); Morcos et al. (2018); Olah et al. (2020); Neyshabur et al. (2017).

3 The Neuron

Scale invariances are inherently relational and that is why isolated parameters fail to capture them. The minimal architectural unit where these symmetries fully manifest is the neuron in its entirety (encapsulating its incoming weights, BN parameters, non-linear activation, and outgoing weights). In this section we discuss how these within-neuron scale symmetries can be factored out, while deferring global scale symmetries to Section 5. For clarity, we derive our theoretical framework under the assumption of a fully connected architecture. However, this formalism seamlessly extends to convolutional networks (see Appendix B.1). Indeed, the architectures evaluated in Section 9 rely on this adaptation. Consider a neuron i with input weights $\boldsymbol{w}_{\text{raw},i} \in \mathbb{R}^n$ and output weights $\boldsymbol{w}_{\text{out},i} \in \mathbb{R}^c$. Let the neuron be subject to learnable BN parameters (γ_i, β_i) and dataset statistics (μ_i, σ_i^2), where $\mu_i \triangleq \mathbb{E}_{\mathcal{X}}[\boldsymbol{w}_{\text{raw},i}^T \boldsymbol{x}]$ and $\sigma_i^2 \triangleq \text{Var}_{\mathcal{X}}(\boldsymbol{w}_{\text{raw},i}^T \boldsymbol{x})$. To capture the signal reaching the non-linearity, we absorb the normalization operations into a set of *effective* parameters:

$$\boldsymbol{w}_{\text{in},i}^{\text{eff}} \triangleq \gamma_i / \sqrt{\sigma_i^2 + \epsilon} \boldsymbol{w}_{\text{raw},i} \quad , \quad b_i \triangleq \beta_i - \gamma_i \mu_i / \sqrt{\sigma_i^2 + \epsilon}\,, \tag{1}$$

where $\epsilon > 0$ is a small constant. The neuron's end-to-end signal mapping, capturing its functional contribution to the subsequent layer for an input $\boldsymbol{x}$, is defined by the continuous function:

$$f_i(\boldsymbol{x}) = \Psi(y_i) \boldsymbol{w}_{\text{out},i} \quad , \quad y_i \triangleq (\boldsymbol{w}_{\text{in},i}^{\text{eff}})^T \boldsymbol{x} + b_i\,, \tag{2}$$

where $\Psi(\cdot)$ denotes a **Positively Homogeneous of degree 1 (PH-1)** activation function. An activation function $\Psi : \mathbb{R} \to \mathbb{R}$ is PH-1 if it satisfies the scaling property $\Psi(cz) = c\Psi(z)$ for all $z \in \mathbb{R}$ and all scalars $c \geq 0$. Examples include `ReLU`, `Leaky ReLU`.

Throughout the HOPE framework, we formally designate this continuous function f_i as the **neuron**. Operating at this atomic level mitigates two primary sources of scaling symmetry: *normalization invariance* (from BN) and *re-parameterization invariance* (from cross-layer weight resharding). We detail the former below, and defer the latter to Section 5.

3.1 Mitigating Normalization Invariance

Normalization invariance arises from BN's standardizing mechanics. Scaling raw input weights $\boldsymbol{w}_{\text{raw},i}$ by a constant factor $\lambda > 0$ increases the pre-activation variance by λ^2. The subsequent BN layer divides by the standard deviation, canceling λ before the non-linearity. Because downstream output remains unchanged, raw weight magnitudes can be deceptive. HOPE mitigates this by evaluating capacity through effective parameters ($\boldsymbol{w}_{\text{in},i}^{\text{eff}}$ and b_i). While HOPE uses raw weights $\boldsymbol{w}_{\text{raw},i}$ alongside BN statistics to construct the data-constrained surrogate dataset $P_{\mathcal{X}}$ (Section 4), their utility ends there. The framework then evaluates the neuron's continuous-functional impact on this surrogate.

Physically, the non-linear activation processes the normalized, shifted signal, not the raw parameter projection. Thus, when computing the continuous integral to compute Hilbert space norms or inner product, HOPE defines the pre-activation signal as $y_i = (\boldsymbol{w}_{\text{in},i}^{\text{eff}})^T \boldsymbol{x} + b_i$. Because $\boldsymbol{w}_{\text{in},i}^{\text{eff}}$ divides by the empirical standard deviation σ_i, any magnitude inflation is canceled. By evaluating the signal impacting the activation function, HOPE guarantees the capacity criterion reflects functional utility rather than scale artifacts.

4 THE NEURAL SIGNAL DISTRIBUTION

Since HOPE operates in a data-free regime, the true input distribution $P^*_{\mathcal{X}}$ (encompassing both initial data and subsequent layer activations) is inaccessible. Yet because we model neurons as continuous Hilbert-Schmidt operators, evaluating their inner products requires integrating over the data distribution. To resolve this, we invoke the Maximum Entropy principle Jaynes (1957) to construct a Gaussian surrogate.

Effectiveness of Gaussian. While a Gaussian approximation may seem idealistic, we argue in two steps why it aligns with modern neural architectures. **1. Gaussianity of Pre-Activation.** While a neuron's true input distribution $P^*_{\mathcal{X}}$ often lies on a complex manifold, the neuron observes its input $\boldsymbol{x}$ through 1-dimensional linear projections $y = \sum_{j=1}^{n} w_j x_j$. As n grows, by the Central Limit Theorem and the Diaconis-Freedman effect Diaconis & Freedman (1984), the pre-activation y converges to a Gaussian distribution. **2. Gaussian Surrogate for the Input $\boldsymbol{x}$.** Since neurons are oblivious to the true shape of the data manifold and perceive their input as a Gaussian signal y, for theoretical convenience, we substitute the complex true data with a tractable surrogate $P_{\mathcal{X}} = \mathcal{N}(\hat{\boldsymbol{\mu}}_x, \hat{\Sigma}_x)$.

Optimal Surrogate. To define the optimal parameters $(\hat{\boldsymbol{\mu}}_x, \hat{\Sigma}_x)$ of the unknown data distribution, we can incorporate the two empirical constraints $\boldsymbol{x} \in \mathbb{R}^n$: $\mathbb{E}[\boldsymbol{w}_{\text{raw},i}^T \boldsymbol{x}] = \mu_i$ and $\text{Var}(\boldsymbol{w}_{\text{raw},i}^T \boldsymbol{x}) = \sigma_i^2$ imposed by BN for any i. We define a shared surrogate for each layer. Since the empirical means $\boldsymbol{\mu}_{\text{BN}} \in \mathbb{R}^n$ represent the 1D shadow of the dataset's center cast through the raw weights, to find the best mean for a layer we compute the optimal least-squares approximation of the dataset's center using the Moore-Penrose pseudo-inverse $\boldsymbol{W}_{\text{raw}}^+$, which leads to $\hat{\boldsymbol{\mu}}_x = \boldsymbol{W}_{\text{raw}}^+ \boldsymbol{\mu}_{\text{BN}}$. To find the optimal covariance matrix $\hat{\Sigma}_x$ for the layer, we maximize the differential entropy of the multivariate Gaussian, $H(\boldsymbol{x}) \propto \log\det(\Sigma_x)$ subject to the BN variance constraints $\boldsymbol{w}_{\text{raw},i}^T \Sigma_x \boldsymbol{w}_{\text{raw},i} = \sigma_i^2$ for $i \in \{1, \ldots, c\}$. This can be solved efficiently without need to invert large covariance matrices. The key is that our pairwise compression operation (neuron pair merging, as will be introduced later) is restricted to a rank-2 subspace, and this allows a closed-form solution in terms of the neural kernel (Appendix G.3).

5 A HILBERT FUNCTIONAL PERSPECTIVE ON NEURONS

To mitigate parameter shape bias and facilitate the identification of dead neurons, we transition from discrete parameter to a continuous function formulation. By embedding each neuron into a Hilbert space[2] $\mathcal{H}$ and treating it as a rank-1 Hilbert-Schmidt operator, our framework evaluates the actual function the neuron computes, effectively *abstracting away its physical matrix shape*. Furthermore, by integrating this continuous function over the analytically derived surrogate $P_{\mathcal{X}}$, HOPE identifies dead neurons via a closed-form expectation, bypassing the need for computationally expensive empirical forward passes over a dataset. Together, these features shift the evaluation criterion from raw parameter counts to functional capacity, quantified as the norm of the neuron's function in $\mathcal{H}$.

We model each neuron as a rank-one **Hilbert operator** $f_i \in \mathcal{H}$. This way, a neuron's identity is not defined **by its evaluation on a single input point $\boldsymbol{x}$**, but its continuous behavior over **the entire surrogate distribution** $\mathcal{X}$. We quantify the capacity of the neuron by $\|f_i\|_{\mathcal{H}}$. We now present this formally. We define our space of neural functions as $\mathcal{H} \triangleq L_2(\mathcal{X}, P_{\mathcal{X}}; \mathbb{R}^c)$, the space of square-integrable functions mapping $\mathcal{X}$ to the c-dimensional output space. Define $g_i : \mathcal{X} \to \mathbb{R}$ as $g_i(\boldsymbol{x}) \triangleq \Psi\left((\boldsymbol{w}_{\text{in},i}^{\text{eff}})^T \boldsymbol{x} + b_i\right)$. We embed g_i as an element of the scalar Hilbert space $\mathcal{H}_{\text{in}} \triangleq L_2(\mathcal{X}, P_{\mathcal{X}}; \mathbb{R})$. Furthermore, the scalar activation of a neuron is sent to the next layer by scaling with the finite-dimensional output weight vector $\boldsymbol{w}_{\text{out},i} \in \mathcal{H}_{\text{out}} \triangleq \mathbb{R}^c$. Since the output across all c dimensions is confined to the one-dimensional subspace spanned by $\boldsymbol{w}_{\text{out},i}$, this entire continuous landscape is embedded exclusively along a single vector direction. By taking the tensor product of the input function and the output vector, we construct a linear mapping across these spaces: $\mathcal{H} \cong \mathcal{H}_{\text{in}} \otimes \mathcal{H}_{\text{out}}$. Thus the vector-valued function $f_i : \mathcal{X} \to \mathcal{H}_{\text{out}}$, i.e. the neuron, is an element within this tensor product space $f_i \triangleq g_i \otimes \boldsymbol{w}_{\text{out},i}$. Since this element is constructed from the outer product of one input function and one output vector, each individual neuron f_i is a rank-1 Hilbert-Schmidt operator. This tensor formulation is fundamental for defining the merging operation in HOPE as we will discuss in Sections 6 and E.1.

[2]See Appendix A for a brief introduction to Hilbert spaces.

Hilbert-Schmidt Inner Product and Capacity. Because our neurons are defined as rank-1 operators residing in the tensor $\mathcal{H} \cong \mathcal{H}_{\text{in}} \otimes \mathcal{H}_{\text{out}}$, we must evaluate their geometric relationship using the inner product defined on this composite space: $\langle f_i, f_j \rangle_{\mathcal{H}} = \langle g_i \otimes \boldsymbol{w}_{\text{out},i}, g_j \otimes \boldsymbol{w}_{\text{out},j} \rangle_{\mathcal{H}} = \langle g_i, g_j \rangle_{\mathcal{H}_{\text{in}}} \cdot \langle \boldsymbol{w}_{\text{out},i}, \boldsymbol{w}_{\text{out},j} \rangle_{\mathcal{H}_{\text{out}}} = \mathbb{E}_{\boldsymbol{x} \sim P_{\mathcal{X}}}[\Psi(y_i)\Psi(y_j)] \cdot \langle \boldsymbol{w}_{\text{out},i}, \boldsymbol{w}_{\text{out},j} \rangle_{\mathbb{R}^c}$. We define the **capacity** of a neuron as its Hilbert-Schmidt norm $\|f_i\|_{\mathcal{H}} = \sqrt{\langle f_i, f_i \rangle_{\mathcal{H}}}$, which we will later use to decide what neuron to prune and which macro block to evict.

Kernel Formulation. We define the kernel of two neurons i, j as $K(i,j) \triangleq \mathbb{E}_{\boldsymbol{x} \sim P_{\mathcal{X}}}[\Psi(y_i)\Psi(y_j)]$. Under this, the capacity of a neuron simplifies to $\|f_i\|_{\mathcal{H}} = \|\boldsymbol{w}_{\text{out},i}\|_2 \cdot \sqrt{K(i,i)}$ and the inner product of two neurons becomes $\langle f_i, f_j \rangle_{\mathcal{H}} = \langle \boldsymbol{w}_{\text{out},i}, \boldsymbol{w}_{\text{out},j} \rangle_{\mathbb{R}^c} K(i,j)$. Closed form expression for these kernels are provided at the end of this section, and their derivation is presented in Appendix G when Ψ is the `ReLU` activation function. Below we only list the final form of these kernels:

$$K(i,i) = (\gamma_i^2 + \beta_i^2)\Phi(\frac{\beta_i}{|\gamma_i|}) + \beta_i |\gamma_i| \phi(\frac{\beta_i}{|\gamma_i|}) \quad , \quad K(i,j) \approx \frac{1}{\pi}(\sqrt{1-\hat{\rho}_{ij}^2} + (\pi - \arccos\hat{\rho}_{ij})\hat{\rho}_{ij})\sqrt{K(i,i)K(j,j)}\,,$$

where ϕ and Φ are the standard Normal PDF and CDF respectively and $\hat{\rho}_{ij}$ is defined as below.

$$\rho_{\text{eff}} \triangleq \frac{\langle \boldsymbol{w}_{\text{in},i}^{\text{eff}}, \boldsymbol{w}_{\text{in},j}^{\text{eff}} \rangle}{\|\boldsymbol{w}_{\text{in},i}^{\text{eff}}\|_2 \, \|\boldsymbol{w}_{\text{in},j}^{\text{eff}}\|_2} \quad , \quad \kappa \triangleq \frac{\rho_{\text{eff}} \, |\gamma_i \, \gamma_j|}{(1-\rho_{\text{eff}}^2) \, \|\boldsymbol{w}_{\text{in},i}^{\text{eff}}\|_2 \, \|\boldsymbol{w}_{\text{in},j}^{\text{eff}}\|_2} \quad , \quad \hat{\rho}_{ij} \triangleq \frac{2\kappa}{1 + \sqrt{1+4\kappa^2}}\,. \tag{3}$$

Neuron Scale Invariance. In networks with PH-1 activations, scaling $\boldsymbol{w}_{\text{in},i}^{\text{eff}}$ and b_i of a neuron by $\lambda > 0$ and $\boldsymbol{w}_{\text{out},i}$ by $1/\lambda$ alters weight magnitudes without changing the function. This symmetry confounds raw magnitude-based criteria. HOPE mitigates this by defining capacity as the Hilbert norm: $\|f_i\|_{\mathcal{H}} = \|\boldsymbol{w}_{\text{out},i}\|_2 \sqrt{K(i,i)}$. Because positive homogeneity scales the kernel K by λ, the opposing factors cancel.

Neuron Shape Invariance. By definition, the neuron's functional capacity $\|f_i\|_{\mathcal{H}} = \|\boldsymbol{w}_{\text{out},i}\|_2 \sqrt{K(i,i)}$ depends on the input space $\mathcal{X}$ solely through the kernel term $K(i,i) \triangleq \mathbb{E}_{\boldsymbol{x} \sim P_{\mathcal{X}}}[\Psi^2(y_i)]$. Rather than counting discrete incoming parameters n or computing weight magnitudes such as $\|\boldsymbol{w}_{\text{in},i}^{\text{eff}}\|$ or $\|\boldsymbol{w}_{\text{raw},i}\|$, this formulation abstracts away the physical dimensionality of the input tensor $\boldsymbol{x} \in \mathbb{R}^n$. Furthermore, when paired with the layer-wise magnitude neutrality axiom, the neuron's evaluation is entirely invariant to both its input and output dimensions.

Neuron Merging via Hilbert-Schmidt Projection. Similar to pruning, merging also reduces the network's neuron count by one, but it can provide a higher-fidelity reduction when the selected pair have a strong cosine similarity in $\mathcal{H}$. We define the merger through an optimal Hilbert-Schmidt projection. Since neurons i, j are each rank-1 operators, their joint contribution $[f_i, f_j]$ spans a 2-dimensional subspace in $\mathcal{H}$ consisting of operators of rank at most 2. Merging them into a single parent neuron is defined as finding the optimal rank-1 approximation of this tensor subspace.

6 Layer Transition Costs

Scale symmetries also manifest globally: shallow layers processing high-variance data often yield neurons with larger capacities than deeper layers operating on compressed latent representations Hanin & Rolnick (2018); Tanaka et al. (2020). For a compression method to account for this bias, it must evaluate individual neurons within the context of their entire layer Lee et al. (2021). We formalize this global context via a *layer state* $\Phi \triangleq (f_1, f_2, \ldots, f_N)$, where N is the number of active neurons in the layer. A single compression step maps an initial state Φ_a to a reduced state Φ_b, where $|\Phi_b| = N - 1$. The entire compression process is thus a chain of discrete state transitions across various layers, guided by a cost $\mathcal{J}(\Phi_a, \Phi_b) > 0$ that quantifies the resulting model distortion. This section focuses on deriving this $\mathcal{J}$.

To develop a layer cost $\mathcal{J}$, we first extend the single-neuron capacity, $\|f_i\|$, to define a layer capacity $E(\Phi)$ for state Φ, where $E(f_1) = \|f_1\|$. A natural requirement is that $E(\Phi)$ remains invariant to arbitrary neuron partitioning. Assuming $E(\Phi)$ is a symmetric, separable, and homogeneous functional of individual capacities, this condition uniquely determines $E(\Phi) = \sum_{k=1}^{N} \|f_k\|_{\mathcal{H}}$ (by Lemma C.1). For some intuition, suppose that $E(\Phi) = (\sum \|f_k\|_{\mathcal{H}}^p)^{1/p}$. Partitioning a neuron f_0 into M fractions f_0/M yields $M^{(1-p)/p}\|f_0\|_{\mathcal{H}}$. Capacity invariance for any M requires $(1-p)/p = 0$, yielding $p = 1$.

Under a few natural axioms, including Magnitude Neutrality ($\mathcal{J}$ must be scale invariant: $\forall k > 0\,;\, \mathcal{J}(k\,\Phi_a, k\,\Phi_b) = \mathcal{J}(\Phi_a, \Phi_b)$) and Connectivity Preservation ($\mathcal{J}$ must prevent layer extinction: $\lim_{E(\Phi_b)\to 0^+} \mathcal{J} = \infty$), we can obtain (see Appendix D)the following form for $\mathcal{J}$:

$$\mathcal{J} \triangleq \frac{N \cdot D(\Phi_a, \tilde{\Phi}_b)}{E(\Phi_b)} \quad \text{where} \quad D(\Phi_a, \tilde{\Phi}_b) \triangleq \|\Phi_a - \tilde{\Phi}_b\|_{\mathcal{H}^N} = \Big(\sum_{k=1}^{N} \|f_k^{(a)} - \tilde{f}_k^{(b)}\|_{\mathcal{H}}^2\Big)^{\frac{1}{2}}. \tag{4}$$

Here $\tilde{\Phi}_b \in \mathcal{H}^N$ is the *pre-deletion* target at $t = 1$ (e.g., a duplicated parent $[f_p, f_p]$) but the N-dimensional structure remains intact. Conversely, $\Phi_b \in \mathcal{H}^{N-1}$ is the true *terminal state*: the layer after the neuron is dropped. This separation ensures no dimensional mismatch in the arguments of D, while the denominator only relies on the $(N-1)$-dimensional post-deletion capacity $E(\Phi_b)$.

Now we can instantiate the pruning and merging costs as special cases of $\mathcal{J}$. Pruning a neuron f_i corresponds to projecting its rank-1 operator down to the null operator $\mathbf{0}$. Because the perturbed subspace only contains this single neuron i and its terminal state is $\mathbf{0}$, the projection error simplifies to $D = \sqrt{\|f_i - \mathbf{0}\|_{\mathcal{H}}^2} = \|f_i\|_{\mathcal{H}}$. Furthermore, the terminal capacity evaluates to $E(\Phi_b) = E_a - \|f_i\|_{\mathcal{H}}$, which determines the final pruning cost $\mathcal{J}_{\text{prune}}$.

Merging a neuron pair is slightly more involved. For neurons i and j, their joint operator $[f_i, f_j]$ spans a rank-2 subspace in $\mathcal{H}$. Because f_i and f_j are vector-valued functions, their joint operator is matrix-valued, denoted as $\boldsymbol{W}_{\text{joint}} \triangleq [f_i, f_j]$. Merging compresses this into a rank-1 approximation $\boldsymbol{W}'_{\text{joint}}$. Classic unconstrained rank truncation prescribes a rank-one basis f_b and independent scaling factors $\alpha, \beta \in \mathbb{R}$ by solving $\min_{f_b \in \mathcal{H}, \alpha, \beta \in \mathbb{R}} \|\boldsymbol{W}_{\text{joint}} - \boldsymbol{W}'_{\text{joint}}\|_{\mathcal{H}}^2$, where $\boldsymbol{W}'_{\text{joint}} = [\alpha f_b, \beta f_b]$. However, because a physical neuron must produce a single unified output, we must restrict the valid replacement pair to $\boldsymbol{W}'_{\text{joint}} = [f_p, f_p]$. This enforces the constraint $\alpha = \beta = 1$, yielding the constrained objective $\min_{f_p \in \mathcal{H}} \|\boldsymbol{W}_{\text{joint}} - \boldsymbol{W}'_{\text{joint}}\|_{\mathcal{H}}^2$ and rendering standard unconstrained projections inapplicable. Deferring the derivation of the optimal parent f_p^* to Appendix E.1, we first establish the objective functional itself. We expand $D^2(\Phi_a, \Phi_b)$ as follows:

$$D^2(\Phi_a, \Phi_b) = \|\boldsymbol{W}_{\text{joint}} - \boldsymbol{W}'_{\text{joint}}\|_{\mathcal{H}}^2 = \mathbb{E}_{\boldsymbol{x} \sim P_{\mathcal{X}}}\left[\|\boldsymbol{W}_{\text{joint}}(\boldsymbol{x}) - \boldsymbol{W}'_{\text{joint}}(\boldsymbol{x})\|_F^2\right] = \|f_i - f_p\|_{\mathcal{H}}^2 + \|f_j - f_p\|_{\mathcal{H}}^2 .$$

The terminal capacity E_b updates by swapping the eliminated children for the new parent: $E_b = E_a - \|f_i\|_{\mathcal{H}} - \|f_j\|_{\mathcal{H}} + \|f_p\|_{\mathcal{H}}$. Substituting D and E_b yields the final merging cost.

$$\mathcal{J}_{\text{prune}} = \frac{N\,\|f_i\|_{\mathcal{H}}}{E_a - \|f_i\|_{\mathcal{H}}} \quad , \quad \mathcal{J}_{\text{merge}} = \frac{N\,\sqrt{\|f_i - f_p\|_{\mathcal{H}}^2 + \|f_j - f_p\|_{\mathcal{H}}^2}}{E_a - \|f_i\|_{\mathcal{H}} - \|f_j\|_{\mathcal{H}} + \|f_p\|_{\mathcal{H}}} . \tag{5}$$

7 Block Eviction

This section expands the granular compression cost $\mathcal{J}_{\text{bound}}$ established in Section 6 to a new macro-level operation: **block eviction**. Focusing on residual blocks in architectures like ResNet-50, we extend the previously developed continuous integral to evaluate block eviction alongside granular operations within a single, unified mathematical framework. Consider the canonical residual block, which processes an input representation X (capitalized to distinguish it from the flattened vector $\boldsymbol{x}$) through a three-stage mapping pathway $F(X)$. This pathway sequentially applies weight parameters W_1, W_2, and W_3, and the result is added to a skip connection to yield the final pre-activation $Y = X + F(X)$. We define **Block Eviction** as forcing $F(X) \to \mathbf{0}$, and thus collapsing the block into a pure identity mapping $Y = X$ (see Figure 1).

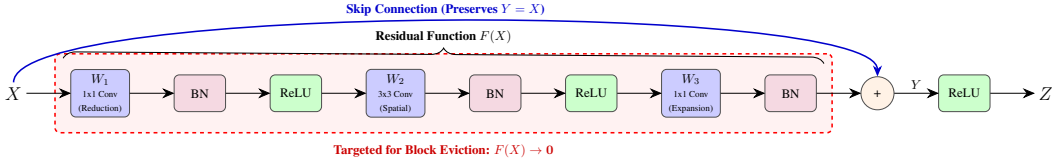


Figure 1: Illustration of the canonical ResNet V1 residual block and its eviction process. Eviction forces the internal pathway $F(X) \to \mathbf{0}$, collapsing the block into a pure identity pre-activation $Y = X$.

To evaluate this macro-operation, we must expand our definition of layer state. We formulate a macroscopic state $\Omega^{(l)} \triangleq (\Phi^{(l)}, \mathcal{I})$ that couples the targeted internal layer $\Phi^{(l)}$ with the ambient skip connection $\mathcal{I}$. For a residual bottleneck comprising two internal convolution layers $l \in \{1, 2\}$ (see Appendix H) the total macroscopic distortion is:

$$\mathcal{J}_{\text{evict}} = \sum_{l=1}^{2} \mathcal{J}_{\text{layer}}(\Omega_a^{(l)}, \Omega_b^{(l)}) = \sum_{l=1}^{2} N_{\text{active}}^{(l)} \left(\frac{E_{\text{active}}^{(l)}}{E_{\text{identity}}} \right).$$

Here, $N_{\text{active}}^{(l)}$ and $E_{\text{active}}^{(l)}$ represent the active operator count and surviving capacity of internal layer l, respectively. The parallel survival capacity evaluates to the expected RMS energy of the identity operators conditioned by the preceding BN layer: $E_{\text{identity}} = \sum_{k=1}^{d_{\text{amb}}} \sqrt{\gamma_k^2 + \beta_k^2}$.

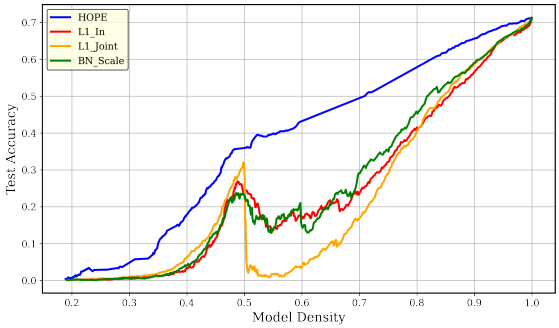
8 Balancing Compression and Distortion

All cost functionals discussed thus far ($\mathcal{J}_{\text{prune}}$, $\mathcal{J}_{\text{merge}}$, and $\mathcal{J}_{\text{evict}}$) measure the projection error incurred when transitioning from a given state to a reduced state. However, the distortion alone cannot characterize lossy compression: lower signal distortion requires a higher bit count, while stronger compression inevitably increases distortion. We therefore need to declare a bit count budget and minimize total distortion under that budget. Formally, let $\mathcal{A} = \{1, 2, \ldots, K\}$ denote the set of all admissible compression operations at the current encoding iteration, encompassing all feasible granular and macro operations. Each action k incurs a distortion penalty $\mathcal{J}_k$ and releases ΔP_k parameters (see Appendix B.2 for details on computing ΔP). Akin to a continuous knapsack problem Dantzig (1957), we choose the best action as $k^* = \arg\min_{k \in \mathcal{A}} \frac{\mathcal{J}_k}{\Delta P_k}$.

9 Proof-of-Concept Applications

9.1 Model Compression

Because its encoding is progressive, any intermediate iteration serves as a valid compressed model, providing users with flexible trade-offs between compression rate and fidelity. Taxonomically, HOPE is a structured method: it eliminates entire neurons rather than zeroing out individual weights. This provides greater practical utility than unstructured pruning, which generates randomly sparse matrices requiring specialized hardware to realize actual computational speedups. We compare HOPE against three structured baselines that eliminate neurons below specific magnitude thresholds: *L_1-Norm Input Pruning* Liu et al. (2017) (scored by incoming weight L_1 norms); *L_1-Norm Joint Pruning* (scored by concatenated incoming and outgoing L_1 norms); and *BN Scale Pruning* Liu et al. (2017) (using the BN scaling factor γ as a proxy for importance).



Our experiments investigated the relationship between test set accuracy and model density (defined as the ratio of active to initial neurons across the entire network). For our compression assessment, we utilized Keras' publicly available ResNet-50 model checkpoint that is pre-trained on ImageNet. As demonstrated in the plot, HOPE yields models with superior accuracy compared to the baselines.

9.2 Cross-Domain Transfer Learning

In this section we use HOPE to develop **Dispersed Elastic Fine-Tuning (DEFT)**. The goal is to partition the neurons in the network into Universal Core or Peripheral Slack through a binary elasticity parameter $E \in \{0, 1\}$ for each neuron. The elasticity E is derived from the pruning cost assigned to each neuron i upon its removal during HOPE's progressive encoding process:

$$E_i = \begin{cases} 1 & \text{if } \mathcal{J}_{\text{prune}}^{(i)} < J_{\text{lock}} \\ 0 & \text{if } \mathcal{J}_{\text{prune}}^{(i)} \geq J_{\text{lock}} \end{cases}, \quad \mathcal{J}_{\text{prune}}^{(i)} = \frac{N^{(i)} \cdot \|f_i\|_{\mathcal{H}}}{E_b^{(i)}}$$

where $N^{(i)}$ and $E_b^{(i)} \triangleq E_a^{(i)} - \|f_i\|_{\mathcal{H}}$ denote the active neuron count and the remaining layer capacity, at the step neuron i is pruned, and J_{lock} is a threshold value. The idea is that, high capacity neurons essential to the source architecture $\mathcal{J}_{\text{prune}}^{(i)} \geq J_{\text{lock}}$ are frozen by $E_i = 0$, whereas low-capacity slack neurons are granted high plasticity via $E_i = 1$.

Obtaining J_{lock}. We collect all pruning costs across the entire encoding process into a set $\mathcal{C}$. We then filter out from $\mathcal{C}$ the extinction artifacts, i.e. elements that satisfy$E_b^{(i)} < \epsilon$ for a small value ϵ. We then compute the threshold $J_P = \text{Percentile}(\mathcal{C}, P)$ and the supremum $J_{\sup} = \max(\mathcal{C})$. Note that unlike HOPE, DEFT has a (single) hyperparamter, which is the value of the percentile P. For numerical stability, the final locking threshold J_{lock} is defined $J_{\text{lock}} = J_P$ if $J_P \geq \epsilon$; $J_{\sup}$ if $J_P < \epsilon \leq J_{\sup}$; and 1 otherwise.

Consistency at Initialization (The Structural Mask): To prevent target-driven updates of the plastic slack from corrupting the frozen core, DEFT applies a structural mask at initialization. It severs all connections pointing from upstream plastic neurons to downstream frozen core neurons. For a weight tensor connecting an upstream layer (with elasticities $\boldsymbol{E}_{\text{in}}$) to a downstream layer (with elasticities $\boldsymbol{E}_{\text{out}}$), the mask $\boldsymbol{M}$ enforces $M_{j,k} = 0$ if $E_{\text{in},k} > 0$ and $E_{\text{out},j} = 0$; and 1 otherwise. The initial weights for the target task are thus constrained to $\boldsymbol{W}_0 = \boldsymbol{M} \odot \boldsymbol{W}_{\text{source}}$.

Gradient Scaling: During optimization, the target gradients are element-wise scaled by the downstream neuron's elasticity $\boldsymbol{E}_{\text{out}}$ (uniformly broadcast across the input channels) $\boldsymbol{g}_t = \boldsymbol{E}_{\text{out}} \odot \nabla_{\boldsymbol{W}} \mathcal{L}_{\text{target}}(\boldsymbol{W}_t)$, where $\boldsymbol{g}_t$ represents the effective gradient passed to the optimizer state, e.g., SGD with momentum.

Experimental Setup. To evaluate the stability-plasticity tradeoff, we rank methods using the **H-Score** commonly used in continual learning Qiu et al. (2024); Xie et al. (2025); Islam et al. (2025), defined as the harmonic mean of source retention and target accuracy. We evaluate models pre-trained on multi-class classification tasks derived from the CIFAR-100 dataset Krizhevsky (2009). Each source task is constructed by randomly sampling 4 superclasses, which yields 20 fine-grained categories (5 per superclass). We then transfer these models to the full 10-class digit classification task in the SVHN dataset (street-level house numbers) Netzer et al. (2011).

Baseline Methods. 1. Standard Full FT: The entire pre-trained backbone is unfrozen, allowing the optimizer to alter representations across all layers. 2. Head-Only FT: The pre-trained backbone is frozen (acting as a static feature extractor), and only the final linear classification head is optimized. 3. PEFT (BN-Tuning): Operating on the premise that feature extraction should remain invariant Frankle et al. (2021), this method applies a binary gradient mask, where core convolutional and dense kernels are frozen while plasticity is granted only to the BN parameters and the newly initialized classification head. 4. EWC (Elastic Weight Consolidation): Allows all parameters to update but applies a quadratic penalty constraining parameters deemed critical to the source task Kirkpatrick et al. (2017). Full details regarding network architecture, hyperparameter optimization, and reproducibility protocols are provided in Appendix I.

Results. The test set evaluations across 4 independent cross-domain trials (20 discrete CIFAR-100 $\rightarrow$ SVHN scenarios) are presented in Table 1. While Standard Full FT achieves the highest target performance 94.09% by freely overwriting network weights, it completely destroys the source representation, crashing source retention to a baseline low of 7.52%. Conversely, Head-Only FT best preserves source knowledge, but its 36.11% target accuracy highlights the severe domain gap; pre-trained features are insufficient to separate SVHN digits linearly. EWC behaves remarkably similarly to Full FT: it learns the target domain (93.94%) but fails to significantly arrest catastrophic forgetting (6.74%). DEFT successfully bridges this stability-plasticity gap. By routing target gradients into elastic neurons characterized by their low capacity, DEFT captures nearly all the plasticity of Standard Full FT (94.09% vs. 89.79%). Simultaneously, by masking the core foundational features, it halts catastrophic forgetting, retaining 52.14% of the source accuracy. Note that DEFT outperforms all the baselines in terms of H-Score.

Table 1: Cross-Domain Transfer Learning (4 Trials, 5 tasks each). Metrics represent Test Set accuracy evaluated at the optimal target validation epoch. All metrics are averaged across all tasks and trials.

Method	Target Acc (SVHN)	Source Retention (CIFAR)	H-Score
DEFT (Ours)	89.79 ± 0.84	52.14 ± 5.29	$\mathbf{65.82 \pm 3.96}$
Head-Only	36.11 ± 2.79	63.13 ± 4.62	45.79 ± 2.05
Full FT	94.09 ± 0.28	7.52 ± 1.63	13.88 ± 2.84
EWC	93.94 ± 0.22	6.74 ± 1.74	12.54 ± 2.99
PEFT	81.91 ± 0.49	5.44 ± 0.98	10.18 ± 1.63

10 AI DISCLOSURE

In this work, we used generative AI tools for: assist in the writing of proofs. We have not used generative AI tools for: Generate synthetic data sets, help develop theoretical models or conceptual frameworks, implement methods, formulate mathematical claims, provide critical ingredients for proving mathematical claims, propose or refine hypotheses, design or provide feedback on research methodology or experiments, clean and reformat dataset, support qualitative and thematic data analysis, interpret results., and assist with translation are not applicable to this work. Additionally, we used generative AI tools for: create or modify scientific figures or images, create or edit software code, summarize or analyse existing literature, edit a research paper to improve readability, identify relevant literature, format references. We have reviewed all AI-assisted work: LLM-generated code and tikz plots were verified and tested for correctness by author. We take responsibility for the final content of this work, including text, claims or artifacts produced with the aid of generative AI.

REFERENCES

Samuel K Ainsworth, Jonathan Hayase, and Siddhartha Srinivasa. Git re-basin: Merging models modulo permutation symmetries. In *International Conference on Learning Representations*, 2023.

Andrei Andrei, Anissa Akil, Nicholas Kharas, Robert Rosenbaum, Krešimir Josić, and Valentin Dragoi. Rapid compensatory plasticity revealed by dynamic correlated activity in monkeys in vivo. *Nature Neuroscience*, 26(11):1960–1969, 2023.

Vijay Badrinarayanan, Bamdev Mishra, and Roberto Cipolla. Understanding symmetries in deep networks. *arXiv preprint arXiv:1511.01029*, 2015.

David Bau, Bolei Zhou, Aditya Khosla, Aude Oliva, and Antonio Torralba. Network dissection: Quantifying interpretability of deep visual representations. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 6541–6549, 2017.

Ali Behrouz, Farnoosh Hashemi, Adel Javanmard, and Vahab Mirrokni. Language models need sleep: Learning to self-modify and consolidate memories, 2026. URL https://arxiv.org/abs/2606.03979.

Dimitri P Bertsekas. *Dynamic programming and optimal control: Volume I*. Athena scientific, 2012.

Anne Billot, Niharika Jhingan, Maria Varkanitsa, Idan Blank, Rachel Ryskin, Swathi Kiran, and Evelina Fedorenko. The language network ages well: Preserved selectivity, lateralization, and within-network functional synchronization in older brains. *bioRxiv*, 2024.

Davis Blalock, Jose Javier Gonzalez Ortiz, Jonathan Frankle, and John Guttag. What is the state of neural network pruning? In *Proceedings of Machine Learning and Systems*, volume 2, pp. 129–146, 2020.

Idan Blank, Nancy Kanwisher, and Evelina Fedorenko. A functional mri investigation of the language network. *Journal of Neurophysiology*, 116(4):1968–1984, 2016.

Trenton Bricken, Adly Templeton, Joshua Batson, Brian Chen, Adam Jermyn, Tom Conerly, Nick Turner, Cem Anil, Carson Denison, Amanda Askell, et al. Towards monosemanticity: Decomposing language models with dictionary learning. *Transformer Circuits Thread*, 2023.

Yaohui Cai, Zhewei Yao, Zhen Dong, Amir Gholami, Michael W Mahoney, and Kurt Keutzer. Zeroq: A novel zero shot quantization framework. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 13169–13178, 2020.

Eduardo F Camacho and Carlos Alba Bordons. *Model predictive control*. Springer Science & Business Media, 2013.

Elena Casto and Evelina Fedorenko. Cerebellar language hubs: Functional segregation of linguistic processing from motor and cognitive domains. *Nature Neuroscience*, 29(2):145–158, 2026.

Youngmin Cho and Lawrence Saul. Kernel methods for deep learning. In Y. Bengio, D. Schuurmans, J. Lafferty, C. Williams, and A. Culotta (eds.), *Advances in Neural Information Processing Systems*, volume 22. Curran Associates, Inc., 2009. URL https://proceedings.neurips.cc/paper_files/paper/2009/file/5751ec3e9a4feab575962e78e006250d-Paper.pdf.

Yatin Dandi, Luca Pesce, Lenka Zdeborova, and Florent Krzakala. The computational advantage of depth in learning high-dimensional hierarchical targets. In *The Thirty-ninth Annual Conference on Neural Information Processing Systems*, 2025.

George B Dantzig. Discrete-variable extremum problems. *Operations research*, 5(2):266–288, 1957.

Grégoire Delétang, Anian Ruoss, Paul-Ambroise Duquenne, Elliot Catt, Tim Genewein, Christopher Mattern, Jordi Grau-Moya, Li Kevin Wenliang, Matthew Aitchison, Laurent Orseau, Marcus Hutter, and Joel Veness. Language modeling is compression. *arXiv*, 2023. doi: 10.48550/arxiv.2309.10668.

Persi Diaconis and David Freedman. Asymptotics of graphical projections. *The Annals of Statistics*, pp. 793–815, 1984.

Laurent Dinh, Razvan Pascanu, Samy Bengio, and Yoshua Bengio. Sharp minima can generalize for deep nets. In *International Conference on Machine Learning*, pp. 1019–1028. PMLR, 2017.

Houssam El Cheairi, David Gamarnik, and Rahul Mazumder. Theoretical compression bounds for wide multilayer perceptrons. In *39th Annual Conference on Learning Theory*, pp. 1–59. Proceedings of Machine Learning Research, 2026.

Joshua Engels, Isaac Liao, Eric J. Michaud, Wes Gurnee, and Max Tegmark. Not all language model features are linear. *arXiv preprint arXiv:2405.14860*, 2024.

Rahim Entezari, Hanie Sedghi, Olga Saund, and Behnam Neyshabur. The role of permutation invariance in linear mode connectivity of neural networks. In *International Conference on Learning Representations*, 2022.

Angela Fan, Edouard Grave, and Armand Joulin. Reducing transformer depth on demand with structured dropout. In *International Conference on Learning Representations (ICLR)*, 2020.

Jonathan Frankle and Michael Carbin. The lottery ticket hypothesis: Finding sparse, trainable neural networks. In *International Conference on Learning Representations*, 2019.

Jonathan Frankle, David J Schwab, and Ari S Morcos. Training batchnorm and only batchnorm: On the expressive power of random features in cnns. In *International Conference on Learning Representations*, 2021.

Michael R Garey and David S Johnson. *Computers and intractability: A guide to the theory of NP-completeness*. W. H. Freeman and Company, 1979.

Tim Genewein, Jordi Grau-Moya, Li Kevin Wenliang, Laurent Orseau, and Marcus Hutter. Algorithmic compression via pretrained neural networks. *Entropy*, 28:596, 2026. doi: 10.3390/e28060596. URL https://doi.org/10.3390/e28060596.

Priya Goyal, Piotr Dollár, Ross Girshick, Pieter Noordhuis, Lukasz Wesolowski, Aapo Kyrola, Andrew Tulloch, Yangqing Jia, and Kaiming He. Accurate, large minibatch sgd: Training imagenet in 1 hour. *arXiv preprint arXiv:1706.02677*, 2017.

Stephen Grossberg. Competitive learning: From interactive activation to adaptive resonance. *Cognitive Science*, 11(1):23–63, 1987.

Song Han, Jeff Pool, John Tran, and William Dally. Learning both weights and connections for efficient neural network. In *Advances in neural information processing systems*, volume 28, 2015.

Boris Hanin and David Rolnick. How to start training: The effect of initialization and architecture. In *Advances in Neural Information Processing Systems*, pp. 571–581, 2018.

Babak Hassibi and David G Storkey. Second order derivatives for network pruning: Optimal brain surgeon. In *Advances in Neural Information Processing Systems*, volume 5, 1992.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Identity mappings in deep residual networks. In *European conference on computer vision (ECCV)*, pp. 630–645. Springer, 2016.

Shwai He, Guoheng Sun, Haichao Zhang, Yun Fu, and Ang Li. Demystifying when pruning works via representation hierarchies, 2026. URL https://arxiv.org/abs/2603.24652.

Yihui He, Xiangyu Zhang, and Jian Sun. Channel pruning for accelerating very deep neural networks. In *Proceedings of the IEEE international conference on computer vision*, pp. 1389–1397, 2017.

Geoffrey E Hinton and Drew Van Camp. Keeping the neural networks simple by minimizing the description length of the weights. In *Proceedings of the Sixth Annual Conference on Computational Learning Theory*, pp. 5–13, 1993.

Sara Hooker, Aaron Courville, Gregory Clark, Yannis Yannakakis, and Kevin Murphy. What do compressed deep neural networks forget? *arXiv preprint arXiv:1911.05248*, 2019.

Neil Houlsby, Andrei Giurgiu, Stanislaw Jastrzebski, Bruna Morrone, Quentin De Laroussilhe, Andrea Gesmundo, Mona Attariyan, and Sylvain Gelly. Parameter-efficient transfer learning for nlp. In *International Conference on Machine Learning*, pp. 2790–2799. PMLR, 2019.

Edward J. Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. LoRA: Low-rank adaptation of large language models. In *International Conference on Learning Representations (ICLR)*, 2022.

Gao Huang, Yu Sun, Zhuang Liu, Daniel Sedra, and Kilian Q Weinberger. Deep networks with stochastic depth. In *European Conference on Computer Vision (ECCV)*, pp. 646–661. Springer, 2016.

HuggingFace Research Team, Leandro von Werra, and Sourab Mangrulkar. Orthogonal subspace fine-tuning (osf): A unified framework for interference-free peft. *arXiv preprint arXiv:2602.04519*, 2026.

Minyoung Huh, Brian Cheung, Tongzhou Wang, and Phillip Isola. The platonic representation hypothesis. *arXiv preprint arXiv:2405.07987*, 2024.

Kaarel Hänni, Jake Mendel, Dmitry Vaintrob, and Lawrence Chan. Mathematical models of computation in superposition. *arXiv*, 2024. doi: 10.48550/arxiv.2408.05451.

Monirul Islam, M. Anwar Ma'sum, Mahardhika Pratama, and Igor Skrjanc. Latest advancements towards catastrophic forgetting under data scarcity: A comprehensive survey on few-shot class incremental learning. *arXiv preprint arXiv:2502.08209*, 2025.

Arthur Jacot, Franck Gabriel, and Clément Hongler. Neural tangent kernel: Convergence and generalization in neural networks. In *Advances in neural information processing systems*, volume 31, 2018.

Herbert Jaeger. Controlling recurrent neural networks by conceptors. *arXiv*, 2014. doi: 10.48550/arxiv.1403.3369.

Edwin T Jaynes. Information theory and statistical mechanics. *Physical review*, 106(4):620, 1957.

Richard M Karp. Reducibility among combinatorial problems. In *Complexity of computer computations*, pp. 85–103. Springer, 1972.

Daniel L Kimmel, Kimberly L Stachenfeld, CD Salzman, and Daphna Shohamy. Neural representations supporting generalization under continual learning. *bioRxiv*, 2026. doi: 10.1101/2026.01.07.698044. URL https://www.biorxiv.org/content/10.64898/2026.01.07.698044.

James Kirkpatrick, Razvan Pascanu, Neil Rabinowitz, Joel Veness, Guillaume Desjardins, Andrei A Rusu, Kieran Milan, John Quan, Tiago Ramalho, Agnieszka Grabska-Barwinska, et al. Overcoming catastrophic forgetting in neural networks. *Proceedings of the national academy of sciences*, 114 (13):3521–3526, 2017.

Lingjing Kong, Xin Liu, Guangyi Chen, Martin Q. Ma, Xiangchen Song, Yuekai Sun, Mikhail Yurochkin, Taylor W. Killian, Ruslan Salakhutdinov, Kun Zhang, Eric P. Xing, and Zhengzhong Liu. From reasoning traces to reusable modules: Understanding compositional generalization in language model reasoning, 2026. URL https://arxiv.org/abs/2606.18089.

Alex Krizhevsky. Learning multiple layers of features from tiny images. pp. 32–33, 2009. URL https://www.cs.toronto.edu/~kriz/learning-features-2009-TR.pdf.

Dharshan Kumaran, Demis Hassabis, and James L McClelland. What learning systems do intelligent agents need? complementary learning systems theory updated. *Trends in Cognitive Sciences*, 20 (7):512–534, 2016.

Yann LeCun, John S. Denker, and Sara Solla. Optimal brain damage. In *Advances in Neural Information Processing Systems (NIPS)*, 1989.

Jaehoon Lee, Yasaman Bahri, Roman Novak, Samuel S Schoenholz, Jeffrey Pennington, and Jascha Sohl-Dickstein. Deep neural networks as gaussian processes. In *International Conference on Learning Representations*, 2018.

Namhoon Lee, Thalaiyasingam Ajanthan, Joost van de Weijer, P. H. S. Chang, and Philip Torr. Layer-adaptive sparsity for the magnitude-based pruning. In *International Conference on Learning Representations*, 2021. URL https://openreview.net/forum?id=H6ATjJ0TKdf.

Hao Li, Asim Kadav, Igor Durdanovic, Hanan Samet, and Hans Peter Graf. Pruning filters for efficient convnets. In *International Conference on Learning Representations (ICLR) Workshop*, 2017.

Hanxiao Liu, Karen Simonyan, and Yiming Yang. Darts: Differentiable architecture search. In *International Conference on Learning Representations (ICLR)*, 2019.

Zhuang Liu, Jianguo Li, Zhiqiang Shen, Gao Huang, Shoumeng Yan, and Changshui Zhang. Learning efficient convolutional networks through network slimming. In *Proceedings of the IEEE international conference on computer vision*, pp. 2736–2744, 2017.

Raphael G Lopes, Stefano Fenu, and Thad Starner. Data-free knowledge distillation for deep neural networks. In *NIPS Workshop on Machine Learning on the Phone and other Consumer Devices*, 2017.

Jian-Hao Luo, Jianxin Wu, and Weiyao Lin. Thinet: A filter level pruning method for deep neural network compression. In *Proceedings of the IEEE International Conference on Computer Vision*, pp. 5058–5066, 2017.

Xinyin Ma, Gongfan Fang, and Xinchao Wang. Llm-pruner: On the structural pruning of large language models. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2023. URL https://arxiv.org/abs/2305.11627.

Arun Mallya and Svetlana Lazebnik. Packnet: Adding multiple tasks to a single network by iterative pruning. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 7765–7773, 2018.

Charles H Martin and Michael W Mahoney. Implicit self-regularization in deep neural networks: Evidence from random matrix theory and implications for learning. *Journal of Machine Learning Research*, 22(165):1–73, 2021. First presented/arXiv in 2018.

James L McClelland, Bruce L McNaughton, and Randall C O'Reilly. Why there are complementary learning systems in the hippocampus and neocortex: insights from the successes and failures of connectionist models of learning and memory. *Psychological Review*, 102(3):419–457, 1995.

Paul Micaelli and Amos J Storkey. Zero-shot knowledge transfer via adversarial belief matching. In *Advances in Neural Information Processing Systems*, volume 32, 2019.

Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. Efficient estimation of word representations in vector space. In *1st International Conference on Learning Representations, ICLR 2013*, 2013.

Pavlo Molchanov, Stephen Tyagi, Apostol Natsev, and Jonathan Krause. Pruning convolutional neural networks for resource efficient inference. In *International Conference on Learning Representations*, 2017.

Pavlo Molchanov, Arun Mallya, Stephen Tyagi, Iuri Bourezak, and Jan Kautz. Importance estimation for neural network pruning. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 11264–11272, 2019.

Ari S Morcos, David GT Barrett, Neil C Rabinowitz, and Matthew Botvinick. On the importance of single directions for generalization. In *International Conference on Learning Representations*, 2018.

Gabriel Moreira, Zita Marinho, Manuel Marques, João Paulo Costeira, and Chenyan Xiong. Native hierarchical and compositional representations with subspace embeddings. In *Proceedings of the 32nd ACM SIGKDD Conference on Knowledge Discovery and Data Mining (KDD '26)*, 2026. doi: 10.1145/3770855.3817808.

Michael C Mozer and Paul Smolensky. Skeletonization: A technique for trimming the fat from a network via relevance assessment. In D. Touretzky (ed.), *Advances in Neural Information Processing Systems*, volume 1. Morgan-Kaufmann, 1988.

Markus Nagel, Mart van Baalen, Tijmen Blankevoort, and Max Welling. Data-free quantization through weight equalization and bias correction. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 1325–1334, 2019.

Radford M Neal. *Bayesian learning for neural networks*, volume 118. Springer Science & Business Media, 1996.

Yuval Netzer, Tao Wang, Adam Coates, Alessandro Bissacco, Bo Wu, and Andrew Y. Ng. Reading digits in natural images with unsupervised feature learning. In *NIPS Workshop on Deep Learning and Unsupervised Feature Learning*, 2011.

Behnam Neyshabur, Ryota Tomioka, and Nathan Srebro. In search of the real inductive bias: On the role of implicit regularization in deep learning. In *International Conference on Learning Representations*, 2015.

Behnam Neyshabur, Srinadh Bhojanapalli, David McAllester, and Nati Srebro. Exploring generalization in deep learning. In *Advances in neural information processing systems*, volume 30, 2017.

Quang Nguyen, Hai Ha Pham, Davide Cassi, and Michele Bellingeri. Depth fragility and skeletal universality: Decoupling topology and function in deep neural networks. *Mathematics*, 14(9):1438, 2026. doi: 10.3390/math14091438. URL https://www.mdpi.com/2227-7390/14/9/1438.

Chris Olah, Nick Cammarata, Ludwig Schubert, Gabriel Goh, Michael Petrov, and Shan Carter. Zoom in: An introduction to circuits. *Distill*, 2020. doi: 10.23915/distill.00024.001. URL https://distill.pub/2020/circuits/zoom-in.

Kiho Park, Yo Joong Choe, and Victor Veitch. The linear representation hypothesis and the geometry of large language models. *arXiv preprint arXiv:2311.03658*, 2023.

Jeffrey Pennington, Richard Socher, and Christopher D Manning. Glove: Global vectors for word representation. In *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*, pp. 1532–1543, 2014.

Wei Qiu, Ming Yin, Mengdi Wang, Peter Bartlett, and Andrea Zanette. Continual learning in the frequency domain. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2024.

Alex Renda, Jonathan Frankle, and Michael Carbin. Comparing rewinding and fine-tuning in neural network pruning. In *International Conference on Learning Representations*, 2020.

Jorma Rissanen. Modeling by shortest data description. *Automatica*, 14(5):465–471, 1978.

Gobinda Saha, Isha Garg, and Kaushik Roy. Gradient projection memory for continual learning. In *International Conference on Learning Representations (ICLR)*, 2021.

Simone Scardapane, Danilo Comminiello, Amir Hussain, and Aurelio Uncini. Group sparse regularization for deep neural networks. *Neurocomputing*, 241:81–89, 2017.

Carolin Scholl, Michael E. Rule, and Matthias H. Hennig. The information theory of developmental pruning: Optimizing global network architectures using local synaptic rules. *PLOS Computational Biology*, 17(10):1–23, 10 2021. doi: 10.1371/journal.pcbi.1009458. URL https://doi.org/10.1371/journal.pcbi.1009458.

Ravid Shwartz-Ziv and Naftali Tishby. Opening the black box of deep neural networks via information. *arXiv preprint arXiv:1703.00810*, 2017.

Sidak Pal Singh and Dan Alistarh. Woodfisher: Efficient second-order approximation for neural network compression. In *Advances in Neural Information Processing Systems*, volume 33, pp. 18098–18109, 2020.

Sidak Pal Singh and Martin Jaggi. Model fusion via optimal transport. In *Advances in Neural Information Processing Systems*, volume 33, pp. 22045–22055, 2020.

Suraj Srinivas and R. Venkatesh Babu. Data-free parameter pruning for deep neural networks. *arXiv*, 2015. doi: 10.48550/arxiv.1507.06149.

George Stoica, Daniel Bolya, Jakob Bales, and Judy Hoffman. Zipit! merging models from different tasks without training. In *International Conference on Learning Representations*, 2024.

Hidenori Tanaka, Daniel Kunin, Daniel L. K. Yamins, and Surya Ganguli. Pruning neural networks without any data by iteratively conserving synaptic flow. *ArXiv*, abs/2006.05467, 2020. URL `https://api.semanticscholar.org/CorpusID:219558821`.

Norman Tatro, Pin-Yu Chen, Payel Das, Prasanna Sattigeri, Renzhi Lai, and Zihang Huan. Optimizing mode connectivity via neuron alignment. In *Advances in Neural Information Processing Systems*, volume 33, pp. 15300–15311, 2020.

Naftali Tishby, Fernando C Pereira, and William Bialek. The information bottleneck method. In *The 37th Annual Allerton Conference on Communication, Control, and Computing*, pp. 368–377, 1999.

Andreas Veit and Serge Belongie. Convolutional networks with adaptive inference graphs. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pp. 3–18, 2018.

Peng Wang, Xiao Li Li, Can Yaras, Zhihui Zhu, Laura Balzano, Wei Hu, and Qing Qu. Understanding deep representation learning via layerwise feature compression and discrimination. *Journal of Machine Learning Research*, 26(47):1–71, 2025. URL `http://jmlr.org`.

Shipeng Wang, Xiaorong Li, Jian Sun, and Zongben Xu. Training networks in null space of feature covariance for continual learning. *2021 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 184–193, 2021. doi: 10.1109/cvpr46437.2021.00025.

Xin Wang, Fisher Yu, Zi-Yi Dou, Trevor Darrell, and Joseph E Gonzalez. Skipnet: Learning dynamic routing in convolutional networks. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pp. 409–424, 2018.

Wei Wen, Chunpeng Wu, Yandan Wang, Yiran Chen, and Hai Li. Learning structured sparsity in deep neural networks. In *Advances in Neural Information Processing Systems (NIPS)*, 2016.

Jingyi Xie, Jintao Yang, Zhunchen Luo, Yunbo Cao, Qiang Gao, Mengyuan Zhang, and Wenpeng Hu. AdaDARE-γ: Balancing stability and plasticity in multi-modal LLMs through efficient adaptation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2025.

Greg Yang. Tensor programs i: Wide feedforward or recurrent neural networks of any architecture are gaussian processes. In *Advances in Neural Information Processing Systems*, volume 32, 2019.

Min Yang, Wei Zhang, and Hao Liu. Lora-null: Directing parameter-efficient adaptation into orthogonal null-spaces. In *Proceedings of the International Conference on Machine Learning (ICML)*, 2025.

Hongxu Yin, Pavlo Molchanov, Jose M Alvarez, Zhizhong Li, Arun Mallya, Derek Hoiem, Niraj K Jha, and Jan Kautz. Dreaming to distill: Data-free knowledge transfer via deepinversion. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 8715–8724, 2020.

Jason Yosinski, Jeff Clune, Yoshua Bengio, and Hod Lipson. How transferable are features in deep neural networks? In *Advances in neural information processing systems*, volume 27, 2014.

Edouard Yvinec, Arnaud Dapogny, Matthieu Cord, and Kevin Bailly. Red : Looking for redundancies for data-free structured compression of deep neural networks. *arXiv*, 2021. doi: 10.48550/arxiv.2105.14797.

Guanxiong Zeng, Yang Chen, Bo Cui, and Shan Yu. Continual learning of context-dependent processing in neural networks. *Nature Machine Intelligence*, 1(8):364–372, 2019.

Friedemann Zenke, Ben Poole, and Surya Ganguli. Continual learning through synaptic intelligence. In *International conference on machine learning*, pp. 3987–3995. PMLR, 2017.

Yedi Zhang, Andrew Saxe, and Peter E. Latham. Saddle-to-saddle dynamics explains a simplicity bias across neural network architectures. *arXiv preprint arXiv:2512.20607*, 2025. URL https://arxiv.org/abs/2512.20607.

Xunlan Zhou, Hongrui Zhao, and Negar Mehr. Taco: Temporal consensus optimization for continual neural mapping. *arXiv preprint arXiv:2602.04516*, 2026.

Barret Zoph and Quoc V Le. Neural architecture search with reinforcement learning. In *International Conference on Learning Representations*, 2017.

APPENDIX TABLE OF CONTENTS

A Hilbert Spaces **20**

A.1 Introduction: Why a Hilbert Space? 20

A.2 The Inner Product: The Ruler of Geometry 20

A.3 The Ambient Space: $L_2(\mathcal{X}, P_{\mathcal{X}})$ 20

A.4 The Tensor Product: Splitting Continuous and Discrete Spaces 21

A.5 Uniqueness and the Total Set Property 21

A.6 Neuron Synthesis by Projection 22

A.7 Summary 22

B Implementation Notes **22**

B.1 Adaptation for Convolutional Layers 22

B.2 Deriving the Parameter Footprint ΔP 23

B.3 Cross-Action Overlap and Uniform Scaling 24

B.4 Computational Complexity and the Decoupled Cache 24

B.5 Numerical Stability and BN Parameters 25

C Main Paper Proofs **26**

C.1 Layer Transition Costs 26

C.2 Generating Parent Neuron 31

C.3 Block Eviction 33

D Layer Transition Cost Derivation **34**

D.1 Continuous-Time Relaxation 34

D.2 Bounding the Projection Cost 34

E Generating the Parent Neuron **35**

E.1 The Parent Neuron in Hilbert Space 35

E.2 From Hilbert Space to Physical Parameters 37

F Derivation of Physical BN Parameters **38**

G Kernel Formulation **39**

G.1 Pre-Activation Distribution 39

G.2 Self-Kernel 40

G.3 Cross-Kernel 41

H Derivations for Block Eviction **43**

H.1 Generalization and Execution Degradation in Depleted Blocks 43

H.2 Derivation of the Unified Macro Cost $\mathcal{J}_{\text{evict}}$ 43

H.3 Generalization to Non-Residual Architectures 45

I Reproducibility Protocols for Cross-Domain Transfer **46**

I.1 Task Construction and Data Partitioning 46

I.2 Network Architecture 46

I.3 Base Training Regimen 47

I.4 EWC Exact Empirical Fisher Calculation 47

I.5 Hyperparameter Tuning and Final Evaluation 47

J Theoretical Guarantees of DEFT **48**

J.1 Algorithmic Axioms and Partitioning of Neurons 49

J.2 Layer-to-Layer Bounding Framework 50

J.3 Dynamic Resolution of Redundancy via Bounded Trade-off 52

K Algorithms **53**

A HILBERT SPACES

This section provides a quick introduction to Hilbert spaces, focusing on the concepts of inner products, completeness, and total sets. We bridge abstract functional analysis with the specific architectural choices of the HOPE framework, demonstrating how the L_2 embedding of neural functions creates a unique and complete geometric environment for optimization over network structures.

A.1 INTRODUCTION: WHY A HILBERT SPACE?

We treat compression operations like pruning and merging as *projections*. When we replace two neurons with one, we are attempting to find a single element that "represents" a multi-dimensional subspace. To perform this operation rigorously, we need three things:

1. A **Space** that contains all possible neural identities.
2. A **Metric** (Inner Product) to measure "closeness" and "alignment."
3. **Completeness** to ensure that our optimizations actually have solutions.

A Hilbert space $\mathcal{H}$ provides these three pillars.

A.2 THE INNER PRODUCT: THE RULER OF GEOMETRY

The defining feature of a Hilbert space is the **inner product**. While a vector space only lets us add and scale elements, an inner product space lets us talk about *angles* and *lengths*.

Definition A.1 (Inner Product Axioms)**.** An inner product on a vector space V over $\mathbb{R}$ is a mapping $\langle \cdot, \cdot \rangle : V \times V \to \mathbb{R}$ satisfying for all $f, g, h \in V$ and $a \in \mathbb{R}$:

- **Symmetry:** $\langle f, g \rangle = \langle g, f \rangle$.
- **Linearity:** $\langle af + g, h \rangle = a \langle f, h \rangle + \langle g, h \rangle$.
- **Positive Definiteness:** $\langle f, f \rangle \geq 0$, and $\langle f, f \rangle = 0 \iff f = 0$.

A.2.1 CONNECTION TO HOPE: THE EXPECTATION METRIC

In the HOPE framework, we operate on functions $f : \mathcal{X} \to \mathbb{R}^c$. We define our inner product relative to a surrogate distribution $P_{\mathcal{X}}$:

$$\langle f_i, f_j \rangle_{\mathcal{H}} \triangleq \mathbb{E}_{\boldsymbol{x} \sim P_{\mathcal{X}}} \left[f_i(\boldsymbol{x})^T f_j(\boldsymbol{x}) \right] \tag{6}$$

Proposition A.1 (Validity of the HOPE Metric)**.** The functional defined in (6) satisfies the inner product axioms.

Proof. Linearity and symmetry follow directly from the linearity of the expectation operator $\mathbb{E}$ and the symmetry of the Euclidean dot product. Positive definiteness is guaranteed because $\langle f, f \rangle = \mathbb{E}[\|f(\boldsymbol{x})\|^2] \geq 0$. The definiteness $\langle f, f \rangle = 0 \implies f = 0$ is satisfied in the L_2 sense (i.e., f is zero "almost everywhere"). □

A.3 THE AMBIENT SPACE: $L_2(\mathcal{X}, P_{\mathcal{X}})$

A Hilbert space is more than just an inner product space; it must be **complete**. In finite dimensions (like $\mathbb{R}^n$), every inner product space is complete. In function spaces, this is not true.

Definition A.2 (Completeness)**.** A space is complete if every Cauchy sequence $\{f_n\}$ (a sequence where elements get arbitrarily close to each other) converges to an element f that is also **inside** the space.

If we worked only with continuous functions, the space would not be complete. For example, a sequence of continuous functions can converge to a step function (which is discontinuous). This would be a disaster for compression, as our "best parent" might not even exist in our space.

A.3.1 THE L_2 EMBEDDING

To avoid this, HOPE embeds neurons into $L_2(\mathcal{X}, P_\mathcal{X})$, the space of square-integrable functions.

- **Energy Bound:** Every function in $\mathcal{H}$ has finite energy: $\mathbb{E}[\|f\|^2] < \infty$.
- **Closure:** By definition, L_2 is complete. Every optimization we perform (minimizing the cost functional of compression operations) is guaranteed to have a valid result within the ambient space.

A.4 THE TENSOR PRODUCT: SPLITTING CONTINUOUS AND DISCRETE SPACES

While we defined our ambient space as vector-valued functions $\mathcal{H} = L_2(\mathcal{X}, P_\mathcal{X}; \mathbb{R}^c)$, computing the inner product directly in this monolithic space obscures the internal structure of a neural network. HOPE simplifies this by utilizing a **tensor product space**.

A neuron's operation naturally splits into two phases:

1. **The Continuous Input Landscape:** The effective input weights (which absorb BN statistics) and the activation function create a continuous scalar landscape $g_i(\boldsymbol{x}) = \Psi((\boldsymbol{w}_{\text{in},i}^{\text{eff}})^T \boldsymbol{x} + b_i)$. We embed this function into a scalar Hilbert space $\mathcal{H}_{\text{in}} \triangleq L_2(\mathcal{X}, P_\mathcal{X}; \mathbb{R})$.
2. **The Discrete Output:** This scalar activation is broadcast to the next layer along a finite-dimensional output weight vector $\boldsymbol{w}_{\text{out},i}$. We define this output space as $\mathcal{H}_{\text{out}} \triangleq \mathbb{R}^c$.

By taking the tensor product of these two spaces, we construct the full ambient space mapping: $\mathcal{H} \cong \mathcal{H}_{\text{in}} \otimes \mathcal{H}_{\text{out}}$. Under this formulation, each individual neuron is modeled as a **rank-1 Hilbert-Schmidt operator**, represented by the outer product of its input function and output vector:

$$f_i \triangleq g_i \otimes \boldsymbol{w}_{\text{out},i} \tag{7}$$

A.4.1 FACTORING THE METRIC

This tensor structure is what makes HOPE computationally tractable. The inner product of two rank-1 tensors elegantly factors into the product of their individual space inner products:

$$\langle f_i, f_j \rangle_\mathcal{H} = \langle g_i \otimes \boldsymbol{w}_{\text{out},i}, g_j \otimes \boldsymbol{w}_{\text{out},j} \rangle_\mathcal{H} = \langle g_i, g_j \rangle_{\mathcal{H}_{\text{in}}} \cdot \langle \boldsymbol{w}_{\text{out},i}, \boldsymbol{w}_{\text{out},j} \rangle_{\mathcal{H}_{\text{out}}} \tag{8}$$

Because $\langle g_i, g_j \rangle_{\mathcal{H}_{\text{in}}}$ is the expected alignment of their non-linear activations over the distribution $P_\mathcal{X}$, we define this as the kernel $K(i, j)$. This allows us to separate the continuous-functional evaluation from the discrete parameters, reducing the full Hilbert space inner product to:

$$\langle f_i, f_j \rangle_\mathcal{H} = K(i, j) \cdot \langle \boldsymbol{w}_{\text{out},i}, \boldsymbol{w}_{\text{out},j} \rangle_{\mathbb{R}^c} \tag{9}$$

A.5 UNIQUENESS AND THE TOTAL SET PROPERTY

One might ask: "We defined the inner product for any functions f, g. But in HOPE, we only ever calculate it for single ReLU neurons. Is that enough to define the whole space?"

This is the most critical part of the theory.

Definition A.3 (Total Set)**.** A set $S \subset \mathcal{H}$ is a **total set** if the set of all finite linear combinations of elements in S is **dense** in $\mathcal{H}$.

If S is total, then knowing the inner product for every pair in S uniquely determines the inner product for the **entire** Hilbert space.

A.5.1 The Universal Approximation Theorem

In the context of HOPE, our "dictionary" of functions is the set of single neurons $\mathcal{N}$:

$$\mathcal{N} = \{f(\boldsymbol{x}) = \boldsymbol{w}_{out}\Psi(\tilde{\boldsymbol{w}}_{\text{in}} \cdot \tilde{\boldsymbol{x}})\} \tag{10}$$

Theorem A.2 (HOPE Uniqueness)**.** The set $\mathcal{N}$ is a total set in $L_2(\mathcal{X}, P_{\mathcal{X}})$.

Discussion. By the Universal Approximation Theorem, linear combinations of ReLU neurons can approximate any square-integrable function to arbitrary precision. In the language of Hilbert spaces, this means $span(\mathcal{N})$ is dense in $\mathcal{H}$.

Because $\mathcal{N}$ is total, the definition of the **function correlation kernel** $K(i, j) = \mathbb{E}[\Psi(y_i)\Psi(y_j)]$ is sufficient to uniquely characterize the metric of the entire ambient space. We do not need a "separate" definition for the inner product of sums of neurons; it is uniquely forced upon the space by the behavior of the single neurons. □

A.6 Neuron Synthesis by Projection

Finally, we see why this matters. The generation process relies on finding an **Optimal Subspace Projection**. When we want to merge neurons i and j, their joint contribution $[f_i, f_j]$ spans a 2-dimensional tensor subspace. Because a physical parent neuron must produce a single unified output, it is modeled as a constrained rank-1 approximation $[f_p, f_p]$.

Rather than a simple orthogonal projection of a sum, HOPE finds the optimal parent f_p^* by minimizing the expected Frobenius projection error under the Hilbert-Schmidt norm, scaled by the remaining capacity of the layer:

$$f_p^* = \text{argmin}_{f_p \in \mathcal{N}} \frac{\sqrt{\|f_i - f_p\|_{\mathcal{H}}^2 + \|f_j - f_p\|_{\mathcal{H}}^2}}{E_a - \|f_i\|_{\mathcal{H}} - \|f_j\|_{\mathcal{H}} + \|f_p\|_{\mathcal{H}}} \tag{11}$$

In a Hilbert space, the numerator translates to a rigorous geometric projection distance. Without the Hilbert space structure (inner products and completeness), the concepts of "closest operator" and "functional alignment" would have no meaning.

A.7 Summary

A summary of the key concepts and their utility in HOPE are provided in Table 2.

Hilbert Concept	**HOPE Implementation**
Inner Product $\langle f, g \rangle$	Expected dot product over $P_{\mathcal{X}}$
Norm $\|f\|$	Functional Capacity (Square root of Signal Energy)
Ambient Space $\mathcal{H}$	Tensor product space $\mathcal{H}_{\text{in}} \otimes \mathcal{H}_{\text{out}}$
Operators	Neurons modeled as rank-1 Hilbert-Schmidt operators
Basis / Total Set $\mathcal{N}$	The manifold of realizable single neurons
Completeness	Guarantees that the "best parent" is well-defined

Table 2: Mapping of Abstract Hilbert Concepts to the HOPE Framework.

B Implementation Notes

B.1 Adaptation for Convolutional Layers

Our formalism extends to convolutional networks by defining a "neuron" as a filter in layer A producing feature map i. The joint parameter vector $\boldsymbol{w}_i = [\boldsymbol{w}_n^\top, \boldsymbol{w}_c^\top]^\top$ is constructed by vectorizing the filter's respective input and output kernels. The input space vector $\boldsymbol{w}_n$ encapsulates the filter's

local receptive field. Assuming layer A possesses a weight tensor $\boldsymbol{K}_A \in \mathbb{R}^{h_A \times w_A \times C_{\text{in}} \times C_{\text{out}}}$, the vector $\boldsymbol{w}_n$ corresponding to filter i is the flattened spatial slice $\boldsymbol{K}_A[:, :, :, i] \in \mathbb{R}^n$, where $n = h_A \times w_A \times C_{\text{in}}$. Conversely, the output space vector $\boldsymbol{w}_c$ captures the filter's downstream influence on the subsequent layer B. Because the activation map of filter i acts as the i-th input channel to layer B, $\boldsymbol{w}_c$ is formed by extracting and flattening the corresponding slice of the downstream tensor $\boldsymbol{K}_B[:, :, i, :] \in \mathbb{R}^c$, where $c = h_B \times w_B \times C_{\text{out},B}$.

To allow the surrogate distribution to serve as a location-invariant prior without requiring intractable coordinate-specific covariance modeling, we assume spatial stationarity (ergodicity) across the feature map and construct it using the globally averaged BN variance σ_i^2. While boundary zero-padding breaks local stationarity, the high-dimensional spatial aggregation of modern BN buffers absorbs these edge-effects into a into a single global average.

B.2 Deriving the Parameter Footprint ΔP

As established in Section 8, the parameter footprint ΔP quantifies the number of physical parameters removed by a compression action. To preserve Dantzig's Axiom of Item Independence, this criterion uses a static surrogate, ΔP^{init}, evaluated on the initial network state and decoupled from the dynamically changing network. For any operation, the footprint tracks the total removed parameters:

$$\Delta P^{\text{init}} = \|W_{\text{in}}\|_0 + \|W_{\text{out}}\|_0 + \|\theta_{\text{aux}}\|_0 \tag{12}$$

where W_{in}, W_{out}, and $\boldsymbol{\theta}_{\text{aux}}$ denote the input weights, output weights, and auxiliary parameters (e.g., BN parameters) respectively, and $\|\cdot\|_0$ counts the number of non-zero elements.

B.2.1 Granular Operations

For pruning or merging, ΔP^{init} comprises the incoming weights, outgoing weights, and BN parameters of a single target neuron or filter. For example:

- **Transformer MLP:** Removing a neuron in an MLP with a $d_{\text{model}} \to 4d_{\text{model}}$ expansion yields $\Delta P^{\text{init}} = 2d_{\text{model}} + 1$.
- **Convolutional Networks:** For a filter with spatial dimensions $H \times W$, the footprint scales with the receptive field and projective kernel: $\Delta P^{\text{init}} = H \cdot W \cdot C_{\text{in}} + C_{\text{out}} + 4$, where 4 accounts for the BN parameters $\gamma, \beta, \mu, \sigma^2$.

Example: Architectural Symmetry in ResNet-50

Evaluating ΔP^{init} locally reveals symmetries across different layer types. Consider a ResNet-50 bottleneck block with a base channel width N and a $4\times$ expansion ratio:

- **1×1 Squeeze Layer:** Removing one filter deletes $4N$ input connections and $9N$ output connections to the subsequent 3×3 layer. Yield: $\Delta P^{\text{init}} = 13N$.
- **3×3 Spatial Layer:** Removing one filter deletes $9N$ input connections and $4N$ output connections to the subsequent 1×1 expansion layer. Yield: $\Delta P^{\text{init}} = 13N$.

Despite differing spatial tensor shapes, the parameter yield per filter in both positions evaluates to $13N$. This symmetry allows the cost $\mathcal{J}$ to arbitrate compression across heterogeneous layers without being biased by raw parameter array shapes.

B.2.2 Macro Operations: Block Eviction

Unlike granular operations that yield incremental savings, block eviction removes entire layers simultaneously. For a residual block with internal convolutional layers W_1 and W_2, and a terminal expansion layer W_3, the static parameter yield expands to:

$$\Delta P^{\text{init}}_{\text{evict}} = \|W_1\|_0 + \|W_2\|_0 + \|\theta_{\text{aux}}\|_0 \tag{13}$$

where $\|W_1\|_0$ and $\|W_2\|_0$ denote the parameters of the internal layers, and $\|\theta_{\text{aux}}\|_0$ counts the BN parameters $\mu, \sigma^2, \gamma, \beta$ across the entire block, including those of W_3. Note that the weight matrix $W_3 \in \mathbb{R}^{d_{\text{amb}} \times d_{\text{bottleneck}}}$ is *not* explicitly included in this sum. Because removing a filter in W_2 removes

its corresponding outgoing connections in W_3, the memory footprint of W_3 is naturally accounted for when evaluating W_2. Explicitly adding the ambient dimensions $\|W_3\|_0 = d_{\text{amb}} \times d_{\text{bottleneck}}$ would count the same parameters twice *within a single macro action*. This *intra-action double-counting* would inflate the parameter footprint of block eviction and artificially lower its Distortion Rate (DR), giving it an unfair advantage over granular operations.

B.3 Cross-Action Overlap and Uniform Scaling

While we avoid counting parameters twice within a *single* action (as seen with W_3), evaluating the entire decision space using a static footprint ΔP^{init} introduces an overlap *between different competing actions*. Let i and j be targeted neurons in adjacent layers l and $l+1$. To preserve Dantzig's Axiom of Item Independence, an action targeting layer l must not alter the state variables used to evaluate layer $l+1$. Consequently, their shared weight $W_{j,i}^{(l)}$ is counted independently in both evaluations: $W_{j,i}^{(l)} \in \Delta P_i^{\text{init}}$ and $W_{j,i}^{(l)} \in \Delta P_j^{\text{init}}$.

For any sequence of actions $\mathcal{S}$, this *cross-action overlap* overestimates the true number of parameters recovered $\sum_{k\in\mathcal{S}} \Delta P_k^{\text{init}} > \Delta P_{\mathcal{S}}^{\text{live}}$. This overestimation artificially lowers the computed DR compared to the live network state:

$$\text{DR}_k = \frac{\mathcal{J}_k}{\Delta P_k^{\text{init}}} < \frac{\mathcal{J}_k}{\Delta P_k^{\text{live}}} \tag{14}$$

To correct this approximation without violating item independence, HOPE relies on uniform scaling. Because this cross-action overlap applies systematically across the entire action space $\mathcal{A}$ (all granular and macro candidates interact with their neighbors), it acts as a uniform scaling factor $\alpha \geq 1$ such that $\Delta P_k^{\text{init}} \approx \alpha \Delta P_k^{\text{live}}$.

Because the distortion cost $\mathcal{J}$ is evaluated independently of the parameter counts, and α applies uniformly, the relative ordering of the distortion rates is preserved:

$$\frac{\mathcal{J}_a}{\Delta P_a^{\text{init}}} < \frac{\mathcal{J}_b}{\Delta P_b^{\text{init}}} \iff \frac{\mathcal{J}_a}{\Delta P_a^{\text{live}}} < \frac{\mathcal{J}_b}{\Delta P_b^{\text{live}}} \tag{15}$$

Since the greedy continuous knapsack solver selects $\arg\min_{k\in\mathcal{A}} \text{DR}_k$, this uniform scaling ensures that the theoretical fairness of the optimal action selection remains intact.

B.4 Computational Complexity and the Decoupled Cache

Evaluating the transition cost $\mathcal{J}$ efficiently poses a computational challenge because it is inversely coupled to the monotonically decreasing layer capacity E_{rem}. In a layer with N neurons, recomputing the non-linear weight-space geometry (e.g., Rank-2 Singular Value Decompositions) for all $\mathcal{O}(N^2)$ candidate pairs every time E_{rem} decreases requires $\mathcal{O}(N^3)$ execution time.

Conversely, caching $\mathcal{J}$ values in a standard priority queue via submodular approximations (e.g., Minoux's Lazy Update) leads to stale estimations. Because the transition cost goes to ∞ as capacity approaches 0, small capacity reductions cause the true costs to spike. A delayed queue would underestimate these costs and cause the optimizer to select sub-optimal actions and potentially collapse the layer. We resolve this bottleneck using an $\mathcal{O}(1)$ *Decoupled Cache*.

Decoupling. The computationally expensive optimal projections $\boldsymbol{u}^*, \boldsymbol{v}^*$ and the cost components ($a = \|f_i\|_{\mathcal{H}}^2 + \|f_j\|_{\mathcal{H}}^2$ and $b = \langle \psi^*, f_i + f_j \rangle_{\mathcal{H}}$, derived in Appendix E.1) depend only on weights and static BN parameters associated with that layer. Since our method does not rely on any cross-layer criterion (such as Fisher Information), we guarantee that $\boldsymbol{u}^*, \boldsymbol{v}^*, a, b$ remain independent of both the downstream architecture and the dynamic capacity E_{rem}.

The only variable dependent on E_{rem} is the optimal scalar magnitude s^*. At initialization, the framework computes $\boldsymbol{u}^*$ and $\boldsymbol{v}^*$ to evaluate and cache only the scalar constants a and b. To prevent $\mathcal{O}(N^2)$ memory exhaustion, the high-dimensional vectors $\boldsymbol{u}^*, \boldsymbol{v}^*$ are then discarded. During the greedy search, evaluating the cost of any action requires querying the cached constants and the live remaining capacity $E_{\text{rem}} = \max(E_a - \|f_i\|_{\mathcal{H}} - \|f_j\|_{\mathcal{H}}, \epsilon)$ to compute s^* and $\mathcal{J}$ analytically in $\mathcal{O}(1)$ time:

$$s^* = \frac{a + bE_{\text{rem}}}{2E_{\text{rem}} + b} \tag{16}$$

Index Determinism and JIT Generation. We sort the active neurons to evaluate undirected pairs only where $i < j$. By enforcing this index determinism, the algorithm guarantees a 100% cache hit rate and makes it viable to abandon priority queues entirely. At every step, the algorithm executes an $\mathcal{O}(1)$ global scan over all remaining pairs using the live layer capacity, maintaining mathematically perfect freshness. Once the global minimum is selected, the framework executes a Just-In-Time (JIT) generation, re-evaluating the fast Rank-2 SVD only for the single winning pair (~ 1 ms) to retrieve its $\boldsymbol{u}^*, \boldsymbol{v}^*$ for physical deployment.

Computing the BN Variance. Deploying the JIT-generated parent requires setting its BN variance γ_p^2. Under the surrogate distribution $P_{\mathcal{X}}$, γ_p^2 depends only on the warped correlation $\hat{\rho}_{ij}$. A naive approach might substitute the simple correlation of the raw weights ρ_{raw} into the variance equation. However, this incorrectly mixes parameter-space metrics with signal-space statistics. Doing so produces incorrect BN moving averages, which miscalibrates the network during the forward pass. The framework avoids this by computing the physical variance using $\hat{\rho}_{ij}$ directly retrieved from the cache. This ensures the deployed parameters maintain the correct statistical behavior without requiring empirical forward passes.

B.5 Numerical Stability and BN Parameters

Section E.2.2 derives the mapping from the parent neuron's effective parameters ($\boldsymbol{w}_{\text{in},p}^{\text{eff}}$ and b_p) to its physical network variables: the raw input weights $\boldsymbol{w}_{\text{in},p}^{\text{raw}}$ and the BN parameters $\gamma_p, \beta_p, \mu_p, \sigma_p^2$. A naive assignment of these physical variables alters the pre-activation signal and breaks the critical equivalence $y_p = (\boldsymbol{w}_{\text{in},p}^{\text{eff}})^T \boldsymbol{x} + b_p$. This mismatch miscalibrates the network and immediately degrades accuracy on the first forward pass.

To prevent this, HOPE enforces the exact analytical mapping. However, deploying these formulas in practice requires safely handling numerical bounds, such as preventing negative variance $\sigma_p^2 < 0$ and avoiding `NaN` in the BN denominator $\sqrt{\sigma_p^2 + \epsilon}$. This section details how to resolve these numerical edge cases and ensure the deployed model remains mathematically well-defined and numerically stable.

B.5.1 The ϵ Boundary Regime and Variance Clamping

To resolve the under-constrained BN system, the framework fixes the physical variance as $\sigma_p^2 = \max(0, \gamma_p^2 - \epsilon)$, where ϵ is a small stability constant (e.g., 10^{-5}). For active features $\gamma_p^2 \geq \epsilon$, this evaluates smoothly. This allows the denominator of the BN transformation to simplify to $|\gamma_p|$.

However, as the progressive encoder shrinks the network, the derived parent variance γ_p^2 may occasionally fall below the numerical floor ϵ. Without the $\max(0, \cdot)$ operator, enforcing $\sigma_p^2 = \gamma_p^2 - \epsilon$ would result in a negative variance, which causes `NaN` during inference. The clamping operator safely bounds the physical variance at $\sigma_p^2 = 0$.

When σ_p^2 is clamped to 0, the BN denominator $\sqrt{\sigma_p^2 + \epsilon}$ evaluates to $\sqrt{\epsilon}$. Consequently, the scale factor becomes $\gamma_p / \sqrt{\epsilon}$. Substituting this into the effective bias equation yields:

$$b_p = \beta_p - \frac{\gamma_p}{\sqrt{\epsilon}} \mu_p \quad \Longrightarrow \quad \mu_p = \frac{\sqrt{\epsilon}}{\gamma_p} (\beta_p - b_p) \tag{17}$$

As the parent neuron's scale γ_p approaches 0, dividing by γ_p causes $\mu_p \to \infty$. To prevent this, the framework bypasses this calculation for inactive features and sets $\boldsymbol{w}_{\text{in},p}^{\text{raw}} = \mathbf{0}$.

B.5.2 Running Variance Offset During Fine-Tuning

Setting the initial BN running variance to $\sigma_p^2 = \gamma_p^2 - \epsilon$ ensures the network's output is perfectly preserved during inference. However, when the model resumes training for fine-tuning, the actual batch variance computed during the forward pass evaluates to $\sigma_{\text{batch}}^2 = \gamma_p^2$. This introduces a minor mismatch between the empirical batch variance and the stored running variance:

$$\sigma_{\text{batch}}^2 - \sigma_p^2 = \gamma_p^2 - (\gamma_p^2 - \epsilon) = \epsilon \tag{18}$$

Because this discrepancy evaluates to the numerical constant ϵ (typically 10^{-5}), its impact is negligible. Standard optimizers (e.g., Adam, AdamW) seamlessly absorb this $\mathcal{O}(\epsilon)$ offset during the initial training steps without destabilizing the network or degrading performance.

C Main Paper Proofs

C.1 Layer Transition Costs

For convenience, we first recall the axioms of the cost $\mathcal{J}$ from the main paper.

Axioms of the Cost Functional $\mathcal{J}$

To ensure a well-posed definition of the cost $\mathcal{J}$, the framework introduces the following three axioms:

- **Magnitude Neutrality:** $\mathcal{J}$ must be scale invariant: $\forall k > 0\,;\, \mathcal{J}(k\,\Phi_a, k\,\Phi_b) = \mathcal{J}(\Phi_a, \Phi_b)$.
- **Connectivity Preservation:** $\mathcal{J}$ must establish an asymptotic barrier preventing layer extinction: $\lim_{E(\Phi_b)\to 0^+} \mathcal{J} = \infty$.
- **Infinitesimal Capacity Dependence:** $\mathcal{J}$ must be additive along continuous paths and driven by the reduction in layer capacity: $\mathcal{J}(\Phi_a, \Phi_b) = \int_0^1 -\xi(\Phi(t))\dot{E}(t)dt$, where $\dot{E}(t) \triangleq dE(\Phi(t))/dt$ and $\xi(\Phi(t)) > 0$ is a state-dependent density function.

Lemma C.1 (Uniqueness of the L_1 Capacity)**.** Let $\Phi = (f_1, \ldots, f_N) \in \mathcal{H}^N$ denote a layer state. Assume the capacity functional $E : \bigcup_{N=1}^{\infty} \mathcal{H}^N \to \mathbb{R}_{\geq 0}$ satisfies:

1. **Identity:** $\forall f \in \mathcal{H},\ E((f)) = \|f\|_{\mathcal{H}}$.
2. **Symmetry & Separability:** $\exists\, g : \mathbb{R}_{\geq 0} \to \mathbb{R}$ (continuous and strictly monotonic) and a function h such that $\forall \Phi \in \mathcal{H}^N,\ E(\Phi) = h\left(\sum_{k=1}^{N} g(\|f_k\|_{\mathcal{H}})\right)$.
3. **Partition Invariance:** $\forall f \in \mathcal{H}, \forall N \in \mathbb{Z}_{\geq 1},\ E((f)) = E\big(\underbrace{(f/N, \ldots, f/N)}_{N \text{ times}}\big)$.

Then $E(\Phi) = \sum_{k=1}^{N} \|f_k\|_{\mathcal{H}}$.

Proof. For a single-neuron state $\Phi = (f)$, Conditions 1 and 2 imply:

$$E((f)) = h\big(g(\|f\|_{\mathcal{H}})\big) = \|f\|_{\mathcal{H}} \implies h \equiv g^{-1} \text{ on } \mathrm{Im}(g)$$

Thus, the functional simplifies to $E(\Phi) = g^{-1}\left(\sum_{k=1}^{N} g(\|f_k\|_{\mathcal{H}})\right)$.

By Condition 3 and the positive homogeneity of the norm ($\|f/N\|_{\mathcal{H}} = \|f\|_{\mathcal{H}}/N$ for $N \geq 1$):

$$\|f\|_{\mathcal{H}} = E\big(\underbrace{(f/N, \ldots, f/N)}_{N \text{ times}}\big) = g^{-1}\left(\sum_{k=1}^{N} g\left(\frac{\|f\|_{\mathcal{H}}}{N}\right)\right) = g^{-1}\left(Ng\left(\frac{\|f\|_{\mathcal{H}}}{N}\right)\right)$$

Applying g to both sides and substituting $x \triangleq \|f\|_{\mathcal{H}} \geq 0$ yields:

$$g(x) = Ng\left(\frac{x}{N}\right) \qquad \forall x \geq 0,\ \forall N \in \mathbb{Z}_{\geq 1}$$

For $x = 0$, $g(0) = Ng(0) \implies g(0) = 0$. For any rational $q = M/N > 0$, substituting $y = x/N$ yields $g(My) = Mg(y) = qNg(y) = qg(Ny)$, meaning $g(qx) = qg(x)$. Since g is continuous, this linearity extends to all $x \in \mathbb{R}_{\geq 0}$, yielding $g(x) = cx$ for some constant c. Since g is strictly monotonic, $c \neq 0$.

Substituting $g(x) = cx$ and $g^{-1}(y) = y/c$ into the expression for $E(\Phi)$ gives:

$$E(\Phi) = \frac{1}{c}\sum_{k=1}^{N} c\,\|f_k\|_{\mathcal{H}} = \sum_{k=1}^{N}\|f_k\|_{\mathcal{H}}$$

□

Theorem C.2 (Integral Formulation of Scale-Invariant Cost)**.** Under the Axioms of Magnitude Neutrality, Connectivity Preservation, and Infinitesimal Capacity Dependence, the transition cost along a continuous deformation path $\Phi : [0,1] \to \mathcal{H}^N$ with boundary conditions $\Phi(0) = \Phi_a$ and $\Phi(1) = \Phi_b$, is determined as the integral:

$$\mathcal{J}(\Phi_a, \Phi_b) = \int_0^1 -c(\Phi(t))\frac{\dot{E}(t)}{E(\Phi(t))}dt^a \tag{19}$$

where $E(\Phi(t))$ is the instantaneous capacity, $\dot{E}(t) < 0$ is the rate of capacity reduction, and $c(\Phi(t)) > 0$ is a scale-invariant factor.

[a] Since the Hilbert norm $\|f\|_{\mathcal{H}}$ is non-differentiable at $f = \mathbf{0}$, this integral is defined over paths where the active capacity remains positive $E(\Phi(t)) > 0$.

Proof. **1. Differential Form:** By the Infinitesimal Capacity Dependence axiom, the differential cost along a path is driven by capacity reduction:

$$\dot{\mathcal{J}}(t) = -\xi(\Phi(t))\dot{E}(t)$$

Since active capacity decreases during compression $\dot{E}(t) < 0$ and the cost rate must be positive $\dot{\mathcal{J}}(t) > 0$, we require $\xi(\Phi(t)) > 0$.

2. Magnitude Neutrality: For any scalar $k > 0$, Magnitude Neutrality requires $\mathcal{J}(k\Phi_a, k\Phi_b) = \mathcal{J}(\Phi_a, \Phi_b)$. By Lemma C.1, capacity scales linearly $E(k\Phi) = kE(\Phi)$, so $\dot{E}(k\Phi(t)) = k\dot{E}(t)$. Integrating over the scaled path yields:

$$\int_0^1 -\xi(k\Phi(t))\,k\,\dot{E}(t)dt = \int_0^1 -\xi(\Phi(t))\dot{E}(t)dt$$

Assuming continuous integrands, since this equality holds for any valid continuous path, the integrands must be identical pointwise:

$$k\,\xi(k\Phi)\dot{E}(t) = \xi(\Phi)\dot{E}(t) \implies \xi(k\Phi) = k^{-1}\xi(\Phi)$$

Thus, ξ is a homogeneous function of degree -1.

3. Scale-Invariant Factor: Define $c(\Phi) \triangleq \xi(\Phi)E(\Phi)$. Scaling the state by k yields:

$$c(k\Phi) = \xi(k\Phi)E(k\Phi) = (k^{-1}\xi(\Phi))(kE(\Phi)) = c(\Phi)$$

This shows $c(\Phi)$ is scale-invariant. Substituting $\xi(\Phi) = c(\Phi)/E(\Phi)$ back into the differential form gives the integral:

$$\mathcal{J}(\Phi_a, \Phi_b) = \int_0^1 -c(\Phi(t))\frac{\dot{E}(t)}{E(\Phi(t))}dt$$

4. Connectivity Preservation: This axiom mandates an infinite cost barrier against layer extinction: $\lim_{E_b \to 0^+}\mathcal{J}(\Phi_a, \Phi_b) = \infty$, where $E_b = E(\Phi_b)$ and $E_a = E(\Phi_a)$. Applying the change of variables $dE = \dot{E}(t)dt$ and reversing the limits (which absorbs the negative sign since $E_b < E_a$ due to $\dot{E}(t) < 0$) gives:

$$\mathcal{J}(\Phi_a, \Phi_b) = \int_{E_a}^{E_b} -c(\Phi)\frac{dE}{E} = \int_{E_b}^{E_a} c(\Phi)\frac{dE}{E}$$

If $c(\Phi)$ is bounded below by a constant $c_{\min} > 0$ along the path to extinction:

$$\mathcal{J}(\Phi_a, \Phi_b) \geq \int_{E_b}^{E_a} c_{\min} \frac{dE}{E} = c_{\min}\big(\ln(E_a) - \ln(E_b)\big)$$

Taking the limit as $E_b \to 0^+$ yields ∞, satisfying the axiom. □

Relative Differential Cost

For any differentiable deformation path $\Phi : [0,1] \to \mathcal{H}^N$, the rate of geometric projection cost accumulation $\dot{\mathcal{J}}_{\text{proj}}(t)$ is defined as:

$$\dot{\mathcal{J}}_{\text{proj}}(t) \triangleq c(\Phi(t)) \frac{\dot{s}(t)}{E(\Phi(t))} \tag{20}$$

where $\dot{s}(t) = \left\|\dot{\Phi}(t)\right\|_{\mathcal{H}^N} \geq 0$ is the instantaneous geometric speed of the state vector.

Derivation. The scalar capacity rate $\dot{E}(t)$ from Theorem C.2 assigns zero penalty to geometric deformations (e.g., orthogonal rotations) where scalar capacity is conserved $\dot{E}(t) = 0$. To capture structural distortion, we calibrate the criterion along a localized orthogonal path.

Consider a compressive path where a single active neuron f_k is scaled toward $\mathbf{0}$ by a decreasing scalar $\alpha(t) \in [0,1]$ $\dot{\alpha}(t) < 0$, while all other neurons remain static. The capacity is $E(\Phi(t)) = \alpha(t)\,\|f_k\|_{\mathcal{H}} + \sum_{i \neq k} \|f_i\|_{\mathcal{H}}$, giving $\dot{E}(t) = \dot{\alpha}(t)\,\|f_k\|_{\mathcal{H}} < 0$.

The geometric speed $\dot{s}(t)$ is the norm of the state derivative vector. Since only the k-th coordinate changes, this vector is 1-sparse, ensuring the norms coincide:

$$\dot{s}(t) = \left\|\dot{\Phi}(t)\right\|_{\mathcal{H}^N} = |\dot{\alpha}(t)|\,\|f_k\|_{\mathcal{H}} = -\dot{\alpha}(t)\,\|f_k\|_{\mathcal{H}} = -\dot{E}(t)$$

Substituting $\dot{s}(t) = -\dot{E}(t)$ into the cost baseline $\dot{\mathcal{J}}_{\text{capacity}}(t) = -c(\Phi)\frac{\dot{E}}{E}$ yields $\dot{\mathcal{J}}(t) = c(\Phi)\frac{\dot{s}}{E}$ for this orthogonal axis. Because the ambient space $\mathcal{H}^N$ is isotropic, we define the geometric projection cost $\dot{\mathcal{J}}_{\text{proj}}(t)$ as this ratio to generalize to arbitrary trajectories. □

Lemma C.3 (Capacity Bound for Correlated Projections)**.** Let $f_i, f_j \in \mathcal{H}$ be active candidate neurons $\|f_i\|_{\mathcal{H}} > 0, \|f_j\|_{\mathcal{H}} > 0$, and let $f_p \in \mathcal{H}$ be their optimal parent neuron. Let $E_{\text{rem}} \triangleq E_a - \|f_i\|_{\mathcal{H}} - \|f_j\|_{\mathcal{H}} \geq 0$. Define the functional correlation as $\rho_{ij} \triangleq \frac{\langle f_i, f_j \rangle_{\mathcal{H}}}{\|f_i\|_{\mathcal{H}} \|f_j\|_{\mathcal{H}}}$. For the continuous straight-line deformation path $\Phi(t)$ in $\mathcal{H}^N$ connecting the initial state Φ_a to the pre-deletion target state $\tilde{\Phi}_b$, there exists a correlation threshold $\rho^*(f_i, f_j) \in (0,1)$ such that if $\rho_{ij} \geq \rho^*$, then:

$$E(\Phi(t)) \geq E(\Phi_b) \quad \forall\, t \in [0,1] \tag{21}$$

where $E(\Phi_b) = E_{\text{rem}} + \|f_p\|_{\mathcal{H}}$ is the capacity of the post-deletion terminal state $\Phi_b \in \mathcal{H}^{N-1}$.

Proof. Parameterizing the path as $\Phi(t) = (1-t)\Phi_a + t\tilde{\Phi}_b$ for $t \in [0,1]$, the targeted neurons transition via $f_i(t) = (1-t)f_i + tf_p$ and $f_j(t) = (1-t)f_j + tf_p$. By Lemma C.1 and the triangle inequality $\|u\|_{\mathcal{H}} + \|v\|_{\mathcal{H}} \geq \|u+v\|_{\mathcal{H}}$:

$$E(\Phi(t)) \geq E_{\text{rem}} + \|(1-t)(f_i + f_j) + 2tf_p\|_{\mathcal{H}}$$

To prove $E(\Phi(t)) \geq E_{\text{rem}} + \|f_p\|_{\mathcal{H}}$, we require:

$$\|(1-t)(f_i + f_j) + 2tf_p\|_{\mathcal{H}} > \|f_p\|_{\mathcal{H}} \quad \forall\, t \in [0,1] \tag{22}$$

We evaluate this in the collinear limit $\rho_{ij} \to 1$. Let $f_i = x\hat{u}$, $f_j = y\hat{u}$, and $f_p = z\hat{u}$ for a shared unit vector $\hat{u}$ and scalars $x, y, z > 0$. The condition simplifies to:

$$(1-t)(x+y) + 2tz > z$$

Since this expression is linear in t, its minimum occurs at the boundaries:

- **At $t = 1$:** $2z > z$, which inherently holds since $z > 0$.
- **At $t = 0$:** $x + y > z$. From the optimal scale derivation (Appendix E.1), $z = \frac{a+bE_{\text{rem}}}{2E_{\text{rem}}+b}$. In the collinear limit, $a = x^2 + y^2$ and $b = x + y$. Thus:

$$\begin{aligned} x + y &> \frac{x^2 + y^2 + (x+y)E_{\text{rem}}}{2E_{\text{rem}} + x + y} \\ (x+y)(2E_{\text{rem}} + x + y) &> x^2 + y^2 + (x+y)E_{\text{rem}} \\ 2E_{\text{rem}}(x+y) + x^2 + 2xy + y^2 &> x^2 + y^2 + E_{\text{rem}}(x+y) \\ E_{\text{rem}}(x+y) + 2xy &> 0 \end{aligned}$$

  Since $x > 0$, $y > 0$, and $E_{\text{rem}} \geq 0$, this strict inequality unconditionally holds.

Since both endpoints satisfy the strict inequality, it holds for all $t \in [0, 1]$ in the collinear limit. Because the Hilbert norm and f_p are continuous with respect to ρ_{ij}, this inequality is preserved in a neighborhood around $\rho_{ij} = 1$. Thus, there exists a threshold $\rho^*(f_i, f_j) \in (0, 1)$ ensuring the condition for $\rho_{ij} \geq \rho^*$. □

Theorem C.4 (Discrete Transition Cost Bound)**.** For any structural reduction from an initial state $\Phi_a \in \mathcal{H}^N$ to a terminal state $\Phi_b \in \mathcal{H}^{N-1}$ (specifically, pruning a neuron, or merging a correlated pair with $\rho_{ij} \geq \rho^*$), the continuous projection cost $\mathcal{J}_{\text{proj}}$ evaluated along the straight-line path to the pre-deletion target $\tilde{\Phi}_b \in \mathcal{H}^N$ is upper-bounded by the discrete proxy $\mathcal{J}_{\text{bound}}$:

$$\mathcal{J}_{\text{proj}}(\Phi_a, \Phi_b) \leq c(\Phi_a) \frac{D(\Phi_a, \tilde{\Phi}_b)}{E(\Phi_b)} \equiv \mathcal{J}_{\text{bound}}(\Phi_a, \Phi_b) \tag{23}$$

where $E(\Phi_b)$ is the post-deletion capacity, and $D(\Phi_a, \tilde{\Phi}_b) = \left\| \Phi_a - \tilde{\Phi}_b \right\|_{\mathcal{H}^N}$ is the Euclidean distance in the configuration space.

Proof. By Definition 20, the cost along $\Phi(t) = (1-t)\Phi_a + t\tilde{\Phi}_b$ is:

$$\mathcal{J}_{\text{proj}}(\Phi_a, \Phi_b) = \int_0^1 c(\Phi(t)) \frac{\dot{s}(t)}{E(\Phi(t))} dt$$

Since the active neuron count remains invariant prior to $t = 1$, $c(\Phi(t)) = c(\Phi_a)$ almost everywhere on $[0, 1]$ and can be factored out.

Next, we bound the dynamic capacity $E(\Phi(t))$:

- **Pruning:** $E(\Phi(t)) = E_a - t \left\| f_i \right\|_{\mathcal{H}}$. Its minimum is $E(1) = E_a - \left\| f_i \right\|_{\mathcal{H}} = E(\Phi_b)$.
- **Merging:** Lemma C.3 guarantees $E(\Phi(t)) \geq E(\Phi_b)$ for all $t \in [0, 1]$.

In both cases, $E(\Phi(t)) \geq E(\Phi_b) > 0$. Substituting $1/E(\Phi(t)) \leq 1/E(\Phi_b)$ into the integral establishes an upper bound:

$$\mathcal{J}_{\text{proj}}(\Phi_a, \Phi_b) \leq \frac{c(\Phi_a)}{E(\Phi_b)} \int_0^1 \dot{s}(t) dt \tag{24}$$

The geometric speed $\dot{s}(t) = \left\|\dot{\Phi}(t)\right\|_{\mathcal{H}^N} = \left\|\tilde{\Phi}_b - \Phi_a\right\|_{\mathcal{H}^N} \equiv D(\Phi_a, \tilde{\Phi}_b)$ is constant. Evaluating the integral yields $D(\Phi_a, \tilde{\Phi}_b)$, leading to:

$$\mathcal{J}_{\text{proj}}(\Phi_a, \Phi_b) \leq c(\Phi_a)\frac{D(\Phi_a, \tilde{\Phi}_b)}{E(\Phi_b)} \equiv \mathcal{J}_{\text{bound}}(\Phi_a, \Phi_b)$$

□

Proposition C.5 (Axiomatic Consistency of the Bounded Proxy)**.** The bounded projection cost $\mathcal{J}_{\text{bound}}(\Phi_a, \Phi_b) = c(\Phi_a)\frac{D(\Phi_a, \tilde{\Phi}_b)}{E(\Phi_b)}$ satisfies Axiom 1 (Magnitude Neutrality) and Axiom 2 (Connectivity Preservation).

Proof. **Axiom 1 (Magnitude Neutrality):** For any $k > 0$, we have $c(k\Phi_a) = c(\Phi_a)$ (scale-invariant by Theorem C.2), $D(k\Phi_a, k\tilde{\Phi}_b) = kD(\Phi_a, \tilde{\Phi}_b)$, and $E(k\Phi_b) = kE(\Phi_b)$ (linear scaling by Lemma C.1). Thus, k cancels out:

$$\mathcal{J}_{\text{bound}}(k\Phi_a, k\Phi_b) = c(\Phi_a)\frac{kD(\Phi_a, \tilde{\Phi}_b)}{kE(\Phi_b)} = \mathcal{J}_{\text{bound}}(\Phi_a, \Phi_b)$$

Axiom 2 (Connectivity Preservation): For an initial state with $E(\Phi_a) > 0$, we evaluate $\lim_{E(\Phi_b)\to 0^+} \mathcal{J}_{\text{bound}}(\Phi_a, \Phi_b)$.

By the definitions of pruning and merging, $E(\tilde{\Phi}_b) \leq 2E(\Phi_b)$.[3] Thus, $E(\Phi_b) \to 0^+ \implies E(\tilde{\Phi}_b) \to 0$. Because norms are equivalent on a finite-dimensional space, $\|\Phi\|_{\mathcal{H}^N} \leq E(\Phi) \leq \sqrt{N}\,\|\Phi\|_{\mathcal{H}^N}$, which implies $\left\|\tilde{\Phi}_b\right\|_{\mathcal{H}^N} \to 0$.

Applying the reverse triangle inequality gives:

$$D(\Phi_a, \tilde{\Phi}_b) = \left\|\Phi_a - \tilde{\Phi}_b\right\|_{\mathcal{H}^N} \geq \left|\|\Phi_a\|_{\mathcal{H}^N} - \left\|\tilde{\Phi}_b\right\|_{\mathcal{H}^N}\right|$$

Taking the limit as $\left\|\tilde{\Phi}_b\right\|_{\mathcal{H}^N} \to 0$ yields a positive lower bound:

$$\lim_{E(\Phi_b)\to 0^+} D(\Phi_a, \tilde{\Phi}_b) = \|\Phi_a\|_{\mathcal{H}^N} \geq \frac{1}{\sqrt{N}}E(\Phi_a) > 0$$

Since the numerator is bounded below by a positive constant, dividing by $E(\Phi_b) \to 0^+$ strictly diverges to ∞. □

Corollary C.6 (Locality of the Projection Error)**.** For a structural reduction modifying a localized subset of neurons $\mathcal{S} \subset \{1, \ldots, N\}$ (e.g., $|\mathcal{S}| = 1$ for pruning, $|\mathcal{S}| = 2$ for merging), the discrete projection bound evaluates over the perturbed subspace:

$$\mathcal{J}_{\text{bound}}(\Phi_a, \Phi_b) = c(\Phi_a)\frac{\sqrt{\sum_{k\in\mathcal{S}}\left\|f_k^{(a)} - \tilde{f}_k^{(b)}\right\|_{\mathcal{H}}^2}}{E(\Phi_b)} \tag{25}$$

where $\tilde{f}_k^{(b)}$ are the components of the pre-deletion target state $\tilde{\Phi}_b \in \mathcal{H}^N$.

Proof. By Theorem C.4, the Euclidean distance expands as:

$$D(\Phi_a, \tilde{\Phi}_b) = \sqrt{\sum_{k\in\mathcal{S}}\left\|f_k^{(a)} - \tilde{f}_k^{(b)}\right\|_{\mathcal{H}}^2 + \sum_{j\notin\mathcal{S}}\left\|f_j^{(a)} - \tilde{f}_j^{(b)}\right\|_{\mathcal{H}}^2}$$

[3] For pruning, $E(\tilde{\Phi}_b) = E(\Phi_b)$. For merging, $E(\tilde{\Phi}_b) = E_{\text{rem}} + 2\,\|f_p\|_{\mathcal{H}} \leq 2(E_{\text{rem}} + \|f_p\|_{\mathcal{H}}) = 2E(\Phi_b)$.

For any unperturbed coordinate $j \notin \mathcal{S}$, $f_j^{(a)} = \tilde{f}_j^{(b)}$, meaning the second summation vanishes.

The terminal capacity $E(\Phi_b)$ is also computed locally via $E(\Phi_b) = E(\Phi_a) - \sum_{k\in\mathcal{S}} \left\| f_k^{(a)} \right\|_{\mathcal{H}} + \sum_{m\in\mathcal{S}_{\text{new}}} \left\| f_m^{(b)} \right\|_{\mathcal{H}}$, where $\mathcal{S}_{\text{new}}$ represents newly generated components (e.g., $\{f_p\}$). Because $|\mathcal{S}|$ and $|\mathcal{S}_{\text{new}}|$ depend solely on the localized operation, evaluating $\mathcal{J}_{\text{bound}}$ requires $\mathcal{O}(1)$ operations, given $E(\Phi_a)$ is cached. □

C.2 Generating Parent Neuron

Theorem C.7 (Exact Optimal Parent Direction)**.** For a fixed magnitude $s > 0$, the inner optimization for merging neurons $f_i, f_j \in \mathcal{N}$:

$$\psi^* = \operatorname{argmin}_{\substack{\psi\in\mathcal{N} \\ \|\psi\|_{\mathcal{H}}=1}} \frac{\sqrt{\|s\psi - f_i\|_{\mathcal{H}}^2 + \|s\psi - f_j\|_{\mathcal{H}}^2}}{E_a - \|f_i\|_{\mathcal{H}} - \|f_j\|_{\mathcal{H}} + s} \tag{26}$$

is minimized by the parameterized function:

$$\psi^* = \frac{\Psi(\boldsymbol{u}^* \cdot \tilde{\boldsymbol{x}})}{\sqrt{K(\boldsymbol{u}^*, \boldsymbol{u}^*)}} \boldsymbol{v}^* \tag{27}$$

where the unit vectors $\boldsymbol{u}^* \in \mathbb{R}^{n+1}$ and $\boldsymbol{v}^* \in \mathbb{R}^c$ are given by:

$$\boldsymbol{u}^* = \operatorname{argmax}_{\substack{\|\boldsymbol{u}\|_2=1 \\ K(\boldsymbol{u},\boldsymbol{u})>0}} \frac{\left\| \sum_{k\in\{i,j\}} K(\boldsymbol{u}, \tilde{\boldsymbol{w}}_{\text{in}}^k) \boldsymbol{w}_{\text{out}}^k \right\|_2}{\sqrt{K(\boldsymbol{u}, \boldsymbol{u})}} \tag{28}$$

$$\boldsymbol{v}^* = \frac{\sum_{k\in\{i,j\}} K(\boldsymbol{u}^*, \tilde{\boldsymbol{w}}_{\text{in}}^k) \boldsymbol{w}_{\text{out}}^k}{\left\| \sum_{k\in\{i,j\}} K(\boldsymbol{u}^*, \tilde{\boldsymbol{w}}_{\text{in}}^k) \boldsymbol{w}_{\text{out}}^k \right\|_2} \tag{29}$$

Proof. Since $s > 0$ and the denominator of (26) is positive and independent of ψ (due to $\|\psi\|_{\mathcal{H}} = 1$), minimizing the objective is equivalent to minimizing the squared numerator:

$$\begin{aligned} \|s\psi - f_i\|_{\mathcal{H}}^2 + \|s\psi - f_j\|_{\mathcal{H}}^2 &= 2\|s\psi\|_{\mathcal{H}}^2 + \|f_i\|_{\mathcal{H}}^2 + \|f_j\|_{\mathcal{H}}^2 - 2\langle s\psi, f_i + f_j\rangle_{\mathcal{H}} \\ &= 2s^2 + \|f_i\|_{\mathcal{H}}^2 + \|f_j\|_{\mathcal{H}}^2 - 2s\langle \psi, f_i + f_j\rangle_{\mathcal{H}} \end{aligned} \tag{30}$$

Because $s > 0$ and $\|f_i\|_{\mathcal{H}}, \|f_j\|_{\mathcal{H}}$ are constant with respect to ψ, this reduces to maximizing the inner product:

$$\psi^* = \operatorname{argmax}_{\substack{\psi\in\mathcal{N} \\ \|\psi\|_{\mathcal{H}}=1}} \langle \psi, f_i + f_j\rangle_{\mathcal{H}} \tag{31}$$

To enforce $\psi \in \mathcal{N}$, we decompose it into unit directions $\boldsymbol{u} \in \mathbb{R}^{n+1}, \boldsymbol{v} \in \mathbb{R}^c$ and magnitudes $\alpha, \beta > 0$. By the PH-1 property of Ψ, $\psi = \beta\boldsymbol{v}\Psi(\alpha\boldsymbol{u} \cdot \tilde{\boldsymbol{x}}) = \alpha\beta\boldsymbol{v}\Psi(\boldsymbol{u} \cdot \tilde{\boldsymbol{x}})$.

The unit-norm constraint $\|\psi\|_{\mathcal{H}} = 1$ requires:

$$1 = (\alpha\beta)^2 \|\boldsymbol{v}\|_2^2 \mathbb{E}_{\boldsymbol{x}\sim P_{\mathcal{X}}} \left[\Psi^2(\boldsymbol{u} \cdot \tilde{\boldsymbol{x}})\right] = (\alpha\beta)^2 K(\boldsymbol{u}, \boldsymbol{u}) \implies \alpha\beta = \frac{1}{\sqrt{K(\boldsymbol{u}, \boldsymbol{u})}}$$

Assuming $K(\boldsymbol{u}, \boldsymbol{u}) > 0$ (otherwise $\psi = \mathbf{0}$), substituting $\alpha\beta$ gives:

$$\psi = \frac{\Psi(\boldsymbol{u} \cdot \tilde{\boldsymbol{x}})}{\sqrt{K(\boldsymbol{u}, \boldsymbol{u})}} \boldsymbol{v}$$

Substituting this into (31) and expanding the tensor inner product $\langle g \otimes \boldsymbol{v}, h \otimes \boldsymbol{w} \rangle_{\mathcal{H}} = \langle g, h \rangle_{\mathcal{H}_{\text{in}}} \langle \boldsymbol{v}, \boldsymbol{w} \rangle_2$ yields:

$$\begin{aligned}\left\langle \frac{\Psi(\boldsymbol{u}\cdot\tilde{\boldsymbol{x}})}{\sqrt{K(\boldsymbol{u},\boldsymbol{u})}}\boldsymbol{v}, f_i + f_j \right\rangle_{\mathcal{H}} &= \sum_{k\in\{i,j\}} \left\langle \frac{\Psi(\boldsymbol{u}\cdot\tilde{\boldsymbol{x}})}{\sqrt{K(\boldsymbol{u},\boldsymbol{u})}}\boldsymbol{v}, \boldsymbol{w}_{\text{out}}^k \Psi(\tilde{\boldsymbol{w}}_{\text{in}}^k \cdot \tilde{\boldsymbol{x}}) \right\rangle_{\mathcal{H}} \\ &= \frac{1}{\sqrt{K(\boldsymbol{u},\boldsymbol{u})}} \sum_{k\in\{i,j\}} \mathbb{E}_{\boldsymbol{x}\sim P_{\mathcal{X}}} \left[\Psi(\boldsymbol{u}\cdot\tilde{\boldsymbol{x}})\Psi(\tilde{\boldsymbol{w}}_{\text{in}}^k \cdot \tilde{\boldsymbol{x}})\right] \left\langle \boldsymbol{v}, \boldsymbol{w}_{\text{out}}^k \right\rangle_2 \\ &= \left\langle \boldsymbol{v}, \frac{1}{\sqrt{K(\boldsymbol{u},\boldsymbol{u})}} \sum_{k\in\{i,j\}} K(\boldsymbol{u}, \tilde{\boldsymbol{w}}_{\text{in}}^k) \boldsymbol{w}_{\text{out}}^k \right\rangle_2 \end{aligned} \tag{32}$$

Let $\boldsymbol{z}(\boldsymbol{u}) = \sum_{k\in\{i,j\}} K(\boldsymbol{u}, \tilde{\boldsymbol{w}}_{\text{in}}^k)\boldsymbol{w}_{\text{out}}^k$. For a fixed $\boldsymbol{u}$, maximizing $\langle \boldsymbol{v}, \boldsymbol{z}(\boldsymbol{u}) \rangle_2$ subject to $\|\boldsymbol{v}\|_2 = 1$ requires $\boldsymbol{v}$ to align with $\boldsymbol{z}(\boldsymbol{u})$ via the Cauchy-Schwarz inequality:

$$\boldsymbol{v}^*(\boldsymbol{u}) = \frac{\boldsymbol{z}(\boldsymbol{u})}{\|\boldsymbol{z}(\boldsymbol{u})\|_2}$$

Substituting $\boldsymbol{v}^*(\boldsymbol{u})$ back into (32) simplifies the inner product to $\|\boldsymbol{z}(\boldsymbol{u})\|_2 / \sqrt{K(\boldsymbol{u},\boldsymbol{u})}$, leaving $\boldsymbol{u}^*$ as the sole maximizer in Equation (28). □

Proposition C.8 (Separability and Homogeneity of PH-1 Kernels)**.** Let $\Psi : \mathbb{R} \to \mathbb{R}$ be a PH-1 activation function $\forall c > 0, \Psi(cz) = c\Psi(z)$. Let the functional kernel over an isotropic probability distribution $P_{\mathcal{V}}$ be $K(\boldsymbol{x}, \boldsymbol{y}) = \mathbb{E}_{\boldsymbol{v}\sim P_{\mathcal{V}}}[\Psi(\boldsymbol{x}^T\boldsymbol{v})\Psi(\boldsymbol{y}^T\boldsymbol{v})]$. For any unit vector $\boldsymbol{x}$ $\|\boldsymbol{x}\|_2 = 1$ and any $\boldsymbol{y} \neq \boldsymbol{0}$, the kernel factorizes as:

$$K(\boldsymbol{x}, \boldsymbol{y}) = \|\boldsymbol{y}\|_2 \; k(\rho) \quad \text{where} \quad \rho = \left\langle \boldsymbol{x}, \frac{\boldsymbol{y}}{\|\boldsymbol{y}\|_2} \right\rangle_2 \tag{33}$$

Proof. Let $\hat{\boldsymbol{y}} = \boldsymbol{y}/\|\boldsymbol{y}\|_2$. By the PH-1 property and linearity of expectation, the positive magnitude $\|\boldsymbol{y}\|_2$ factors out:

$$K(\boldsymbol{x}, \boldsymbol{y}) = \mathbb{E}_{\boldsymbol{v}\sim P_{\mathcal{V}}}[\Psi(\boldsymbol{x}^T\boldsymbol{v})\Psi(\|\boldsymbol{y}\|_2 \, \hat{\boldsymbol{y}}^T\boldsymbol{v})] = \|\boldsymbol{y}\|_2 \, \mathbb{E}_{\boldsymbol{v}\sim P_{\mathcal{V}}}[\Psi(\boldsymbol{x}^T\boldsymbol{v})\Psi(\hat{\boldsymbol{y}}^T\boldsymbol{v})]$$

Since $P_{\mathcal{V}}$ is isotropic, it is invariant under orthogonal transformations. Let $\boldsymbol{R}$ be an orthogonal matrix such that $\boldsymbol{R}\boldsymbol{x} = \boldsymbol{e}_1$ and $\boldsymbol{R}\hat{\boldsymbol{y}} = \rho\boldsymbol{e}_1 + \sqrt{1-\rho^2}\boldsymbol{e}_2$, where $\rho = \langle \boldsymbol{x}, \hat{\boldsymbol{y}} \rangle_2$.

Applying the change of variables $\boldsymbol{u} = \boldsymbol{R}\boldsymbol{v}$, we have $\boldsymbol{u} \sim P_{\mathcal{V}}$. Substituting $\boldsymbol{x}^T\boldsymbol{v} = \boldsymbol{x}^T\boldsymbol{R}^T\boldsymbol{u} = (\boldsymbol{R}\boldsymbol{x})^T\boldsymbol{u} = \boldsymbol{e}_1^T\boldsymbol{u} = u_1$ and similarly $\hat{\boldsymbol{y}}^T\boldsymbol{v} = \hat{\boldsymbol{y}}^T\boldsymbol{R}^T\boldsymbol{u} = (\boldsymbol{R}\hat{\boldsymbol{y}})^T\boldsymbol{u} = \rho u_1 + \sqrt{1-\rho^2}u_2$ yields:

$$\mathbb{E}_{\boldsymbol{v}\sim P_{\mathcal{V}}}[\Psi(\boldsymbol{x}^T\boldsymbol{v})\Psi(\hat{\boldsymbol{y}}^T\boldsymbol{v})] = \mathbb{E}_{\boldsymbol{u}\sim P_{\mathcal{V}}}[\Psi(u_1)\Psi(\rho u_1 + \sqrt{1-\rho^2}u_2)]$$

Because this expectation depends only on ρ, we can define it as $k(\rho)$, establishing the separable form $K(\boldsymbol{x}, \boldsymbol{y}) = \|\boldsymbol{y}\|_2 \, k(\rho)$. □

Proposition C.9 (Boundary Identity)**.** Let Ψ be a PH-1 function. Under a standard bivariate Gaussian reference distribution with correlation ρ, the induced angular kernel $k(\rho) = \mathbb{E}[\Psi(X)\Psi(Y)]$ satisfies $k(1) = k'(1)$.

Proof. By the PH-1 property, $\Psi(x) = C_+ x\mathbb{I}_{x>0} + C_- x\mathbb{I}_{x<0}$ for constants $C_+ = \Psi(1)$ and $C_- = -\Psi(-1)$. Its derivative (defined almost everywhere) is $\Psi'(x) = C_+\mathbb{I}_{x>0} + C_-\mathbb{I}_{x<0}$.

Let (X, Y) be standard bivariate Gaussian with correlation ρ. At $\rho = 1$, $X = Y$ almost surely, giving $k(1) = \mathbb{E}[\Psi(X)^2]$. By Price's Theorem $\frac{\partial}{\partial\rho}\mathbb{E}[f(X)g(Y)] = \mathbb{E}[f'(X)g'(Y)]$, we have $k'(\rho) = \mathbb{E}[\Psi'(X)\Psi'(Y)]$, which at $\rho = 1$ gives $k'(1) = \mathbb{E}[\Psi'(X)^2]$.

Evaluating these expectations:

$$k(1) = \mathbb{E}\big[(C_+ X\mathbb{I}_{X>0} + C_- X\mathbb{I}_{X<0})^2\big] = C_+^2\mathbb{E}[X^2\mathbb{I}_{X>0}] + C_-^2\mathbb{E}[X^2\mathbb{I}_{X<0}]$$
$$k'(1) = \mathbb{E}\big[(C_+\mathbb{I}_{X>0} + C_-\mathbb{I}_{X<0})^2\big] = C_+^2\mathbb{E}[\mathbb{I}_{X>0}] + C_-^2\mathbb{E}[\mathbb{I}_{X<0}]$$

By the symmetry of the standard normal distribution, $\mathbb{E}[X^2\mathbb{I}_{X>0}] = \mathbb{E}[X^2\mathbb{I}_{X<0}] = \frac{1}{2}\mathbb{E}[X^2] = \frac{1}{2}$ and $\mathbb{E}[\mathbb{I}_{X>0}] = \mathbb{E}[\mathbb{I}_{X<0}] = \frac{1}{2}$. Substituting these yields $k(1) = \frac{1}{2}(C_+^2 + C_-^2)$ and $k'(1) = \frac{1}{2}(C_+^2 + C_-^2)$. Thus, $k(1) = k'(1)$. □

Proposition C.10 (Kernel Positivity at Boundary)**.** For any non-trivial PH-1 function $\Psi(x) \not\equiv 0$, the induced angular kernel satisfies $k(1) > 0$.

Proof. As derived in Proposition C.9, $k(1) = \frac{1}{2}(C_+^2 + C_-^2)$. Since Ψ is non-trivial, at least one of C_+ or C_- is non-zero. Consequently, $C_+^2 + C_-^2 > 0$, implying $k(1) > 0$. □

C.3 Block Eviction

Proposition C.11. Let $\Omega = (\Phi_1, \ldots, \Phi_n)$ be a macroscopic state, where each $\Phi_i \in \mathcal{H}_i^{N_i}$ is a layer state consisting of rank-1 operators $f_{i,j} \in \mathcal{H}_i$. If the macroscopic capacity $E(\Omega)$ is a symmetric, separable, and homogeneous function of the capacities of its constituent operators, and is invariant under arbitrary operator partitioning, it is uniquely defined as the sum of the constituent Hilbert-Schmidt norms:

$$E(\Omega) = \sum_{i=1}^{n} E(\Phi_i) = \sum_{i=1}^{n}\sum_{j=1}^{N_i} \|f_{i,j}\|_{\mathcal{H}_i} \,. \tag{34}$$

Proof. Because $E(\Omega)$ is a symmetric, separable, and homogeneous functional of the individual capacities, it must take the form of an L_p norm:

$$E(\Omega) = \left(\sum_{i=1}^{n}\sum_{j=1}^{N_i} \|f_{i,j}\|_{\mathcal{H}_i}^p\right)^{\frac{1}{p}} \quad \text{for some } p > 0\,. \tag{35}$$

The capacity invariance axiom requires that $E(\Omega)$ remains invariant if any operator is partitioned. Consider an arbitrary operator $f_0 \in \Omega$. Partitioning f_0 into M identical fractional operators yields M operators, each defined as f_0/M. By the positive homogeneity of the Hilbert-Schmidt norm, each fractional operator has capacity $\|f_0/M\|_{\mathcal{H}} = \frac{1}{M}\|f_0\|_{\mathcal{H}}$.

Evaluating the capacity of this partitioned subset under the L_p functional yields:

$$E_{\text{subset}} = \left(\sum_{m=1}^{M} \left\|\frac{f_0}{M}\right\|_{\mathcal{H}}^p\right)^{\frac{1}{p}} = \left(M\left(\frac{1}{M}\|f_0\|_{\mathcal{H}}\right)^p\right)^{\frac{1}{p}} = M^{\frac{1-p}{p}}\|f_0\|_{\mathcal{H}} \,. \tag{36}$$

For the macroscopic capacity to remain invariant for any partition scale $M \geq 1$, we must have $E_{\text{subset}} = \|f_0\|_{\mathcal{H}}$. This equality holds if and only if $M^{\frac{1-p}{p}} = 1$. Since M is arbitrary, the exponent must be zero:

$$\frac{1-p}{p} = 0 \implies p = 1\,. \tag{37}$$

Substituting $p = 1$ reduces the L_p norm to the L_1 sum of scalar norms, yielding $E(\Omega) = \sum_{i=1}^{n}\sum_{j=1}^{N_i}\|f_{i,j}\|_{\mathcal{H}_i}$. □

D LAYER TRANSITION COST DERIVATION

D.1 CONTINUOUS-TIME RELAXATION

Despite the conceptual clarity of these transitions, their discrete nature (where the architecture hops from state Φ_a to Φ_b via pruning or merging) presents a significant barrier to mathematical analysis. To bridge the gap between abstract analysis and algorithmic execution, we proceed in two steps. First, we perform a **continuous relaxation**: instead of an instantaneous jump, we define a continuous deformation $\Phi(t)$ over $t \in [0, 1]$, interpolating between $\Phi(0) = \Phi_a$ and a pre-deletion target $\Phi(1) = \tilde{\Phi}_b$. This allows us to use differential equations to express the infinitesimal cost of shrinking a layer's capacity. Second, to compute the total transition cost, we integrate this differential cost with respect to t. Our objective is to resolve this integral into an expression that depends only on the physically realizable endpoints (Φ_a and $\tilde{\Phi}_b$), bypassing the need to evaluate fictitious intermediate states along the continuous path. However, since this integral generally lacks a closed-form solution, and numerical evaluation incurs computationally prohibitive runtime overhead, we instead derive a **closed-form upper bound** on the analytically intractable integral.

Axiomatic Cost Formulation. To ensure a well-posed definition of $\mathcal{J}$, we introduce the following natural axioms: *1. Magnitude Neutrality:* $\mathcal{J}$ must be scale invariant: $\forall k > 0\,;\, \mathcal{J}(k\,\Phi_a, k\,\Phi_b) = \mathcal{J}(\Phi_a, \Phi_b)$. *2. Connectivity Preservation:* $\mathcal{J}$ must establish an asymptotic barrier preventing layer extinction: $\lim_{E(\Phi_b)\to 0^+} \mathcal{J} = \infty$. *3. Infinitesimal Capacity Dependence:* $\mathcal{J}$ must be additive along continuous paths and be driven by the reduction in layer capacity: $\mathcal{J}(\Phi_a, \Phi_b) = \int_0^1 -\xi(\Phi(t))\dot{E}(t)dt$, where $\dot{E}(t) \triangleq dE(\Phi(t))/dt$ and $\xi(\Phi(t)) > 0$ is a state-dependent density function. While Axioms 1 and 2 define boundaries of the theory, Axiom 3 acts as an idealized analytical tool modeling a continuous capacity drain $\dot{E}(t) < 0$. This allows us to deduce the fundamental shape of the cost function. Under these premises, we can prove (By Theorem C.2) that $\mathcal{J}$ must obey $\mathcal{J}_{\text{capacity}}(\Phi_a, \Phi_b) = \int_0^1 -c(\Phi(t))\frac{\dot{E}(t)}{E(\Phi(t))}dt$, where $\dot{E}(t) < 0$ (due to capacity reduction) and $c(\Phi(t)) > 0$ is a scale-invariant factor (i.e., $c(k\Phi) = c(\Phi)$ for any $\Phi \in \mathcal{H}^N$ and $k > 0$).

Piecewise Constant $c(\Phi(.))$. To bridge continuous theory with discrete execution, we restrict $c(\Phi)$ to remain constant along any discrete state transition $\Phi_a \to \Phi_b$, e.g. $c(\Phi(t)) = c(\Phi_a)$ for $t \in [0, 1]$. This allows us to factor $c(\Phi)$ out of the integral for both $\mathcal{J}_{\text{capacity}}$ and all subsequently derived cost functionals; for $\mathcal{J}_{\text{capacity}}$, this directly yields the analytical solution $\mathcal{J}_{\text{capacity}} = c(\Phi_a)\ln(\frac{E_a}{E_b})$. Upon reaching the terminal state, physically removing extinguished neurons causes $c(\Phi(t))$ to snap to a new value $c(\Phi_b)$. Consequently, $c(\Phi(t))$ acts as a globally piecewise constant function that remains locally constant during any integration step. While $\mathcal{J}_{\text{capacity}}$ is not yet the final objective used in our optimizer, confirming that it satisfies Axioms 1 and 2 ensures we are on track, while its derivation via integration inherently satisfies the idealized capacity dependence assumption.

D.2 BOUNDING THE PROJECTION COST

While $\dot{\mathcal{J}}(t)$ is driven by the relative capacity reduction $-\dot{E}/E$, our framework needs to minimize projection error[4] (Section 5). We bridge the two by calibrating along an orthogonal trajectory where $ds = -dE$ translates the abstract capacity loss $-\dot{E}$ into a geometric speed $\dot{s}$. Here $s(t) = \int_0^t \|\dot{\Phi}(\tau)\|_{\mathcal{H}^N} d\tau$ is the arc-length swept by $\Phi(t)$ through the space $\mathcal{H}^N$. Because $\mathcal{H}^N$ is isotropic, this substitution generalizes to any arbitrary deformation path, yielding $\dot{\mathcal{J}}_{\text{proj}}(t) = c(\Phi(t))\frac{\dot{s}(t)}{E(\Phi(t))}$ (Definition 20). This substitution shifts $\mathcal{J}$ from pure capacity loss to any distance traversed, meaning the strict $\dot{E}(t) < 0$ assumption from the idealized model is no longer required along the physical path.

The compression algorithm executes discrete leaps (e.g., snapping neurons f_i and f_j to a shared parent f_p). Evaluating the cost of this transition conceptually requires integrating $\dot{\mathcal{J}}_{\text{proj}} = c \cdot \dot{s}/E$

[4]Transitioning from $\mathcal{J}_{\text{capacity}}$ to $\mathcal{J}_{\text{proj}}$ ensures sensitivity to feature alignment. For instance, merging two orthogonal neurons introduces a severe subspace projection error while $\mathcal{J}_{\text{capacity}}$ evaluates this catastrophic alignment loss identically to a merge between two collinear (hence redundant) neurons due to their equivalent linear capacity reductions. Shifting to $\mathcal{J}_{\text{proj}}$ reorients the optimization objective from macroscopic reduction to minimizing distortion within the network's internal mapping.

over the jump path. However, because runtime integration is computationally prohibitive, we seek a fast, closed-form proxy. Since underestimating this integral risks destructive jumps (e.g., removing orthogonal features) and breaching layer depletion barriers before the continuous cost can diverge, we derive a *closed-form upper bound* to enforce cautious greedy optimization. We construct this bound by exploiting the inverse relationship between $\dot{\mathcal{J}}_{\text{proj}}$ and E in $\dot{\mathcal{J}}_{\text{proj}} = c \cdot \dot{s}/E$.

For any arbitrary deformation path connecting Φ_a to Φ_b, we can establish an upper bound on the integral cost by replacing the dynamic capacity $E(t)$ with a constant minimum, $E_{\min}$, allowing us to pull the denominator outside the integral. This yields a bounded fractional cost where the numerator is the path's total arc length, $\int_0^1 \dot{s}(t)dt$. Because infinitely many curves in $\mathcal{H}^N$ connect the two states, this establishes a family of valid upper bounds. To tighten this proxy cost, we minimize the numerator by selecting the path with the shortest arc length: the straight-line trajectory in $\mathcal{H}^N$. This evaluates to the traversed Euclidean distance $D(\Phi_a, \tilde{\Phi}_b) \triangleq \|\Phi_a - \tilde{\Phi}_b\|_{\mathcal{H}^N} = (\sum_{k=1}^N \|f_k^{(a)} - \tilde{f}_k^{(b)}\|_{\mathcal{H}}^2)^{\frac{1}{2}}$.

Next, to complete this bound, we must safely approximate the denominator's minimum $E_{\min}$ along this chosen straight-line path. Because the straight-line geometric path acts as a secant across the space of functions (abandoning the strict $\dot{E}(t) < 0$ assumption), the capacity $E(t)$ can temporarily dip below the pre-deletion target $E(\tilde{\Phi}_b)$. To safely absorb this without breaking the integral bound, we introduce a safety buffer by evaluating the denominator at the true terminal state $E(\Phi_b)$. For highly correlated neuron pairs, $E(t) \geq E(\Phi_b)$ throughout the straight-line transition (Lemma C.3).

Substituting the minimized numerator D and evaluating the constant denominator as $E(\Phi_b)$ yields the final bound $\mathcal{J}_{\text{proj}}(\Phi_a, \Phi_b) \leq c(\Phi_a)\frac{D(\Phi_a, \tilde{\Phi}_b)}{E(\Phi_b)} \equiv \mathcal{J}_{\text{bound}}(\Phi_a, \Phi_b)$ (see Theorem C.4).

Practical Notes

The Correlation Constraint. The assumption $E(t) \geq E(\Phi_b)$ holds only for highly correlated neurons, but this poses no practical limitation. Because the projection error $D(\Phi_a, \Phi_b)$ vanishes for collinear candidates, the greedy optimizer naturally minimizes $\mathcal{J}_{\text{bound}}$ by actively selecting highly correlated pairs, inherently satisfying this requirement.

Locality of the Projection Error. Evaluating $\mathcal{J}$ across a wide layer might seem computationally intractable. However, for reductions modifying only a small subset of neurons $\mathcal{S}$ (e.g., pruning or merging), the cost restricts entirely to the perturbed subspace: $\mathcal{J}_{\text{bound}}(\Phi_a, \Phi_b) = c(\Phi_a)\sqrt{\sum_{k\in\mathcal{S}} \|f_k^{(a)} - f_k^{(b)}\|_{\mathcal{H}}^2}/E(\Phi_b)$ (Corollary C.6). This isolates the computation from the total architectural width, guaranteeing $\mathcal{O}(1)$ execution time.

Choice of $c(\Phi) = N$. We previously specified $c(\Phi)$ to be piecewise constant; we now propose a more specific definition: setting $c(\Phi) = N$ for each continuous piece. This is to avoid unfair removal of critical diversity from wide layers by the global optimizer before addressing obvious redundancies in narrow bottlenecks, which may occur as capacity $E(\Phi)$ intrinsically scales with layer width. To mitigate this width bias, we normalize $\mathcal{J}$ using the *average feature capacity*[5] $E(\Phi)/N$, which can be implemented by setting $c(\Phi) = N$. Substituting this $c(\Phi)$ into $\mathcal{J}_{\text{bound}}$ yields the final cost $\mathcal{J}_{\text{final}} \triangleq \frac{N \cdot D(\Phi_a, \tilde{\Phi}_b)}{E(\Phi_b)}$.

E Generating the Parent Neuron

E.1 The Parent Neuron in Hilbert Space

We determine the optimal parent neuron f_p^* by minimizing $\mathcal{J}_{\text{merge}}(f_p)$ subject to $f_p \in \mathcal{N}$, where $\mathcal{N}$ denotes the space of realizable neurons:

$$\mathcal{N} \triangleq \{f \mid f(\boldsymbol{x}) = \boldsymbol{w}_{\text{out}}\Psi(\tilde{\boldsymbol{w}}_{\text{in}} \cdot \tilde{\boldsymbol{x}})\} \subset \mathcal{H}\,. \tag{38}$$

[5]Consider a mean-field assumption where each active neuron contributes an average capacity $\bar{e}$. The incremental cost of pruning a single neuron evaluates to $\mathcal{J}_{\text{prune}} \approx \frac{N\cdot\bar{e}}{N\cdot\bar{e}} = 1$. This normalization renders the penalty invariant to the instantaneous layer width. Without this dynamic coupling (e.g., if c were anchored to N_{initial}), the incremental cost would artificially explode as the live capacity shrinks, forcing an artificial uniformity that prevents the optimizer from fully extinguishing noisy, redundant blocks.

Here $\tilde{\boldsymbol{x}} = [\boldsymbol{x}, 1]^T$ and $\tilde{\boldsymbol{w}}_{\text{in}} = [\boldsymbol{w}_{\text{in}}^{\text{eff}}, b]^T$ denote the augmented inputs and weights. Any non-zero function $f \in \mathcal{H}$ can be decomposed into a scalar magnitude $s > 0$ and a direction $\psi \in \mathcal{H}$, such that $f = s\psi$ and $\|\psi\|_{\mathcal{H}} = 1$. Applying this to the parent neuron f_p allows us to reformulate the search for f_p^* as the following nested optimization problem:

$$\min_{s\in\mathbb{R}^+} \min_{\psi\in\mathcal{N}} \frac{\sqrt{\|f_p - f_i\|_{\mathcal{H}}^2 + \|f_p - f_j\|_{\mathcal{H}}^2}}{E_a - \|f_i\|_{\mathcal{H}} - \|f_j\|_{\mathcal{H}} + \|f_p\|_{\mathcal{H}}} \qquad \text{s.t} \qquad f_p = s\,\psi \quad , \quad \|\psi\|_{\mathcal{H}} = 1 \quad , \quad s > 0 \tag{39}$$

E.1.1 Optimal Direction

We first focus on the inner optimization problem of (39). Expanding the squared numerator of the cost functional reveals that for a fixed magnitude $s > 0$, minimizing the cost in ψ is equivalent to maximizing the alignment $\langle \psi, f_i + f_j \rangle_{\mathcal{H}}$ in ψ. To solve the latter, we enforce the realizability $\psi \in \mathcal{N}$ and unit-norm $\|\psi\|_{\mathcal{H}} = 1$ constraints by decoupling the input and output parameters, yielding the parametric form:

$$\psi = \frac{\Psi(\boldsymbol{u} \cdot \tilde{\boldsymbol{x}})}{\sqrt{K(\boldsymbol{u}, \boldsymbol{u})}} \boldsymbol{v}\,. \tag{40}$$

Substituting this parametric form into the unconstrained alignment objective and distributing the Hilbert inner product via the kernel identity isolates the output direction $\boldsymbol{v}$. By the Cauchy-Schwarz inequality, the optimal $\boldsymbol{v}^*$ must align with $\sum_{k\in\{i,j\}} K(\boldsymbol{u}^*, \tilde{\boldsymbol{w}}_{\text{in}}^k)\boldsymbol{w}_{\text{out}}^k$. Substituting this optimal $\boldsymbol{v}^*$ back into the objective simplifies the alignment inner product to the Euclidean norm of that sum, yielding the final objective for the optimal $\boldsymbol{u}^*$ (Theorem C.7):

$$\boldsymbol{v}^* = \frac{\sum_{k\in\{i,j\}} K(\boldsymbol{u}^*, \tilde{\boldsymbol{w}}_{\text{in}}^k)\boldsymbol{w}_{\text{out}}^k}{\|\sum_{k\in\{i,j\}} K(\boldsymbol{u}^*, \tilde{\boldsymbol{w}}_{\text{in}}^k)\boldsymbol{w}_{\text{out}}^k\|} \;,\; \boldsymbol{u}^* = \operatorname{argmax}_{\|\boldsymbol{u}\|=1} \frac{\|\sum_{k\in\{i,j\}} K(\boldsymbol{u}, \tilde{\boldsymbol{w}}_{\text{in}}^k)\boldsymbol{w}_{\text{out}}^k\|}{\sqrt{K(\boldsymbol{u}, \boldsymbol{u})}} \text{ s.t. } K(\boldsymbol{u}, \boldsymbol{u}) > 0 \tag{41}$$

The above optimization[6] in $\boldsymbol{u}$ generally lacks a closed-form solution due to the non-linear nature of the kernel K. To maintain computational tractability, we introduce an approximation scheme that reduces the objective to an eigenvalue problem. Our approximation assumes that for any unit vector $\boldsymbol{x}$ and non-zero $\boldsymbol{y}$, the kernel factors as $K(\boldsymbol{x}, \boldsymbol{y}) = \|\boldsymbol{y}\| k\left(\left\langle \boldsymbol{x}, \frac{\boldsymbol{y}}{\|\boldsymbol{y}\|} \right\rangle\right)$ for some angular function $k : [-1, 1] \to \mathbb{R}$ bounded by $k(\rho) \le 1$. Additionally, we require $k(1) = k'(1)$ and $k(1) > 0$. These conditions naturally hold for all PH-1 functions (piecewise linear with a single knot at the origin), e.g., ReLU, Leaky-ReLU; see Propositions C.8 to C.10. For highly correlated neuron pairings, the optimal parent direction $\boldsymbol{u}$ aligns closely with its children, pushing their cosine similarity $\rho \triangleq \left\langle \boldsymbol{u}, \frac{\tilde{\boldsymbol{w}}_{\text{in}}}{\|\tilde{\boldsymbol{w}}_{\text{in}}\|} \right\rangle$ toward 1. Expanding $k(\rho)$ to first order around $\rho = 1$ and applying the $k(1) = k'(1)$ identity cancels the constant terms, yielding the linear approximation $k(\rho) \approx \rho k(1)$. While this degrades for unaligned vectors, the phase-check provided later in the section corrects anti-alignment by flipping the sign of $\boldsymbol{u}$, ensuring the optimization trajectory remains safely within this linear domain.

Applying the linear approximation to the numerator of the objective, and defining the constant matrix $\boldsymbol{A} \triangleq \boldsymbol{w}_{\text{out}}^i(\tilde{\boldsymbol{w}}_{\text{in}}^i)^T + \boldsymbol{w}_{\text{out}}^j(\tilde{\boldsymbol{w}}_{\text{in}}^j)^T$, the summation factors neatly: $\sum_{k\in\{i,j\}} K(\boldsymbol{u}, \tilde{\boldsymbol{w}}_{\text{in}}^k)\, \boldsymbol{w}_{\text{out}}^k \approx k(1)\boldsymbol{A}\boldsymbol{u}$. Conversely, the denominator requires no approximation; because $\|\boldsymbol{u}\| = 1$, self-alignment $\rho = 1$ evaluates to $K(\boldsymbol{u}, \boldsymbol{u}) = k(1)$. Substituting these into the original optimization problem gives: $\widehat{\boldsymbol{u}} = \arg\max_{\boldsymbol{u}} \frac{\|k(1)\boldsymbol{A}\boldsymbol{u}\|}{\sqrt{k(1)}} \qquad \text{s.t.} \qquad \|\boldsymbol{u}\| = 1$. Because $k(1) > 0$, the scalars pull out. Dropping these constants and squaring the strictly non-negative objective simplifies the unconstrained problem to a standard quadratic form: $\widehat{\boldsymbol{u}} = \arg\max_{\boldsymbol{u}} \boldsymbol{u}^T\boldsymbol{A}^T\boldsymbol{A}\boldsymbol{u} \qquad \text{s.t.} \qquad \|\boldsymbol{u}\| = 1$. The optimal direction $\widehat{\boldsymbol{u}}$ is simply the principal eigenvector of $\boldsymbol{A}^T\boldsymbol{A}$. While explicitly constructing this ambient matrix is computationally prohibitive, $\boldsymbol{A}$ is fundamentally rank-2. Restricting the eigendecomposition to this rank-2 subspace bypasses the ambient dimension entirely, yielding the principal eigenvector via a fast closed-form solution.

Determining the Sign of $\boldsymbol{u}$. For PH-1 activations, the kernel is sign-sensitive $K(\boldsymbol{u}, \tilde{\boldsymbol{w}}_{\text{in}}) \neq K(-\boldsymbol{u}, \tilde{\boldsymbol{w}}_{\text{in}})$. However, because our linearization approximation relies on the leading eigenvector $\widehat{\boldsymbol{u}}$ of $\boldsymbol{A}^T\boldsymbol{A}$, we only recover the solution up to a sign ambiguity. We resolve this by evaluating

[6] By the PH-1 property of Ψ, the mapping $\boldsymbol{u} \mapsto K(\boldsymbol{u}, \boldsymbol{u})$ is homogeneous, implying that the objective is invariant to the transformation $\boldsymbol{u} \leftarrow c\boldsymbol{u}$ for any $c > 0$. We arbitrarily enforce $\|\boldsymbol{u}\| = 1$ to keep the problem well-posed.

both candidate polarities $\pm\widehat{\boldsymbol{u}}$ in the exact, non-linearized objective (41):

$$\boldsymbol{u}_{\text{correct}} = \text{argmax}_{\boldsymbol{u}\in\{\widehat{\boldsymbol{u}},-\widehat{\boldsymbol{u}}\}} \frac{\left\|\sum_{k\in\{i,j\}} K(\boldsymbol{u}, \tilde{\boldsymbol{w}}_{\text{in}}^k)\boldsymbol{w}_{\text{out}}^k\right\|}{\sqrt{K(\boldsymbol{u},\boldsymbol{u})}} . \tag{42}$$

E.1.2 Optimal Scale

Recall from (40) that the unit-norm direction $\psi \in \mathcal{N}$ is parameterized by unit vectors $\boldsymbol{u}$ and $\boldsymbol{v}$ as $\psi = \frac{\Psi(\boldsymbol{u}\cdot\tilde{\boldsymbol{x}})}{\sqrt{K(\boldsymbol{u},\boldsymbol{u})}}\boldsymbol{v}$. Substituting $f = s\psi$ into the merging cost (39) and defining constants $a \triangleq \|f_i\|_{\mathcal{H}}^2 + \|f_j\|_{\mathcal{H}}^2$, $b \triangleq \langle\psi, f_i + f_j\rangle_{\mathcal{H}}$, and $E_{\text{rem}} \triangleq E_a - \|f_i\|_{\mathcal{H}} - \|f_j\|_{\mathcal{H}}$, minimizing the squared objective reduces to the 1D problem $s^* = \text{argmin}_{s>0} \frac{2s^2-2bs+a}{(s+E_{\text{rem}})^2}$. Setting the derivative with respect to s to zero yields the unique minimizer $s^* = \frac{a+bE_{\text{rem}}}{2E_{\text{rem}}+b}$. This solution is also stable. By definition, the residual capacity $E_{\text{rem}} \geq 0$, and the prior phase-check ensures the alignment in function space $b > 0$. Thus, the denominator is strictly positive, guaranteeing a unique global minimum in the positive domain (simplifying cleanly to $s^* = a/b$ in the event of a total layer collapse where $E_{\text{rem}} = 0$). Once the optimal scale $s^* > 0$ is determined, the parent neuron is fully characterized as shown below.

Optimal Parent Neuron

$$f_p^*(\tilde{\boldsymbol{x}}) = s^*\psi^*(\tilde{\boldsymbol{x}}) \quad , \quad \psi^*(\tilde{\boldsymbol{x}}) = \frac{\Psi(\boldsymbol{u}_{\text{c}}\cdot\tilde{\boldsymbol{x}})}{\sqrt{K(\boldsymbol{u}_{\text{c}},\boldsymbol{u}_{\text{c}})}}\boldsymbol{v}^* \quad , \quad s^* = \frac{\|f_i\|_{\mathcal{H}}^2 + \|f_j\|_{\mathcal{H}}^2 + E_{\text{rem}}\ \langle\psi^*, f_i + f_j\rangle_{\mathcal{H}}}{2E_{\text{rem}} + \langle\psi^*, f_i + f_j\rangle_{\mathcal{H}}} \tag{43}$$

$$\boldsymbol{u}_{\text{c}} = \text{argmax}_{\boldsymbol{u}\in\{\widehat{\boldsymbol{u}},-\widehat{\boldsymbol{u}}\}} \frac{\|\sum_{k\in\{i,j\}} K(\boldsymbol{u}, \tilde{\boldsymbol{w}}_{\text{in}}^k)\boldsymbol{w}_{\text{out}}^k\|}{\sqrt{K(\boldsymbol{u},\boldsymbol{u})}} \quad , \quad \boldsymbol{v}^* = \frac{\sum_{k\in\{i,j\}} K(\boldsymbol{u}_{\text{c}}, \tilde{\boldsymbol{w}}_{\text{in}}^k)\boldsymbol{w}_{\text{out}}^k}{\|\sum_{k\in\{i,j\}} K(\boldsymbol{u}_{\text{c}}, \tilde{\boldsymbol{w}}_{\text{in}}^k)\boldsymbol{w}_{\text{out}}^k\|} \tag{44}$$

$$\widehat{\boldsymbol{u}} = \arg\max_{\|\boldsymbol{u}\|=1} \left\|\left(\boldsymbol{w}_{\text{out}}^i(\tilde{\boldsymbol{w}}_{\text{in}}^i)^T + \boldsymbol{w}_{\text{out}}^j(\tilde{\boldsymbol{w}}_{\text{in}}^j)^T\right)\boldsymbol{u}\right\| . \tag{45}$$

E.2 From Hilbert Space to Physical Parameters

This section bridges the **abstract function space** and the **physical parameter space** by mapping the mathematical operator derived in $\mathcal{H}$ back into physical parameters. This parameter recovery is only necessary for merging. For pruning, the projection target is simply the null operator $\mathbf{0}$, which leads to $f(\boldsymbol{x}) = 0$; this is trivially realized by zeroing out the neuron's incoming weights, outgoing weights, and BN parameters. However, deploying the parent neuron $f_p^* \in \mathcal{H}$ derived in (43) requires determining the physical parameters (weights $\boldsymbol{w}_p^{\text{raw}}, b_p, \boldsymbol{w}_{p,\text{out}}$ and BN statistics $\beta_p, \gamma_p, \mu_p, \sigma_p$) that will configure the forward pass to reproduce its targeted non-zero activation profile.

E.2.1 Input/Output Scaling

To form a standard realizable neuron as described in (38), we equate $f_p^*(\tilde{\boldsymbol{x}}) = \boldsymbol{w}_{\text{out}}^*\Psi(\tilde{\boldsymbol{w}}_{\text{in}}^*\cdot\tilde{\boldsymbol{x}})$ and then specify the parameters $\tilde{\boldsymbol{w}}_{\text{in}}^*$ and $\boldsymbol{w}_{\text{out}}^*$. Because the PH-1 activation Ψ exhibits scale symmetry, the amplitude $s^*/\sqrt{K(\boldsymbol{u}^*,\boldsymbol{u}^*)}$ can be factored into arbitrary input and output scales $\tilde{\boldsymbol{w}}_{\text{in}}^* = s_{\text{in}}\boldsymbol{u}^*$ and $\boldsymbol{w}_{\text{out}}^* = s_{\text{out}}\boldsymbol{v}^*$, for any $s_{\text{in}}, s_{\text{out}} \geq 0$ satisfying $s_{\text{in}}s_{\text{out}} = s^*/\sqrt{K(\boldsymbol{u}^*,\boldsymbol{u}^*)}$. While any factorization yields the same mapping $\mathcal{X} \to \mathbb{R}^c$, amplitude distribution impacts fine-tuning dynamics. To preserve the original layer's balance, we define the subspace Frobenius ratio $R_F \triangleq \|\boldsymbol{W}_{\text{in}}\|_F/\|\boldsymbol{W}_{\text{out}}\|_F$, where $\boldsymbol{W}_{\text{in}} = [\tilde{\boldsymbol{w}}_{\text{in}}^i \mid \tilde{\boldsymbol{w}}_{\text{in}}^j]$ and $\boldsymbol{W}_{\text{out}} = [\boldsymbol{w}_{\text{out}}^i \mid \boldsymbol{w}_{\text{out}}^j]$. Constraining the parent neuron to this ratio requires $\|\tilde{\boldsymbol{w}}_{\text{in}}^*\|_2/\|\boldsymbol{w}_{\text{out}}^*\|_2 = R_F$. Since $\boldsymbol{u}^*$ and $\boldsymbol{v}^*$ are unit vectors, $s_{\text{in}}/s_{\text{out}} = R_F$. This uniquely determines the scale factors, yielding the final parameters:

$$\tilde{\boldsymbol{w}}_{\text{in}}^* = \sqrt{s^*R_F}\cdot K_{\text{self}}^{-1/4}\,\boldsymbol{u}^* \quad , \quad \boldsymbol{w}_{\text{out}}^* = \sqrt{\frac{s^*}{R_F}}\cdot K_{\text{self}}^{-1/4}\,\boldsymbol{v}^* \quad , \quad K_{\text{self}} \triangleq K(\boldsymbol{u}^*,\boldsymbol{u}^*) . \tag{46}$$

E.2.2 Raw Input and BN Parameters

While the Hilbert space formulation operates entirely on the **effective** input parameters $\tilde{\boldsymbol{w}}_{\text{in}} \triangleq (\boldsymbol{w}_{\text{in}}^{\text{eff}}, b)$, realizing the physical network requires recovering the underlying physical parameters: $\boldsymbol{w}^{\text{raw}}, \beta, \gamma, \mu$, and σ. Since the parent direction $\widehat{\boldsymbol{u}}$ lies within the 2D subspace spanned by the augmented children, there exist projection coefficients c_1 and c_2 that produce the effective parameters:

$$\boldsymbol{w}_{p,\text{in}}^{\text{eff}} = c_1 \boldsymbol{w}_{\text{in},i}^{\text{eff}} + c_2 \boldsymbol{w}_{\text{in},j}^{\text{eff}} \quad \text{and} \quad b_p = c_1 b_i + c_2 b_j \,. \tag{47}$$

By mapping these coefficients through the pre-activation distributions of the children, we can deduce the required BN statistics for the parent neuron. Because the physical BN equations form an under-constrained system, we resolve the ambiguity by anchoring the variance such that $\sigma_p^2 = \max(0, \gamma_p^2 - \epsilon)$. As rigorously derived in Appendix F, this anchoring yields a closed-form recovery of all physical parameters. For the active regime $\gamma_p^2 \geq \epsilon$, these evaluate to:

$$\boldsymbol{w}_{\text{in},p}^{\text{raw}} = \boldsymbol{w}_{\text{in},p}^{\text{eff}} \qquad , \qquad \mu_p = c_1 \beta_i + c_2 \beta_j - b_p \tag{48}$$

$$\beta_p = c_1 \beta_i + c_2 \beta_j \qquad , \qquad \sigma_p = \gamma_p = \sqrt{c_1^2 \gamma_i^2 + c_2^2 \gamma_j^2 + 2 c_1 c_2 |\gamma_i| |\gamma_j| \hat{\rho}_{ij}} \,. \tag{49}$$

where $\hat{\rho}_{ij}$ is from (3). Note that $\sigma_p \approx \gamma_p$ is an approximation that assumes the numerical stability constant ϵ is negligible. The exact boundary-safe formulation $\sigma_p^2 = \max(0, \gamma_p^2 - \epsilon)$ and the edge case for inactive features $\gamma_p^2 < \epsilon$ are deferred to Appendix B.5. Furthermore, Appendix F provides the full step-by-step derivation, along with a proof demonstrating that the physical forward pass acts as a self-correcting mechanism that ensures the network's mapping remains invariant to the sign of the recovered scale γ_p.

F Derivation of Physical BN Parameters

This appendix provides the complete derivation for recovering the physical BN parameters and raw weights from the effective parameters, expanding upon Section E.2.2.

Since the parent direction $\widehat{\boldsymbol{u}}$ lies within the 2D subspace spanned by the augmented children, there exist projection coefficients c_1, c_2 such that:

$$\boldsymbol{w}_{p,\text{in}}^{\text{eff}} = c_1 \boldsymbol{w}_{\text{in},i}^{\text{eff}} + c_2 \boldsymbol{w}_{\text{in},j}^{\text{eff}} \quad \text{and} \quad b_p = c_1 b_i + c_2 b_j \,. \tag{50}$$

By the definition of the pre-activation signal $y = (\boldsymbol{w}_{\text{in}}^{\text{eff}})^T \boldsymbol{x} + b$, it follows that $y_p = c_1 y_i + c_2 y_j$. The physical BN parameters β_p and γ_p correspond to the mean and standard deviation of y_p. Taking the expectation and variance yields:

$$\beta_p = \mathbb{E}[y_p] = c_1 \mathbb{E}[y_i] + c_2 \mathbb{E}[y_j] = c_1 \beta_i + c_2 \beta_j \tag{51}$$

$$\gamma_p^2 = \text{Var}[y_p] = c_1^2 \gamma_i^2 + c_2^2 \gamma_j^2 + 2 c_1 c_2 \,\text{Cov}(y_i, y_j) \,. \tag{52}$$

Substituting the closed-form covariance $\text{Cov}(y_i, y_j) = |\gamma_i||\gamma_j|\hat{\rho}_{ij}$ (where $\hat{\rho}_{ij}$ is defined in Equation 3) gives:

$$\gamma_p = \pm \sqrt{c_1^2 \gamma_i^2 + c_2^2 \gamma_j^2 + 2 c_1 c_2 |\gamma_i| |\gamma_j| \hat{\rho}_{ij}} \,. \tag{53}$$

Next, we determine the raw weights $\boldsymbol{w}_{\text{in},p}^{\text{raw}}$ and dataset statistics μ_p, σ_p. By definition (1):

$$\boldsymbol{w}_{\text{in},p}^{\text{eff}} = \frac{\gamma_p}{\sqrt{\sigma_p^2 + \epsilon}} \boldsymbol{w}_{\text{in},p}^{\text{raw}} \qquad \text{and} \qquad b_p = \beta_p - \frac{\gamma_p}{\sqrt{\sigma_p^2 + \epsilon}} \mu_p \,. \tag{54}$$

This system is under-constrained. To resolve the ambiguity, we anchor the variance such that $\sigma_p^2 = \max(0, \gamma_p^2 - \epsilon)$. As noted in the main text, we focus on the active regime $\gamma_p^2 \geq \epsilon$ and defer the edge case $\gamma_p^2 < \epsilon$ to Appendix B.5. In the active regime, the physical scaling factor simplifies directly to its sign:

$$\frac{\gamma_p}{\sqrt{\sigma_p^2 + \epsilon}} = \frac{\gamma_p}{|\gamma_p|} = \text{sign}(\gamma_p) \,. \tag{55}$$

Substituting this back into (54) yields:

$$\boldsymbol{w}_{\text{in},p}^{\text{eff}} = \text{sign}(\gamma_p) \boldsymbol{w}_{\text{in},p}^{\text{raw}} \qquad \text{and} \qquad b_p = \beta_p - \text{sign}(\gamma_p) \mu_p \,. \tag{56}$$

Multiplying both equations by $\text{sign}(\gamma_p)$ (and noting that $\text{sign}(\gamma_p)^2 = 1$ for $\gamma_p \neq 0$) isolates the raw physical parameters:

$$\boldsymbol{w}_{\text{in},p}^{\text{raw}} = \text{sign}(\gamma_p)\boldsymbol{w}_{\text{in},p}^{\text{eff}} \qquad \text{and} \qquad \mu_p = \text{sign}(\gamma_p)(\beta_p - b_p)\,. \tag{57}$$

Because γ_p in (53) can take either sign, we arbitrarily choose the positive root $\gamma_p > 0$ without loss of generality. Under this choice, $\text{sign}(\gamma_p) = 1$, and the recovered parameters simplify to:

$$\boldsymbol{w}_{\text{in},p}^{\text{raw}} = \boldsymbol{w}_{\text{in},p}^{\text{eff}} \quad , \quad \mu_p = c_1\beta_i + c_2\beta_j - b_p \quad , \quad \sigma_p = \gamma_p = \sqrt{c_1^2\gamma_i^2 + c_2^2\gamma_j^2 + 2c_1c_2|\gamma_i||\gamma_j|\hat{\rho}_{ij}}\,. \tag{58}$$

To prove that the physical forward pass is invariant to the chosen sign of γ_p, we substitute the recovered parameters back into the BN inference equation. The true pre-activation signal y_p is computed as:

$$\begin{aligned} y_p &= \frac{\gamma_p}{\sqrt{\sigma_p^2 + \epsilon}} \left((\boldsymbol{w}_{\text{in},p}^{\text{raw}})^T \boldsymbol{x} - \mu_p \right) + \beta_p \\ &= \text{sign}(\gamma_p) \Big[\big(\text{sign}(\gamma_p) \boldsymbol{w}_{\text{in},p}^{\text{eff}} \big)^T \boldsymbol{x} - \text{sign}(\gamma_p)(\beta_p - b_p) \Big] + \beta_p \\ &= \text{sign}(\gamma_p)^2 \Big[(\boldsymbol{w}_{\text{in},p}^{\text{eff}})^T \boldsymbol{x} - (\beta_p - b_p) \Big] + \beta_p\,. \end{aligned} \tag{59}$$

Since $\text{sign}(\gamma_p)^2 = 1$, the shift β_p cancels out:

$$y_p = (\boldsymbol{w}_{\text{in},p}^{\text{eff}})^T \boldsymbol{x} - \beta_p + b_p + \beta_p = (\boldsymbol{w}_{\text{in},p}^{\text{eff}})^T \boldsymbol{x} + b_p\,. \tag{60}$$

Thus, regardless of the polarity of γ_p, the deployed pre-activation robustly realizes the target effective geometry.

G Kernel Formulation

Expressing the Hilbert space inner products and norms via a kernel decouples finite-dimensional vector operations from infinite-dimensional function integrals. We define the correlation kernel $K(i,j)$ of two neurons as:

$$K(i,j) \triangleq \mathbb{E}_{\boldsymbol{x} \sim P_{\mathcal{X}}}[\Psi(y_i)\Psi(y_j)] \tag{61}$$

Under this definition, the inner product in $\mathcal{H}$ evaluates to:

$$\begin{aligned} \langle f_i, f_j \rangle_{\mathcal{H}} &= \langle g_i \otimes \boldsymbol{w}_{\text{out},i}, g_j \otimes \boldsymbol{w}_{\text{out},j} \rangle_{\mathcal{H}} \\ &= \langle g_i, g_j \rangle_{\mathcal{H}_{\text{in}}} \langle \boldsymbol{w}_{\text{out},i}, \boldsymbol{w}_{\text{out},j} \rangle_{\mathbb{R}^c} \\ &= K(i,j) \langle \boldsymbol{w}_{\text{out},i}, \boldsymbol{w}_{\text{out},j} \rangle_{\mathbb{R}^c} \end{aligned} \tag{62}$$

Consequently, the capacity of a single neuron is formulated as:

$$\|f_i\|_{\mathcal{H}} = \|\boldsymbol{w}_{\text{out},i}\|_2 \sqrt{K(i,i)} \tag{63}$$

where $K(i,i) = \mathbb{E}_{\boldsymbol{x} \sim P_{\mathcal{X}}}[\Psi(y_i)^2]$ represents the expected squared energy of the activation signal.

Evaluating $K(i,j)$ requires integrating over a high-dimensional space. We derive closed-form analytical expressions for these integrals under the `ReLU` activation.[7]

G.1 Pre-Activation Distribution

To evaluate the integrals analytically, we first determine the distribution of the pre-activation signal $y_i = (\boldsymbol{w}_{\text{in},i}^{\text{eff}})^T \boldsymbol{x} + b_i$ under the surrogate distribution $P_{\mathcal{X}} = \mathcal{N}(\hat{\boldsymbol{\mu}}_x, \hat{\Sigma}_x)$.

Because y_i is an affine transformation of a Gaussian vector, it is univariate Gaussian. Recall from Section 4 that the surrogate mean is defined as $\hat{\boldsymbol{\mu}}_x = \boldsymbol{W}_{\text{raw}}^{+} \boldsymbol{\mu}_{\text{BN}}$. Because $\boldsymbol{w}_{\text{raw},i}^T \hat{\boldsymbol{\mu}}_x$ is the i-th element

[7] This evaluation connects to the Arc-Cosine Kernel family of order $n = 1$ Cho & Saul (2009). Because HOPE relies on the translation vector β to capture the full statistical state of the signal, we provide the formal proof for the biased formulation here to ensure the paper remains self-contained.

of $\boldsymbol{W}_{\text{raw}}\hat{\boldsymbol{\mu}}_x$, we have $\boldsymbol{W}_{\text{raw}}\hat{\boldsymbol{\mu}}_x = \boldsymbol{W}_{\text{raw}}\boldsymbol{W}_{\text{raw}}^{+}\boldsymbol{\mu}_{\text{BN}}$. Since the empirical BN mean vector $\boldsymbol{\mu}_{\text{BN}}$ resides in the column space of $\boldsymbol{W}_{\text{raw}}$, this projection simplifies to $\boldsymbol{\mu}_{\text{BN}}$. Thus, the expected value of the raw projection recovers the empirical mean: $\mathbb{E}[\boldsymbol{w}_{\text{raw},i}^T\boldsymbol{x}] = \mu_i$. Similarly, the surrogate covariance $\hat{\Sigma}_x$ is constrained to satisfy $\text{Var}(\boldsymbol{w}_{\text{raw},i}^T\boldsymbol{x}) = \sigma_i^2$.

Using the effective parameters $\boldsymbol{w}_{\text{in},i}^{\text{eff}} = \frac{\gamma_i}{\sqrt{\sigma_i^2+\epsilon}}\boldsymbol{w}_{\text{raw},i}$ and $b_i = \beta_i - \frac{\gamma_i\mu_i}{\sqrt{\sigma_i^2+\epsilon}}$, the mean of y_i evaluates to:

$$\mathbb{E}[y_i] = \frac{\gamma_i}{\sqrt{\sigma_i^2+\epsilon}}\mathbb{E}[\boldsymbol{w}_{\text{raw},i}^T\boldsymbol{x}] + b_i = \frac{\gamma_i\mu_i}{\sqrt{\sigma_i^2+\epsilon}} + \beta_i - \frac{\gamma_i\mu_i}{\sqrt{\sigma_i^2+\epsilon}} = \beta_i \tag{64}$$

Similarly, its variance evaluates to:

$$\text{Var}(y_i) = \frac{\gamma_i^2}{\sigma_i^2+\epsilon}\,\text{Var}(\boldsymbol{w}_{\text{raw},i}^T\boldsymbol{x}) = \gamma_i^2\left(\frac{\sigma_i^2}{\sigma_i^2+\epsilon}\right) \tag{65}$$

Assuming the numerical stability constant ϵ is negligible compared to the data variance $\epsilon \to 0$, the variance simplifies to γ_i^2. Thus, under $P_{\mathcal{X}}$, the pre-activation density is:

$$y_i \sim \mathcal{N}(\beta_i, \gamma_i^2) \tag{66}$$

Realizing Empirical Constraints in Practice

Adapting HOPE to unnormalized networks requires a lightweight empirical calibration pass over a small, unlabeled data batch to measure the marginal pre-activation statistics (μ_i, σ_i^2). Defining the effective scaling as $\gamma_i = \sigma_i$ and the shift as $\beta_i = \mu_i + b_{\text{raw},i}$ anchors the evaluation to the data manifold and recovers the decoupling required to evaluate the kernel analytically.[a]

[a] Networks using LayerNorm or GroupNorm still require this empirical calibration pass to determine the marginal channel statistics. Because these layers condition the signal variance, their calibration converges with remarkably few samples, and the resulting pre-activations adhere to the Gaussian surrogate assumptions.

G.2 Self-Kernel

Proposition G.1 (Dimensionality Reduction)**.** For a Gaussian pre-activation $y_i \sim \mathcal{N}(\beta_i, \gamma_i^2)$ and any activation function Ψ, the expected energy $K(i,i) = \mathbb{E}[\Psi(y_i)^2]$ reduces to a 1D integral, independent of the input space dimensionality.

Proof. By definition, $K(i,i) = \mathbb{E}_{\boldsymbol{x}\sim P_{\mathcal{X}}}[\Psi(y_i)^2]$. Because y_i is a scalar random variable with density $p(y_i) = \mathcal{N}(y_i; \beta_i, \gamma_i^2)$, the expectation evaluates as $\int_{-\infty}^{\infty}\Psi(y)^2 p(y)dy$, decoupling the computation from the ambient space $\mathcal{X}$. □

Theorem G.2 (Closed-Form Self-Kernel for ReLU)**.** For $y_i \sim \mathcal{N}(\beta_i, \gamma_i^2)$ with $|\gamma_i| > 0$, the expected squared energy for $\Psi(y_i) = \max(0, y_i)$ evaluates to:

$$K(i,i) = (\gamma_i^2 + \beta_i^2)\Phi\left(\frac{\beta_i}{|\gamma_i|}\right) + \beta_i|\gamma_i|\phi\left(\frac{\beta_i}{|\gamma_i|}\right) \tag{67}$$

where ϕ and Φ are the standard Normal PDF and CDF.

Proof. By Proposition G.1, $K(i,i) = \int_0^{\infty} y^2 p(y)dy$. Applying the substitution $z = \frac{y-\beta_i}{|\gamma_i|}$, we have $y = |\gamma_i|z + \beta_i$ and $dy = |\gamma_i|dz$. Setting the integration limit $c = \frac{\beta_i}{|\gamma_i|}$, we obtain:

$$K(i,i) = \int_{-c}^{\infty}(|\gamma_i|z+\beta_i)^2\phi(z)dz = \beta_i^2\int_{-c}^{\infty}\phi(z)dz + 2\beta_i|\gamma_i|\int_{-c}^{\infty}z\phi(z)dz + \gamma_i^2\int_{-c}^{\infty}z^2\phi(z)dz$$

We evaluate each term using integration by parts and the identity $\phi'(z) = -z\phi(z)$:

- $\int_{-c}^{\infty} \phi(z)dz = 1 - \Phi(-c) = \Phi(c)$
- $\int_{-c}^{\infty} z\phi(z)dz = [-\phi(z)]_{-c}^{\infty} = \phi(c)$
- $\int_{-c}^{\infty} z^2\phi(z)dz = [-z\phi(z)]_{-c}^{\infty} + \int_{-c}^{\infty} \phi(z)dz = \Phi(c) - c\phi(c)$

Substituting these evaluations yields:

$$\begin{aligned} K(i,i) &= \beta_i^2\Phi(c) + 2\beta_i|\gamma_i|\phi(c) + \gamma_i^2\big(\Phi(c) - c\phi(c)\big) \\ &= (\gamma_i^2 + \beta_i^2)\Phi(c) + \big(2\beta_i|\gamma_i| - \gamma_i^2 c\big)\phi(c) \end{aligned}$$

Since $\gamma_i^2 c = \gamma_i^2(\frac{\beta_i}{|\gamma_i|}) = |\gamma_i|\beta_i$, the coefficient for $\phi(c)$ simplifies to $\beta_i|\gamma_i|$. Substituting $c = \frac{\beta_i}{|\gamma_i|}$ completes the proof. □

G.3 Cross-Kernel

G.3.1 The Local Pairwise Surrogate Distribution

Let $\rho_{\text{eff}} = \frac{\langle \boldsymbol{w}_{\text{in},i}^{\text{eff}}, \boldsymbol{w}_{\text{in},j}^{\text{eff}} \rangle}{\|\boldsymbol{w}_{\text{in},i}^{\text{eff}}\|_2 \|\boldsymbol{w}_{\text{in},j}^{\text{eff}}\|_2}$ be the cosine similarity between the effective input weights. The cross-kernel relies on the joint pre-activation distribution of y_i and y_j. Because optimizing the global covariance $\hat{\Sigma}_x$ is computationally prohibitive, we restrict the maximum entropy formulation to the local 2×2 subspace spanned by the neuron pair.

Proposition G.3 (Pairwise Warped Correlation)**.** Under a local pairwise maximum entropy surrogate, the correlation $\hat{\rho}_{ij}$ between y_i and y_j is given analytically by:

$$\hat{\rho}_{ij} = \frac{2\kappa}{1 + \sqrt{1 + 4\kappa^2}} \tag{68}$$

where κ is the blending constant uniquely defined by the input weight geometry and empirical standard deviations[a]:

$$\kappa = \left(\frac{\rho_{\text{eff}}}{1 - \rho_{\text{eff}}^2}\right)\left(\frac{|\gamma_i|}{\left\|\boldsymbol{w}_{\text{in},i}^{\text{eff}}\right\|_2}\right)\left(\frac{|\gamma_j|}{\left\|\boldsymbol{w}_{\text{in},j}^{\text{eff}}\right\|_2}\right) \tag{69}$$

[a] To prevent undefined $0/0$ states, this evaluation is restricted to active features where the effective weight norm is non-zero $\left\|\boldsymbol{w}_{\text{in}}^{\text{eff}}\right\|_2 > 0$.

Proof. Let $\boldsymbol{W} = [\boldsymbol{w}_{\text{raw},i}, \boldsymbol{w}_{\text{raw},j}] \in \mathbb{R}^{n\times 2}$ be the raw weights and $\boldsymbol{G} = \boldsymbol{W}^T\boldsymbol{W}$ be their Gram matrix. The maximum entropy distribution constrained by the variances of these two projections has a precision matrix $\hat{\Sigma}_x^{-1} = \boldsymbol{I} + \boldsymbol{W}\boldsymbol{\Lambda}\boldsymbol{W}^T$, where $\boldsymbol{\Lambda} = \text{diag}(\lambda_i, \lambda_j)$ contains the Lagrange multipliers. The output covariance is $\boldsymbol{C} = \boldsymbol{W}^T\hat{\Sigma}_x\boldsymbol{W} = \boldsymbol{W}^T(\boldsymbol{I} + \boldsymbol{W}\boldsymbol{\Lambda}\boldsymbol{W}^T)^{-1}\boldsymbol{W}$.

Applying the Woodbury matrix identity to the inner inverse yields:

$$(\boldsymbol{I} + \boldsymbol{W}\boldsymbol{\Lambda}\boldsymbol{W}^T)^{-1} = \boldsymbol{I} - \boldsymbol{W}(\boldsymbol{\Lambda}^{-1} + \boldsymbol{W}^T\boldsymbol{W})^{-1}\boldsymbol{W}^T = \boldsymbol{I} - \boldsymbol{W}(\boldsymbol{\Lambda}^{-1} + \boldsymbol{G})^{-1}\boldsymbol{W}^T$$

Substituting this back into the expression for $\boldsymbol{C}$:

$$\boldsymbol{C} = \boldsymbol{W}^T\left[\boldsymbol{I} - \boldsymbol{W}(\boldsymbol{\Lambda}^{-1} + \boldsymbol{G})^{-1}\boldsymbol{W}^T\right]\boldsymbol{W} = \boldsymbol{G} - \boldsymbol{G}(\boldsymbol{\Lambda}^{-1} + \boldsymbol{G})^{-1}\boldsymbol{G}$$

We factor out $\boldsymbol{G}$ to simplify the expression:

$$\begin{aligned} \boldsymbol{C} &= \boldsymbol{G}\left[\boldsymbol{I} - (\boldsymbol{\Lambda}^{-1} + \boldsymbol{G})^{-1}\boldsymbol{G}\right] \\ &= \boldsymbol{G}\left[(\boldsymbol{\Lambda}^{-1} + \boldsymbol{G})^{-1}(\boldsymbol{\Lambda}^{-1} + \boldsymbol{G}) - (\boldsymbol{\Lambda}^{-1} + \boldsymbol{G})^{-1}\boldsymbol{G}\right] \\ &= \boldsymbol{G}(\boldsymbol{\Lambda}^{-1} + \boldsymbol{G})^{-1}\boldsymbol{\Lambda}^{-1} \end{aligned}$$

Inverting both sides yields the localized precision matrix $\boldsymbol{C}^{-1}$:

$$\boldsymbol{C}^{-1} = \boldsymbol{\Lambda}(\boldsymbol{\Lambda}^{-1} + \boldsymbol{G})\boldsymbol{G}^{-1} = \boldsymbol{\Lambda}\boldsymbol{\Lambda}^{-1}\boldsymbol{G}^{-1} + \boldsymbol{\Lambda}\boldsymbol{G}\boldsymbol{G}^{-1} = \boldsymbol{G}^{-1} + \boldsymbol{\Lambda}$$

Because $\boldsymbol{\Lambda}$ is diagonal, its addition only perturbs the diagonal entries of $\boldsymbol{G}^{-1}$. Therefore, the off-diagonal entries are identical: $[\boldsymbol{C}^{-1}]_{12} = [\boldsymbol{G}^{-1}]_{12}$.

Let ρ_{raw} be the correlation in $\boldsymbol{G}$ and r_{raw} be the target correlation in $\boldsymbol{C}$. Evaluating the inverse off-diagonal elements of these 2×2 matrices yields:

$$[\boldsymbol{G}^{-1}]_{12} = \frac{-\rho_{\text{raw}}}{\|\boldsymbol{w}_{\text{raw},i}\|_2 \|\boldsymbol{w}_{\text{raw},j}\|_2 (1 - \rho_{\text{raw}}^2)} \quad , \quad [\boldsymbol{C}^{-1}]_{12} = \frac{-r_{\text{raw}}}{\sigma_i \sigma_j (1 - r_{\text{raw}}^2)}$$

Equating them gives:

$$\frac{r_{\text{raw}}}{1 - r_{\text{raw}}^2} = \left(\frac{\rho_{\text{raw}}}{1 - \rho_{\text{raw}}^2}\right)\left(\frac{\sigma_i}{\|\boldsymbol{w}_{\text{raw},i}\|_2}\right)\left(\frac{\sigma_j}{\|\boldsymbol{w}_{\text{raw},j}\|_2}\right)$$

By the definition of the effective weights, $\boldsymbol{w}_{\text{in},i}^{\text{eff}} = \frac{\gamma_i}{\sqrt{\sigma_i^2 + \epsilon}} \boldsymbol{w}_{\text{raw},i}$. The effective correlations relate to the raw correlations via the signs of these scaling parameters: $\rho_{\text{raw}} = \text{sign}(\gamma_i \gamma_j)\rho_{\text{eff}}$ and $r_{\text{raw}} = \text{sign}(\gamma_i \gamma_j)\hat{\rho}_{ij}$. Substituting these into the odd function $f(x) = \frac{x}{1-x^2}$ causes the sign terms to cancel. Taking the limit as $\epsilon \to 0$, we substitute $\frac{\sigma_i}{\|\boldsymbol{w}_{\text{raw},i}\|_2} = \frac{|\gamma_i|}{\|\boldsymbol{w}_{\text{in},i}^{\text{eff}}\|_2}$, which yields the blended constant κ:

$$\frac{\hat{\rho}_{ij}}{1 - \hat{\rho}_{ij}^2} = \kappa \implies \kappa\hat{\rho}_{ij}^2 + \hat{\rho}_{ij} - \kappa = 0$$

Solving for $\hat{\rho}_{ij}$ via the quadratic formula and selecting the root that satisfies the boundary condition $\lim_{\rho_{\text{eff}} \to 0} \hat{\rho}_{ij} = 0$, we obtain $\frac{-1+\sqrt{1+4\kappa^2}}{2\kappa}$. Multiplying the numerator and denominator by the conjugate $(\sqrt{1+4\kappa^2} + 1)$ ensures numerical stability as $\kappa \to 0$, producing the final formulation $\frac{2\kappa}{1+\sqrt{1+4\kappa^2}}$. □

Consequently, the joint pre-activation distribution under the surrogate space is:

$$\begin{pmatrix} y_i \\ y_j \end{pmatrix} \sim \mathcal{N}\left(\begin{pmatrix} \beta_i \\ \beta_j \end{pmatrix}, \begin{pmatrix} \gamma_i^2 & |\gamma_i||\gamma_j|\hat{\rho}_{ij} \\ |\gamma_i||\gamma_j|\hat{\rho}_{ij} & \gamma_j^2 \end{pmatrix}\right) \tag{70}$$

Regardless of the activation Ψ, any valid Cross-Kernel $K(i,j) = \mathbb{E}[\Psi(y_i)\Psi(y_j)]$ must satisfy three properties:

1. **Diagonal Consistency:** If $\hat{\rho}_{ij} = 1$ and the marginals are identical ($\beta_i = \beta_j$, $|\gamma_i| = |\gamma_j|$), the interaction recovers the self-kernel: $K(i,j) = K(i,i)$.
2. **Cauchy-Schwarz Compliance:** The magnitude is bounded: $|K(i,j)| \leq \sqrt{K(i,i)K(j,j)}$.
3. **Weight-Space Correlation Dependency:** The interaction is monotonic with respect to $\hat{\rho}_{ij}$.

G.3.2 Exact Bivariate Cross-Kernel for Biased ReLUs

Evaluating $K(i,j) = \mathbb{E}[\max(0, y_i)\max(0, y_j)]$ requires integrating over the joint positive orthant. Defining $c_i = \frac{\beta_i}{|\gamma_i|}$ and $c_j = \frac{\beta_j}{|\gamma_j|}$, the exact cross-kernel evaluates to the closed-form moments of a truncated bivariate normal:

$$K(i,j) = |\gamma_i \gamma_j| \Bigg[(c_i c_j + \hat{\rho}_{ij})\Phi_2(c_i, c_j; \hat{\rho}_{ij}) + c_i \phi(c_j)\Phi(c_{i|j}) + c_j \phi(c_i)\Phi(c_{j|i}) + (1 - \hat{\rho}_{ij}^2)\phi_2(c_i, c_j; \hat{\rho}_{ij}) \Bigg] \tag{71}$$

where Φ_2 and ϕ_2 are the standard bivariate normal CDF and PDF evaluated at (c_i, c_j) with correlation $\hat{\rho}_{ij}$, ϕ and Φ are the standard univariate normal PDF and CDF, and the conditional integration boundaries are $c_{i|j} = \frac{c_i - \hat{\rho}_{ij} c_j}{\sqrt{1-\hat{\rho}_{ij}^2}}$ and $c_{j|i} = \frac{c_j - \hat{\rho}_{ij} c_i}{\sqrt{1-\hat{\rho}_{ij}^2}}$. By centering the evaluation on β_i rather than the weight bias b_i, we correctly evaluate active feature detectors in the function space.

G.3.3 Zero-Bias Approximation for Large-Scale Networks

Because calculating Φ_2 for all neuron pairs is computationally intensive, we approximate the interaction by assuming bias shifts are negligible $\beta_i, \beta_j \approx 0$. Under this assumption, $K(i,j)$ factors into the geometric mean of the capacities scaled by a normalized interaction function $\mathcal{I}(\hat{\rho}_{ij})$:

$$K(i,j) \approx \mathcal{I}(\hat{\rho}_{ij})\sqrt{K(i,i)K(j,j)} \tag{72}$$

For the ReLU activation, applying the Arc-Cosine kernel of order $n = 1$ yields the approximate cross-kernel:

$$K(i,j) \approx \frac{1}{\pi}\left(\sqrt{1-\hat{\rho}_{ij}^2} + (\pi - \arccos\hat{\rho}_{ij})\hat{\rho}_{ij}\right)\sqrt{K(i,i)K(j,j)} \tag{73}$$

This formulation complies with the Cauchy-Schwarz inequality $|K(i,j)| \leq \sqrt{K(i,i)K(j,j)}$ and ensures diagonal consistency ($K(i,j) = K(i,i)$ when $\hat{\rho}_{ij} = 1$).

H Derivations for Block Eviction

This appendix provides the mathematical framework and extended derivations for the block eviction operation introduced in Section 7.

H.1 Generalization and Execution Degradation in Depleted Blocks

A dedicated macro-level operation is necessary because standard granular pruning cannot fully deplete a residual block due to architectural constraints. Because the residual connection computes $X + F(X)$, the output dimension of $F(X)$ (and thus the final weight tensor W_3) must match that of X. While granular pruning can safely compress internal neurons within W_1 and W_2, pruning the output filters of W_3 would cause a dimensional mismatch in the element-wise addition $X + F(X)$. Consequently, granular compression leaves the output channels of W_3 intact. Leaving a residual pathway active with heavily depleted W_1 and W_2 degrades both model generalization and execution efficiency:

- **Model Generalization.** The output of the final BN layer evaluates to $F(X) = \boldsymbol{\gamma} \odot \frac{W_3 H_2 - \boldsymbol{\mu}}{\sqrt{\boldsymbol{\sigma}^2+\epsilon}} + \boldsymbol{\beta}$. When $H_2 \to \mathbf{0}$, this reduces to $F(X) = \boldsymbol{\beta} - \boldsymbol{\gamma} \odot \frac{\boldsymbol{\mu}}{\sqrt{\boldsymbol{\sigma}^2+\epsilon}}$. As defined in (1), this is the effective bias vector $\boldsymbol{b}$. Once the block is depleted, $\boldsymbol{b}$ loses its normalization purpose and instead acts as an uncalibrated bias injected into the skip connection, yielding $Y = X + \boldsymbol{b}$. This forcefully shifts downstream feature maps out of their calibrated domain. Since the block terminates with a ReLU activation $Z = \texttt{ReLU}(X + \boldsymbol{b})$, a negative $\boldsymbol{b}$ causes catastrophic clipping and irreversible information loss.
- **Execution Efficiency.** Even if H_2 partially survives, the shape of W_3 remains locked at its ambient output size. Retaining this massive parameter tensor simply to process a negligible, low-rank subspace is computationally wasteful.

Block eviction resolves these issues by projecting $F(X) \to \mathbf{0}$, yielding a pure identity mapping $Y = X$. Unlike $Y = X + \boldsymbol{b}$, residual architectures are natively robust to pure identity mappings. For instance, initializing $\gamma = 0$ in the final layer of each residual branch Goyal et al. (2017); He et al. (2016) ensures blocks begin training as $F(X) = \mathbf{0}$.

H.2 Derivation of the Unified Macro Cost $\mathcal{J}_{\text{EVICT}}$

Recall that a state tuple in layer l has the form $\Phi^{(l)} = (f_1^{(l)}, \ldots, f_{N_l}^{(l)}) \in \mathcal{H}_l^{N_l}$. Let $\mathcal{I} = (I_1, \ldots, I_{d_{\text{amb}}}) \in \mathcal{H}_A^{d_{\text{amb}}}$ be the tuple of identity mappings comprising the skip connection, where $\mathcal{H}_A$ is the Hilbert space of operators over the ambient dimension $\mathbb{R}^{d_{\text{amb}}} \to \mathbb{R}^{d_{\text{amb}}}$. The aggregate mapping of any tuple is the sum of its constituent operators:

$$\mathcal{F}(\Phi^{(l)}) \triangleq \sum_{i=1}^{N_l} f_i^{(l)} \qquad , \qquad \mathcal{F}(\mathcal{I}) = \sum_{k=1}^{d_{\text{amb}}} I_k \,. \tag{74}$$

The composite block operator $\mathcal{B} \in \mathcal{H}_A$ is formalized as the addition of the residual pathway $\mathcal{F}_{\text{res}} \in \mathcal{H}_A$ and the skip connection:

$$\mathcal{B} = \mathcal{F}_{\text{res}}\big(\Phi^{(1)}, \Phi^{(2)}\big) + \mathcal{F}(\mathcal{I})\,. \tag{75}$$

We define eviction iteratively: we project internal layers to null operators until the block is fully depleted. Evaluating the projection of an internal layer $\Phi^{(l)}$ over $t \in [0,1]$ induces a continuous trajectory $\Phi^{(l)}(t) \to (\mathbf{0}, \ldots, \mathbf{0})$. Let $\mathcal{F}_{\text{res}}(t)$ denote the sequential pathway where $\Phi^{(l)}(t) \to \mathbf{0}$ while other layers are held constant. The composite block trajectory is:

$$\mathcal{B}(t) = \mathcal{F}_{\text{res}}(t) + \mathcal{F}(\mathcal{I})\,. \tag{76}$$

From Section 6, the capacity cost for a microscopic layer transition is:

$$\mathcal{J}_{\text{capacity}}(\Phi_a, \Phi_b) = \int_0^1 -c(\Phi(t))\frac{\frac{d}{dt}E(\Phi(t))}{E(\Phi(t))}dt\,. \tag{77}$$

To generalize this to macroscopic blocks, we expand the state to a composite tuple $\Omega \triangleq (T_1, \ldots, T_n)$, where $T_i \in \mathcal{H}_i^{N_i}$. By Proposition C.11, extending $E(\cdot)$ under constraints of symmetry, separability, and homogeneity yields:

$$E(\Omega) = \sum_{i=1}^{n} E(T_i) = \sum_{i=1}^{n}\sum_{j=1}^{N_i} \|f_{i,j}\|_{\mathcal{H}_i}\,. \tag{78}$$

For consistency, the constituent tuples of Ω must interact strictly additively within $\mathcal{B}$. Grouping sequentially composed layers (e.g., $\Phi^{(1)}$ and $\Phi^{(2)}$) into the same state creates a contradiction: projecting $\Phi^{(1)} \to \mathbf{0}$ collapses the entire sequential pathway $\mathcal{F}_{\text{res}} \to \mathbf{0}$, leaving $\mathcal{B} = \mathcal{F}(\mathcal{I})$ with a true capacity of $E(\mathcal{I})$. However, $E(\Omega)$ would erroneously evaluate to $E(\Phi^{(2)}) + E(\mathcal{I})$. To avoid this, we must restrict the macroscopic state strictly to additive components: a single internal layer l and the skip connection $\mathcal{I}$.

Furthermore, because the mapping $\Phi^{(l)}$ and the skip connection $\mathcal{I}$ operate in different dimensional spaces ($d_{\text{bottleneck}}$ versus d_{amb}), direct operator addition $\Phi^{(l)} + \mathcal{I}$ is mathematically undefined. We bypass this by defining the state as the composite pair $\Omega^{(l)} \triangleq (\Phi^{(l)}, \mathcal{I})$. The capacity cost for this macroscopic state is:

$$\mathcal{J}_{\text{capacity}}(\Omega_a^{(l)}, \Omega_b^{(l)}) = \int_0^1 -c(\Omega^{(l)}(t))\frac{\frac{d}{dt}E(\Phi^{(l)}(t)) + \frac{d}{dt}E(\mathcal{I})}{E(\Phi^{(l)}(t)) + E(\mathcal{I})}dt\,. \tag{79}$$

To mitigate layer-width bias, $c(\cdot)$ acts as a counting measure of compressible operators. Being additive over disjoint sets, $c(\Omega^{(l)}) = c(\Phi^{(l)}) + c(\mathcal{I})$. Since the skip connection $\mathcal{I}$ is fixed, it has no compressible operators $c(\mathcal{I}) = 0$. Thus, $c(\Omega^{(l)}(t)) = c(\Phi^{(l)}(t))$. Combined with the stationarity of the skip connection $\frac{d}{dt}E(\mathcal{I}) = 0$, the integral simplifies to:

$$\mathcal{J}_{\text{capacity}}(\Omega_a^{(l)}, \Omega_b^{(l)}) = \int_0^1 -c(\Phi^{(l)}(t))\frac{\frac{d}{dt}E(\Phi^{(l)}(t))}{E(\Phi^{(l)}(t)) + E(\mathcal{I})}dt\,. \tag{80}$$

During the projection $\Phi_a^{(l)} \to \mathbf{0}$, $c(\Phi^{(l)}(t))$ drops from $N_{\text{active}}^{(l)}$ to 0. Bounding this density by its maximum pre-action value $N_{\text{active}}^{(l)}$ allows a closed-form integration. Defining $E(t) \triangleq E(\Phi^{(l)}(t))$, $\dot{E}(t) \triangleq \frac{d}{dt}E(t)$, and $E_{\text{identity}} \triangleq E(\mathcal{I})$:

$$\mathcal{J}_{\text{capacity}}(\Omega_a^{(l)}, \Omega_b^{(l)}) \le N_{\text{active}}^{(l)} \int_0^1 \frac{-\dot{E}(t)}{E(t) + E_{\text{identity}}}\,dt = N_{\text{active}}^{(l)} \ln\left(1 + \frac{E_{\text{active}}^{(l)}}{E_{\text{identity}}}\right)\,. \tag{81}$$

This logarithmic formulation penalizes capacity reduction continuously. However, compression executes finite capacity removals $\Delta E = E_{\text{active}}^{(l)}$ in discrete leaps. Evaluating macroscopic evictions logarithmically while governing granular micro-actions (Section 6) via linear ratios creates an inconsistent optimization hierarchy. Driven by this logarithmic discount, a greedy optimizer would view massive architectural deletions as artificially cheap. To align the macro-eviction cost with the linear micro-transition cost and ensure optimization stability, we apply the standard inequality $\ln(1+x) \le x$ for $x \ge 0$. Substituting $x = E_{\text{active}}^{(l)}/E_{\text{identity}}$ establishes a strict linear upper bound:

$$\mathcal{J}_{\text{layer}}(\Omega_a^{(l)}, \Omega_b^{(l)}) \triangleq N_{\text{active}}^{(l)} \frac{E_{\text{active}}^{(l)}}{E_{\text{identity}}}\,. \tag{82}$$

H.2.1 PARALLEL SURVIVAL CAPACITY E_{IDENTITY}

Flattening the tensor X into a vector $\boldsymbol{x} \in \mathbb{R}^{d_{\text{amb}}}$, we decompose it into d_{amb} independent identity operators $\mathcal{I}_k(\boldsymbol{x}) = \langle \boldsymbol{e}_k, \boldsymbol{x} \rangle \boldsymbol{e}_k = x_k \boldsymbol{e}_k$. Over the input distribution $P_{\mathcal{X}}$, the Hilbert-Schmidt capacity evaluates to the Root Mean Square (RMS) energy. Given $\|\boldsymbol{e}_k\|_2 = 1$:

$$\|\mathcal{I}_k\|_{\mathcal{H}} = \left(\mathbb{E}_{\boldsymbol{x} \sim P_{\mathcal{X}}}[\|x_k \boldsymbol{e}_k\|_2^2]\right)^{1/2} = \sqrt{\mathbb{E}_{\boldsymbol{x} \sim P_{\mathcal{X}}}[x_k^2]}\,. \tag{83}$$

Assuming $\boldsymbol{x}$ is conditioned by a preceding BN layer with scale γ_k and shift β_k, the expected energy evaluates to $\mathbb{E}[x_k^2] = \text{Var}(x_k) + (\mathbb{E}[x_k])^2 = \gamma_k^2 + \beta_k^2$. The aggregate capacity of the skip connection is therefore:

$$E_{\text{identity}} = \sum_{k=1}^{d_{\text{amb}}} \sqrt{\gamma_k^2 + \beta_k^2}\,. \tag{84}$$

For a stably normalized network $\gamma_k \approx 1, \beta_k \approx 0$, this naturally evaluates to the ambient physical dimension: $E_{\text{identity}} \approx d_{\text{amb}}$.

H.2.2 RESOLVING THE EXTINCTION DIVERGENCE

Block eviction projects a layer l to $\mathbf{0}$. Since PH-1 activations satisfy $\Psi(\mathbf{0}) = \mathbf{0}$, this ensures the entire sequential residual pathway vanishes $\mathcal{F}_{\text{res}} \to \mathbf{0}$, reducing the composite block mapping to $\mathcal{B} = \mathcal{F}(\mathcal{I})$. Axiom 2 (Connectivity Preservation) requires the transition cost to diverge $\mathcal{J} \to \infty$ as the state capacity approaches zero, penalizing graph disconnections.

If evaluated solely on the isolated layer $\Phi^{(l)}$, the terminal capacity would evaluate to zero, yielding an infinite penalty that would permanently prevent the optimizer from selecting an eviction operation. However, because the post-eviction mapping of the block is $\mathcal{F}(\mathcal{I})$, the network retains a minimum capacity $E_{\text{identity}} > 0$. Evaluating the integral over the macroscopic state $\Omega^{(l)} = (\Phi^{(l)}, \mathcal{I})$ naturally tracks this non-vanishing capacity. Since the differential change is driven entirely by the active layer $dE(\Omega) = dE(\Phi^{(l)})$, we have:

$$\mathcal{J}_{\text{layer}}^{(l)} = \int_{E_{\text{active}}^{(l)}}^{0} \frac{-N_{\text{active}}^{(l)}\, dE}{E + E_{\text{identity}}}\,. \tag{85}$$

As $E \to 0$, the denominator is bounded by $\lim_{E \to 0} E(\Omega) = E_{\text{identity}} > 0$. The extinction divergence is thus mitigated.

H.3 GENERALIZATION TO NON-RESIDUAL ARCHITECTURES

While E_{identity} structurally resolves the division-by-zero limit for residual networks, architectures without additive skip connections lack this architectural anchor. Examples include sequential VGG-style blocks, Inception modules, DenseNets, or intermediate expansion layers inside Transformer FFNs (prior to the global residual addition). In these architectures, evicting a block leaves no parallel identity mapping. This means the surviving capacity reaches zero.

To evaluate block eviction in these domains without encountering an infinite penalty $\mathcal{J} \to \infty$, we replace the dynamic composite denominator with a static historical constant: the block's initial, pre-pruning capacity $E_{\text{init}}^{(l)}$. The individual layer-wise cost becomes:

$$\mathcal{J}_{\text{layer}}(\Omega_a^{(l)}, \Omega_b^{(l)}) = N_{\text{active}}^{(l)} \frac{E_{\text{active}}^{(l)}}{E_{\text{init}}^{(l)}} \tag{86}$$

For a block containing multiple internal layers, the total macroscopic eviction cost is the linear sum of these bounds: $\mathcal{J}_{\text{evict}} = \sum_l \mathcal{J}_{\text{layer}}^{(l)}$.

From the perspective of pure functional analysis, E_{init} acts as a heuristic. It violates the Markov property of continuous state transitions (because the system must maintain memory of a pristine architectural state that no longer exists) and departs from Axiom 2 by freezing the denominator rather than tracking an architectural asymptote. However, it serves as an optimal proxy for three reasons:

- **Scale Invariance:** Using an arbitrary absolute constant C would yield $\mathcal{J} \propto E^{(l)}_{\text{active}}/C$, violating scale neutrality. The ratio $E^{(l)}_{\text{active}}/E^{(l)}_{\text{init}} \in [0, 1]$ normalizes the capacity reduction against the layer's own baseline. This ensures blocks of varying widths are penalized fairly based on the relative fraction of capacity destroyed.
- **Avoiding Normalization Collapse:** Normalizing by the current active capacity $E^{(l)}_{\text{active}}$ would collapse the cost to $\mathcal{J}_{\text{layer}} = N^{(l)}_{\text{active}}$. This artificially discounts wide, pristine layers, driving the greedy optimizer to evict them prematurely instead of targeting heavily pruned, redundant layers.
- **Dynamic Cost Decay:** The progressive encoder selects actions that minimize the Distortion Rate DR $= \mathcal{J}/\Delta P$. Under a mean-field assumption, active capacity scales linearly with surviving width $E_{\text{active}} \propto N_{\text{active}}$. By locking the denominator to E_{init}, the structural cost decays quadratically as the layer is pruned $\mathcal{J} \propto N_{\text{active}} E_{\text{active}} \propto N^2_{\text{active}}$. Because the physical parameter yield also scales linearly $\Delta P \propto N_{\text{active}}$, the overall distortion rate decreases linearly: DR $\propto N_{\text{active}}$. This guarantees that the relative cost of evicting a block naturally decreases as granular micro-actions deplete it.

Thus, while E_{identity} is the rigorously derived anchor for residual networks, E_{init} provides the necessary constraints to enable automated, progressive eviction in non-residual architectures.

I Reproducibility Protocols for Cross-Domain Transfer

This section details the experimental setup, algorithmic formulations, and hyperparameter grids necessary to reproduce the cross-domain transfer results presented in Section 9.2.

I.1 Task Construction and Data Partitioning

Tasks are dynamically constructed from the datasets. We perform 4 independent trials, with each trial comprised of 5 tasks to yield 20 distinct cross-domain transfer scenarios.

- **Source Tasks (CIFAR-100):** To ensure the source task requires learning cohesive structural features, the task comprises 20 classes sampled by selecting 4 random superclasses and utilizing all 5 of their constituent fine classes.
- **Target Tasks (SVHN):** The target tasks consist of 10-class classification utilizing all SVHN digits.

Both CIFAR-100 and SVHN datasets are partitioned into an 80% training split and a 20% validation split. The original test sets are preserved entirely for final evaluation. Prior to training, images are cast to floating-point tensors and scaled to $[0, 1]$ by dividing by 255. No data augmentation is applied during training.

I.2 Network Architecture

All experiments utilize a VGG-style 8-layer sequential baseline, modernized with BN. To prevent the final classification head from dominating the network's parameter footprint, we replace the classic flattened Dense layers with Global Average Pooling (GAP). This ensures that the model's overall capacity, and consequently ΔP evaluated by our compression algorithm, remains localized to the convolutional filters rather than spatial dense transitions. This prevents artificially skewed Distortion Rates (DR). The network consists of the following blocks:

1. **Block 1:** Conv2D (128 filters, 3x3) $\rightarrow$ BN $\rightarrow$ ReLU $\rightarrow$ MaxPool (2x2)
2. **Block 2:** Conv2D (256 filters, 3x3) $\rightarrow$ BN $\rightarrow$ ReLU $\rightarrow$ MaxPool (2x2)
3. **Block 3:** Conv2D (512 filters, 3x3) $\rightarrow$ BN $\rightarrow$ ReLU $\rightarrow$ Conv2D (512 filters, 3x3) $\rightarrow$ BN $\rightarrow$ ReLU $\rightarrow$ MaxPool (2x2)
4. **Block 4:** Conv2D (512 filters, 3x3) $\rightarrow$ BN $\rightarrow$ ReLU $\rightarrow$ Conv2D (512 filters, 3x3) $\rightarrow$ BN $\rightarrow$ ReLU $\rightarrow$ MaxPool (2x2)

5. **Transition:** GlobalAveragePooling2D
6. **Bottleneck:** Dense (512 neurons) $\rightarrow$ BN $\rightarrow$ ReLU
7. **Classification Head:** Dense (10 or 20 neurons) for multi-class logit output.

Note that BN layers are positioned *between* the affine transformations (Conv2D/Dense) and the ReLU non-linearities. To prevent redundancy and ensure capacity accounting, biases in the Conv2D and Dense layers preceding a BN layer are disabled.

I.3 Base Training Regimen

Both pre-training and fine-tuning utilize SGD with standard heavy-ball momentum of 0.9, a fixed batch size of 16, and a Sparse Categorical Cross-Entropy (CE) loss function.

- **Source Pre-training:** The model is trained from a random initialization for 100 epochs. To suppress mini-batch noise and ensure the parameters in the slack space converge to zero, the learning rate follows a Cosine Decay schedule starting at $\eta = 0.05$ and decaying to a near-zero $\alpha = 0.001$. Early stopping is deliberately disabled. To actively clear out the network's peripheral slack during this phase, an L_2 regularization penalty of 5×10^{-4} is applied to all multi-dimensional kernels as well as all 1D BN scale parameters γ.
- **Target Fine-Tuning:** All algorithms fine-tune for 30 epochs and use a *fixed* learning rate rather than a decay schedule. During target adaptation, the L_2 penalty is restricted to tensors with a rank greater than 1 to leave the calibrated 1D BN scale parameters immune to weight decay.

I.4 EWC Exact Empirical Fisher Calculation

To properly lock foundational source features for the EWC baseline, the empirical diagonal Fisher Information Matrix (FIM) is evaluated over the **source training dataset** (bypassing validation sets).

Note that the mathematical formulation calculates the expected squared gradient of the source CE loss at the *per-example* level before batch averaging:

$$\text{FIM}_i = \mathbb{E}_{\boldsymbol{x} \sim \mathcal{X}_{\text{source_train}}} \left[\left(\frac{\partial \mathcal{L}_{\text{CE}}}{\partial \boldsymbol{\theta}_i} \right)^2 \right] \tag{87}$$

Many naive EWC implementations incorrectly square the averaged mini-batch gradient, which misrepresents the true Fisher Information Matrix. By computing the per-example squared gradients directly, our evaluation avoids this theoretical pitfall. During target domain optimization, the target CE loss is augmented by the EWC penalty:

$$\mathcal{L}_{\text{total}} = \mathcal{L}_{\text{target_CE}} + \frac{\lambda}{2} \sum_{i \in \text{Backbone}} \text{FIM}_i (\boldsymbol{\theta}_i - \boldsymbol{\theta}_i^*)^2 \tag{88}$$

where λ is a global regularization hyperparameter, and $\boldsymbol{\theta}_i^*$ represents the model parameters of the optimal pre-trained source. Note that the summation is restricted to the backbone parameters; the newly initialized target classification head is excluded from the penalty.

I.5 Hyperparameter Tuning and Final Evaluation

To ensure a fair comparison, all algorithms undergo a hyperparameter pre-sweep prior to final evaluation. Hyperparameters are selected by maximizing the **H-Score**, defined as the Harmonic Mean of the validation accuracies on the target and source tasks:

$$\text{H-Score} = \frac{2 \cdot \text{Acc}_{\text{val}}^{\text{target}} \cdot \text{Acc}_{\text{val}}^{\text{source}}}{\text{Acc}_{\text{val}}^{\text{target}} + \text{Acc}_{\text{val}}^{\text{source}}} \tag{89}$$

This evaluation uses the *Validation Sets* of both SVHN (target) and CIFAR-100 (source) to prevent data leakage. To accelerate the pre-sweep, the target training set is artificially capped at $1,000$ samples during the tuning phase.

Rather than re-evaluating the grid for all 20 distinct transfer scenarios, the algorithm isolates the very first task (Task 0) as a representative sample, sweeps the grids to find the globally optimal parameters, and locks them in for the entire benchmark run. The sweep evaluates grids specific to each methodology:

- **DEFT:** A grid search across the target percentile $P \in \{60, 40, 30, 20\}$, and fixed fine-tuning learning rate $\eta \in \{0.04, 0.02, 0.01, 0.005, 0.001\}$.
- **EWC:** A grid search across the regularization strength $\lambda \in \{0.1, 0.5, 1.0, 5.0, 10.0, 20.0, 50.0\}$ and learning rate $\eta \in \{0.01, 0.005, 0.001, 0.0005, 0.0001\}$.
- **PEFT:** Unfreezes all 1D tensors (including layer biases and BN affine parameters) alongside the target head, while freezing all 2D/4D kernels. Optimized via a line search for $\eta \in \{0.01, 0.005, 0.0025, 0.001, 0.0005, 0.0001\}$.
- **Standard Full FT:** Unfreezes the entire network architecture. Optimized via a line search for $\eta \in \{0.02, 0.01, 0.005, 0.001, 0.0005, 0.0001\}$.
- **Standard Head-Only FT:** Freezes the entire backbone, optimizing only the final classification head. Optimized via a line search for $\eta \in \{0.04, 0.02, 0.01, 0.005, 0.001, 0.0005\}$.

Once optimal hyperparameters are secured, models are fine-tuned on the *full* SVHN training set for 30 epochs. Performance is tracked epoch-by-epoch using the SVHN Validation Set, and the algorithm identifies the best epoch yielding peak target validation accuracy. The final reported metrics in the main text are extracted at this best epoch by running the model against the unseen **SVHN Test Set** and **CIFAR-100 Test Set**.

Source Retention Protocol: To evaluate source retention, the original pre-trained source classification head is temporarily grafted back onto the fine-tuned backbone. Furthermore, a universal mask is applied to the grafted source head. This zeros out connections originating from upstream "slack" features $E > 0$ to ensure target-adapted weights do not corrupt the source logits. **Critically, the BN moving statistics μ, σ^2 are reverted**; they are unlocked from the target-adapted backbone and restored to their original source states to guarantee a fair evaluation.

J THEORETICAL GUARANTEES OF DEFT

Continual learning seeks a parameter update $\Delta\boldsymbol{\theta}$ that minimizes target risk while bounding the degradation of the source representation. Rather than analyzing non-convex loss landscapes, we abstract the network into continuous-functional operators and evaluate layer-wise distortion. Under HOPE, the capacity of neuron i is its Hilbert-Schmidt norm over $P_{\mathcal{X}}$:

$$\|f_i\|_{\mathcal{H}} = \|\boldsymbol{w}_{\text{out},i}\|_2 \sqrt{K(i,i)} \quad \text{where} \quad K(i,i) = \mathbb{E}_{\boldsymbol{x}\sim P_{\mathcal{X}}}\left[\Psi(y_i(\boldsymbol{x}))^2\right] . \tag{90}$$

The Setup: Core vs. Slack. DEFT partitions the network into two disjoint sets based on a capacity threshold τ:

- **Universal Core $\|f_i\|_{\mathcal{H}} > \tau$:** Highly active neurons carrying source knowledge. We freeze these $E_i = 0$ to prevent forgetting.
- **Slack Subset $\|f_j\|_{\mathcal{H}} \leq \tau$:** Weak, inactive neurons. We make these fully plastic $E_j = 1$ to learn the target task.

To prevent the target-driven updates of the slack subset from corrupting the core, DEFT applies a **structural mask** at initialization $t = 0$, severing all connections from upstream slack neurons to downstream core neurons.

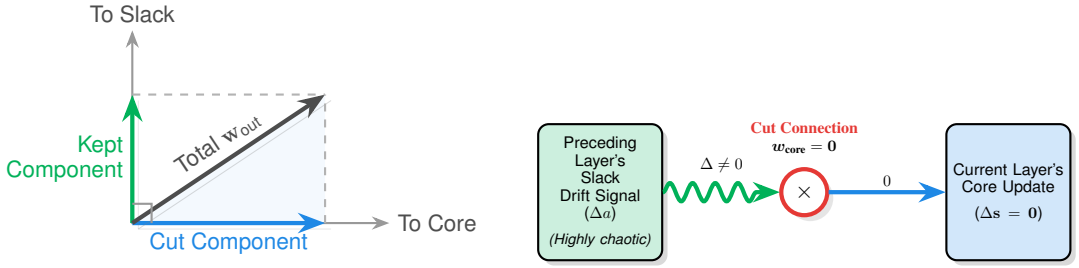
We establish the stability and plasticity of this setup through four main guarantees:

1. Bounded Initialization Shock (Theorem J.1). Severing the slack-to-core connections at $t = 0$ introduces a static error. **As visualized in Figure 2(a)**, the severed core-directed weights $\boldsymbol{w}_{\text{core},j}$ of a slack neuron j form a sub-vector of its total outgoing weights $\boldsymbol{w}_{\text{out},j}$. Thus, $\|\boldsymbol{w}_{\text{core},j}\|_2 \leq \|\boldsymbol{w}_{\text{out},j}\|_2$. Because we only sever connections from slack neurons (where $\|f_j\|_{\mathcal{H}} \leq \tau$), the initial distortion injected into the core is strictly bounded by $|\mathcal{N}^{(l)}_{\text{slack}}|\tau^{(l)}$.

2. Dynamic Decoupling (Theorem J.2). During training $t > 0$, slack neurons drift to learn the target task. Because the structural mask severs cross-connections at initialization $\boldsymbol{w}_{\text{core},j}^{(0)} = \mathbf{0}$ and the zero elasticity prevents gradient updates $\partial \boldsymbol{w}_{\text{core},j}/\partial t = \mathbf{0}$, these weights remain zero. **As illustrated in Figure 2(b)**, a changing signal multiplied by zero is zero; thus, target parameter drift cannot penetrate the core.

3. Freeing Space Safely (Proposition J.4). Freezing the core might leave insufficient parameter space for the target task. Deep networks often fragment a feature across M correlated neurons. Statically freezing them incorrectly locks redundant volume. Instead, DEFT compresses these M neurons into a single rank-1 parent operator. This preserves the feature while releasing the remaining $M-1$ children to the slack subset, freeing the parameter space $\mathbb{R}^{(M-1)\times c}$ for optimization. Because the optimal parent minimizes the Hilbert-Schmidt projection error, this structural distortion is upper-bounded by the sum of the children's initial capacities: $\delta_k \leq \sum_{m=1}^{M} \|f_m\|_{\mathcal{H}^{(l)}}$. (While we structurally guarantee this capacity release, we do not formally prove gradient descent finds a global optimum in this non-convex space).

4. Unified Cumulative Bound (Corollary J.5). Cutting connections and merging neurons across layers introduces multiple distortions. However, because Theorem J.2 guarantees zero interference during training, the network's total error does not compound exponentially. By the triangle inequality in $\mathcal{H}^{(l)}$, the global degradation in the function space is static and bounded by the linear sum of the severed slack connections and the merge projection errors.



(a) Vector Geometry (Theorem J.1) (b) Dynamic Signal Blocking (Theorem J.2)

Figure 2: Visual intuition for the theoretical bounds of DEFT. (a) Vector decomposition demonstrates the initialization shock is bounded by the slack capacities. (b) A changing signal multiplied by the zeroed structural mask ensures dynamic decoupling.

J.1 Algorithmic Axioms and Partitioning of Neurons

Here we formally define how DEFT partitions neurons into the peripheral slack and universal core. We also present a few axioms that merely mirror the mechanisms designated in the DEFT algorithm itself. These *axioms do not impose any extra assumptions or restrictions on DEFT beyond the entirety of the algorithm itself.*

Definition J.1 (Neurons Partition)**.** For any intermediate layer l containing a set of active neurons $\mathcal{I}^{(l)}$, a continuous capacity threshold $\tau^{(l)} > 0$ divides the layer into two disjoint functional subsets:

$$\mathcal{N}_{\text{core}}^{(l)} = \{i \in \mathcal{I}^{(l)} \mid \|f_i\|_{\mathcal{H}^{(l)}} > \tau^{(l)}\} \tag{91}$$

$$\mathcal{N}_{\text{slack}}^{(l)} = \{j \in \mathcal{I}^{(l)} \mid \|f_j\|_{\mathcal{H}^{(l)}} \leq \tau^{(l)}\} \tag{92}$$

Axiom 1 (The Algorithmic Structural Mask)**.** At initialization $t = 0$, DEFT prevents cross-subset interference by severing the projections from the upstream slack subset to the downstream core. For any weight $w_{u,j}$ from an upstream slack neuron $j \in \mathcal{N}_{\text{slack}}^{(l)}$ to a downstream core neuron $u \in \mathcal{N}_{\text{core}}^{(l+1)}$, the mask $\boldsymbol{M}$ enforces $w_{u,j} = 0$.

Axiom 2 (The Gradient Elasticity)**.** During target optimization $t > 0$, the parameter update for any parameter vector $\boldsymbol{\theta}_k$ belonging to a destination neuron k is governed by:

$$\frac{\partial \boldsymbol{\theta}_k}{\partial t} = E_k \cdot \nabla_{\boldsymbol{\theta}_k} \mathcal{L}_{\text{target}} \tag{93}$$

where $E_k \in \{0, 1\}$ is the elasticity multiplier. Slack neurons are fully plastic $E_j = 1 \, \forall j \in \mathcal{N}_{\text{slack}}$ and core neurons are strictly frozen $E_u = 0 \, \forall u \in \mathcal{N}_{\text{core}}$.

Axiom 3 (The Merged-Vessel Release)**.** When the continuous encoder collapses redundant neurons into a rank-1 parent operator, DEFT assigns full plasticity $E_j = 1$ to the structurally released child neurons.

J.2 Layer-to-Layer Bounding Framework

We now establish a rigorous bound on the structural distortion, decoupled entirely from target parameter drift.

Assumption 1 (Bipartite Separability)**.** Let the pre-activation y_u of a downstream core neuron $u \in \mathcal{N}_{\text{core}}^{(l+1)}$ be a linear combination of upstream activations $g_k(\boldsymbol{x}) = \Psi(y_k(\boldsymbol{x}))$ for $k \in \mathcal{I}^{(l)}$, governed by parameterized weights $w_{u,k}$ and unparameterized identity connections denoted by an adjacency indicator $\mathbb{I}_{u,k} \in \{0, 1\}$. We assume that for all upstream slack neurons $j \in \mathcal{N}_{\text{slack}}^{(l)}$, the network identity routing is strictly zero:

$$\mathbb{I}_{u,j} = 0 \quad \forall j \in \mathcal{N}_{\text{slack}}^{(l)}, \forall u \in \mathcal{N}_{\text{core}}^{(l+1)} \tag{94}$$

This ensures that the mask strictly dictates inter-subset signal flow.

Remark on Feed-Forward Architectures: For purely sequential architectures (e.g., standard MLPs or VGG), $\mathbb{I}_{u,k} \equiv 0$ globally. This assumption only becomes non-trivial in architectures with unparameterized residual connections (e.g., ResNets), where an identity mapping could bypass the mask if the architecture cross-wired the subsets.

Theorem J.1 (The Static Initialization Shock Bound)**.** Let $g_k(\boldsymbol{x}) \triangleq \Psi(y_k(\boldsymbol{x}))$, and let $\boldsymbol{s}_{\text{core}}(\boldsymbol{x}) \triangleq \sum_{k \in \mathcal{I}^{(l)}} \boldsymbol{w}_{\text{core},k} \, g_k(\boldsymbol{x})$ be the input to the downstream core. The initialization mask $\boldsymbol{M}$ introduces a distortion $\Delta \boldsymbol{s}_{\text{core}}^{\text{init}}(\boldsymbol{x}) \triangleq \sum_{j \in \mathcal{N}_{\text{slack}}^{(l)}} \boldsymbol{w}_{\text{core},j} g_j(\boldsymbol{x})$ bounded over $P_{\mathcal{X}}$ by:

$$\left\| \Delta \boldsymbol{s}_{\text{core}}^{\text{init}} \right\|_{\mathcal{H}^{(l)}} \leq |\mathcal{N}_{\text{slack}}^{(l)}| \, \tau^{(l)} \tag{95}$$

Proof. By Assumption 1, $\boldsymbol{s}_{\text{core}}(\boldsymbol{x})$ decomposes strictly over the parameterized upstream subsets:

$$\boldsymbol{s}_{\text{core}}(\boldsymbol{x}) = \sum_{i \in \mathcal{N}_{\text{core}}^{(l)}} g_i(\boldsymbol{x}) \, \boldsymbol{w}_{\text{core},i} + \sum_{j \in \mathcal{N}_{\text{slack}}^{(l)}} g_j(\boldsymbol{x}) \, \boldsymbol{w}_{\text{core},j} \tag{96}$$

where $\boldsymbol{w}_{\text{core},k}$ is the sub-vector of structural weights connecting upstream neuron k to the downstream core. By Axiom 1, DEFT enforces $\boldsymbol{w}_{\text{core},j} = \boldsymbol{0} \, \forall j \in \mathcal{N}_{\text{slack}}^{(l)}$. The resulting functional distortion evaluates to the severed projections:

$$\Delta \boldsymbol{s}_{\text{core}}^{\text{init}}(\boldsymbol{x}) = \sum_{j \in \mathcal{N}_{\text{slack}}^{(l)}} g_j(\boldsymbol{x}) \, \boldsymbol{w}_{\text{core},j} \tag{97}$$

Applying the triangle inequality in $\mathcal{H}^{(l)}$ yields:

$$\left\| \Delta \boldsymbol{s}_{\text{core}}^{\text{init}} \right\|_{\mathcal{H}^{(l)}} \leq \sum_{j \in \mathcal{N}_{\text{slack}}^{(l)}} \left\| \boldsymbol{w}_{\text{core},j} \right\|_2 \left\| g_j \right\|_{\mathcal{H}_{\text{in}}^{(l)}} \tag{98}$$

By Definition 90, $\|g_j\|_{\mathcal{H}_{\text{in}}^{(l)}} = \sqrt{K(j,j)}$. Furthermore, because $\boldsymbol{w}_{\text{out},j} = \boldsymbol{w}_{\text{core},j} \oplus \boldsymbol{w}_{\text{slack},j}$, its Euclidean norm bounds its sub-vectors: $\|\boldsymbol{w}_{\text{core},j}\|_2 \leq \|\boldsymbol{w}_{\text{out},j}\|_2$. Substituting these properties gives:

$$\left\|\Delta \boldsymbol{s}_{\text{core}}^{\text{init}}\right\|_{\mathcal{H}^{(l)}} \leq \sum_{j \in \mathcal{N}_{\text{slack}}^{(l)}} \|\boldsymbol{w}_{\text{out},j}\|_2 \sqrt{K(j,j)} = \sum_{j \in \mathcal{N}_{\text{slack}}^{(l)}} \|f_j\|_{\mathcal{H}^{(l)}} \tag{99}$$

By Definition J.1, the capacity of every slack neuron satisfies $\|f_j\|_{\mathcal{H}^{(l)}} \leq \tau^{(l)}$. Therefore, the functional distortion is strictly bounded by:

$$\left\|\Delta \boldsymbol{s}_{\text{core}}^{\text{init}}\right\|_{\mathcal{H}^{(l)}} \leq \sum_{j \in \mathcal{N}_{\text{slack}}^{(l)}} \tau^{(l)} = \tau^{(l)} \, |\mathcal{N}_{\text{slack}}^{(l)}| \tag{100}$$

□

J.2.1 Dynamic Decoupling

The dynamic decoupling of the core emerges as a direct consequence of Axiom 2.

Theorem J.2 (Dynamic Decoupling)**.** During target fine-tuning $t > 0$, the operators $f_i^{(l)}$ of all neurons within the core experience zero dynamic deviation:

$$\forall\, l\,,\ i \in \mathcal{N}_{\text{core}}^{(l)}\,;\ \left\| f_i^{(l),(t)} - f_i^{(l),(0)} \right\|_{\mathcal{H}^{(l)}} = 0 \tag{101}$$

Proof. We proceed by structural induction over the network layers.

Base Case: The input $\boldsymbol{x} \sim P_{\mathcal{X}}$ is a static anchor unmodified by optimization $\Delta \boldsymbol{x} = \mathbf{0}$.

Inductive Step: Assume that for layer l, the core functional mappings exhibit zero drift: $g_i^{(t)}(\boldsymbol{x}) = g_i^{(0)}(\boldsymbol{x})$ for all $i \in \mathcal{N}_{\text{core}}^{(l)}$.

The dynamic deviation of the signal injected into the downstream core of layer $l+1$ at time t is defined by $\Delta \boldsymbol{s}_{\text{core}}(\boldsymbol{x}) = \boldsymbol{s}_{\text{core}}^{(t)}(\boldsymbol{x}) - \boldsymbol{s}_{\text{core}}^{(0)}(\boldsymbol{x})$. Expanding this difference over the core and slack subsets yields:

$$\Delta \boldsymbol{s}_{\text{core}}(\boldsymbol{x}) = \sum_{i \in \mathcal{N}_{\text{core}}^{(l)}} \left(\boldsymbol{w}_{\text{core},i}^{(t)} g_i^{(t)}(\boldsymbol{x}) - \boldsymbol{w}_{\text{core},i}^{(0)} g_i^{(0)}(\boldsymbol{x}) \right) + \sum_{j \in \mathcal{N}_{\text{slack}}^{(l)}} \left(\boldsymbol{w}_{\text{core},j}^{(t)} g_j^{(t)}(\boldsymbol{x}) - \boldsymbol{w}_{\text{core},j}^{(0)} g_j^{(0)}(\boldsymbol{x}) \right) \tag{102}$$

For the first summation, Axiom 2 freezes downstream core weights $E_u = 0 \implies \boldsymbol{w}_{\text{core},i}^{(t)} = \boldsymbol{w}_{\text{core},i}^{(0)}$, and the inductive hypothesis guarantees $g_i^{(t)}(\boldsymbol{x}) = g_i^{(0)}(\boldsymbol{x})$. The first summation is therefore zero.

For the second summation, Axiom 1 severs cross-subset weights at initialization $\boldsymbol{w}_{\text{core},j}^{(0)} = \mathbf{0}$. Because $E_u = 0$, these structural weights receive zero gradient updates and remain zero $\boldsymbol{w}_{\text{core},j}^{(t)} = \mathbf{0}$. Thus, the second summation evaluates to zero irrespective of the slack drift $g_j^{(t)}$.

Consequently, $\Delta \boldsymbol{s}_{\text{core}}(\boldsymbol{x}) \equiv \mathbf{0}$. Since the downstream core parameters $\boldsymbol{\theta}_u \in \{\boldsymbol{w}_{\text{in},u}, b_u, \gamma_u, \beta_u\}$ are also frozen $E_u = 0$, the downstream core mappings experience zero drift $g_u^{(t)}(\boldsymbol{x}) = g_u^{(0)}(\boldsymbol{x})$. By induction, the target learning is decoupled from the core representation across all layers. □

Remark on Covariate Shift: Theorem J.2 guarantees the immutability of the core operators relative to the source distribution anchor $P_{\mathcal{X}}$. In practice, the network processes novel target data $\mathcal{D}_T$ during fine-tuning. Consequently, empirical activations flowing through the core will inevitably shift due to standard covariate shift, not parameter degradation. DEFT ensures that any shift in downstream representations is driven entirely by the target data itself, free from the compounding noise of parameter degradation. This preserves foundational knowledge as an uncorrupted lens for processing new domains.

Corollary J.3 (Cumulative Masking-Induced Bound)**.** Assuming a masking-based reduction without feature merging, the global structural distortion $D_{\text{global}}^{\text{mask}}$ evaluates as the cumulative sum of normalized layer-wise distortions. Normalizing the initialization shock at each layer by the downstream core capacity $E_{\text{core}}^{(l+1)} \triangleq \sum_{u \in \mathcal{N}_{\text{core}}^{(l+1)}} \|f_u\|_{\mathcal{H}^{(l+1)}}$, the total distortion is bounded by:

$$D_{\text{global}}^{\text{mask}} \triangleq \sum_{l=1}^{L-1} \frac{\left\| \Delta \boldsymbol{s}_{\text{core}}^{\text{init},(l+1)} \right\|_{\mathcal{H}^{(l)}}}{E_{\text{core}}^{(l+1)}} \leq \sum_{l=1}^{L-1} \left(\frac{|\mathcal{N}_{\text{slack}}^{(l)}| \tau^{(l)}}{E_{\text{core}}^{(l+1)}} \right) \tag{103}$$

This confirms that source task degradation is strictly bounded by the chosen capacity thresholds, circumventing the exponential scaling issues of global Lipschitz constants.

J.3 Dynamic Resolution of Redundancy via Bounded Trade-off

A static capacity threshold evaluates functional volume in isolation, which ignores the *redundancy trap* where networks fragment a feature across multiple correlated neurons. To compress this redundant volume and avoid incorrectly locking it, DEFT employs HOPE's `MERGE` operation.

Proposition J.4 (Bounded Distortion of Merging)**.** Consolidating M correlated neurons into a rank-1 parent neuron guarantees the release of target optimization volume while bounding the distortion of the source representation.

Proof. Suppose M correlated neurons $\{f_m\}_{m=1}^M$ are assigned to the core $E_m = 0$. By generating an optimal parent $f_p \in \mathcal{H}^{(l)}$, DEFT collapses M copies into 1 and frees $M-1$ child vessels. By Axiom 3, DEFT assigns these children full plasticity $E_j = 1$, successfully releasing the parameter space $\mathbb{R}^{(M-1)\times c}$ for optimization.

The distortion δ_k introduced by this operation is the composite projection error between the parent f_p and its children $\{f_m\}_{m=1}^M$. Because the optimal parent minimizes this distance in $\mathcal{H}^{(l)}$, the error is upper-bounded by the sub-optimal projection to the null operator $f_p \equiv \mathbf{0}$:

$$\delta_k \triangleq \min_{f_p} \left(\sum_{m=1}^{M} \|f_m - f_p\|_{\mathcal{H}^{(l)}}^2 \right)^{\frac{1}{2}} \leq \left(\sum_{m=1}^{M} \|f_m - \mathbf{0}\|_{\mathcal{H}^{(l)}}^2 \right)^{\frac{1}{2}} \tag{104}$$

Applying the standard ℓ_p-norm inequality $\|\boldsymbol{x}\|_2 \leq \|\boldsymbol{x}\|_1$, we bound this sub-optimal projection by the linear sum of the child capacities:

$$\delta_k \leq \sum_{m=1}^{M} \|f_m\|_{\mathcal{H}^{(l)}} \tag{105}$$

Thus, the algorithm safely releases massive target parameter volume while the distortion remains strictly bounded by the initial capacities. □

Corollary J.5 (The Unified Cumulative Bound)**.** By the triangle inequality in $\mathcal{H}^{(l)}$, the total perturbation injected into the core of layer $l+1$ is bounded by the sum of the missing slack projections (Theorem J.1) and the structural projection errors from $K^{(l)}$ merge operations. Because dynamic interference evaluates to zero via structural induction (Theorem J.2), the global normalized functional distortion is bounded as:

$$D_{\text{global}}^{\text{total}} \triangleq \sum_{l=1}^{L-1} \frac{\left\| \Delta \boldsymbol{s}_{\text{core}}^{(l+1)} \right\|_{\mathcal{H}^{(l)}}}{E_{\text{core}}^{(l+1)}} \leq \sum_{l=1}^{L-1} \left(\frac{|\mathcal{N}_{\text{slack}}^{(l)}| \tau^{(l)} + \sum_{k=1}^{K^{(l)}} \delta_k}{E_{\text{core}}^{(l+1)}} \right) \tag{106}$$

where δ_k is the bounded projection error of the k-th merge operation. Therefore, DEFT releases target parameter volume while bounding cumulative source degradation to an algorithmically verifiable constant.

K ALGORITHMS

Algorithm 1 HOPE Progressive Encoding Loop

Require: Pre-trained model $\mathcal{M}$, Target density ρ_{target}

Initialize CostManager()

▷ **PHASE 1:** $\mathcal{O}(N^2)$ **Initialization**

for each layer $L \in \mathcal{M}$ **do**
    InitializeCaches(L) ▷ Computes initial capacities and $\mathcal{O}(N^2)$ geometry
    Anchor initial uncompressed capacity $E_0^{(L)}$
end for

▷ **PHASE 2: True** $\mathcal{O}(1)$ **Greedy Scan**

while Density($\mathcal{M}$) $> \rho_{\text{target}}$ **do**
    $a^* \leftarrow$ null, $\text{DR}_{\min} \leftarrow \infty$
    ▷ *Evaluate all cached actions using live residual capacities*
    for each active layer L **do**
        $E_{\text{rem}} \leftarrow$ GetResidualCapacity(L) ▷ Dynamic tracking of layer shrinkage
        $N_{\text{live}} \leftarrow L$.ActiveCount() ▷ Live layer width for unbiased normalization
        for each candidate $c \in L$.ActiveActions() **do** ▷ Prunes, Merges, Evicts
            $\Delta P_{\text{static}} \leftarrow$ GetStaticPayoff(c) ▷ Action-specific static payoff
            $\mathcal{J}_{\text{cost}} \leftarrow$ ComputeDistortion(c, E_{rem}, N_{live}) ▷ $\mathcal{O}(1)$ scalar arithmetic
            $\text{DR} \leftarrow \mathcal{J}_{\text{cost}} / \Delta P_{\text{static}}$
            if $\text{DR} < \text{DR}_{\min}$ **then**
                $\text{DR}_{\min} \leftarrow \text{DR}$
                $a^* \leftarrow c$
            end if
        end for
    end for
    ▷ *Execution and Local Synthesis*
    ExecutePhysicalReduction($\mathcal{M}$, a^*)
    UpdateResidualCapacity($a^*.L$, $a^*.\mathcal{J}_{\text{cost}}$)
    ▷ $\mathcal{O}(1)$ *Local Recalculation (Strict intra-layer isolation)*
    if a^*.type == MERGE **then**
        $f_p \leftarrow a^*$.parent
        Cache.UpdatePruneCapacity(f_p) ▷ Update solitary capacity for new parent
        for each $f_{\text{neighbor}} \in a^*.L$.ActiveNeurons() **do**
            $\text{geo}_{\text{new}} \leftarrow$ PrecomputePairGeometry(f_p, f_{neighbor})
            Cache.Insert($(f_p, f_{\text{neighbor}})$, geo_{new})
        end for
    end if
    ▷ ZERO downstream recalculation required due to the static ΔP approximation
end while
return $\mathcal{M}$

Algorithm 2 HOPE Subroutine: UPDATERESIDUALCAPACITY

Require: Executed compression action a^* (contains action type, target layer(s) L, and targeted neurons)

$\epsilon \leftarrow 10^{-12}$ ▷ Numerical stability floor for capacity tracking
if a^*.type == PRUNE **then**
  $L \leftarrow a^*.L$
  $f_{\text{vic}} \leftarrow a^*$.victim
  $\Delta E \leftarrow$ Cache.GetPruneCapacity(f_{vic})
  $L.E_{\text{rem}} \leftarrow \max(L.E_{\text{rem}} - \Delta E, \epsilon)$
  L.RemoveNeuron(f_{vic})
else if a^*.type == MERGE **then**
  $L \leftarrow a^*.L$
  $f_i, f_j \leftarrow a^*$.children
  $f_p \leftarrow a^*$.parent
  ▷ Net flux: capacity of the extinguished children minus the new parent
  $\Delta E \leftarrow$ Cache.GetPruneCapacity(f_i) + Cache.GetPruneCapacity(f_j) − COMPUTECAPACITY(f_p)
  $L.E_{\text{rem}} \leftarrow \max(L.E_{\text{rem}} - \Delta E, \epsilon)$
  L.RemoveNeuron(f_j) ▷ Child j is purged; Child i serves as the vessel for f_p
else if a^*.type == EVICT **then**
  ▷ Eviction targets all internal reduction layers within the residual pathway
  for each internal layer $L \in a^*$.block **do**
    $L.E_{\text{rem}} \leftarrow \epsilon$ ▷ Pathway capacity forcibly collapsed
    L.ClearAllActiveNeurons() ▷ All internal operators are projected to **0**
  end for
end if

Algorithm 3 Dispersed Elastic Fine-Tuning (DEFT)

Require: Pre-trained source parameters θ_{src}, Source validation data $\mathcal{D}_s^{\text{val}}$, Target data $\mathcal{D}_t$, Percentile threshold P

▷ **PHASE 1: Capacity Evaluation & Core/Slack Partitioning**

Merge Redundancies: Use HOPE to compress highly correlated neurons into parent operators, freeing child vessels.
Compute Capacities: Calculate the functional capacity $\|f_i\|_{\mathcal{H}}$ for all active neurons.
Thresholding: Determine global capacity threshold τ corresponding to the P-th percentile.
for each neuron i in the network **do**
  if $\|f_i\|_{\mathcal{H}} > \tau$ **and** neuron i is active **then**
    $E_i \leftarrow 0$ ▷ **Universal Core:** Freeze to protect source knowledge
  else
    $E_i \leftarrow 1$ ▷ **Plastic Slack:** Freed vessels and weak neurons learn target
  end if
end for

▷ **PHASE 2: Consistency at Initialization (The Structural Mask)**

Initialize target weights $\theta_0 \leftarrow \theta_{\text{src}}$.
for each weight $w_{u,j}$ connecting an upstream neuron j to a downstream neuron u **do**
  if $E_j = 1$ **and** $E_u = 0$ **then**
    $M_{u,j} \leftarrow 0$ ▷ Sever cross-connections to prevent Slack drift from entering Core
  else
    $M_{u,j} \leftarrow 1$
  end if
end for
Apply Mask: $\theta_0 \leftarrow \boldsymbol{M} \odot \theta_0$

▷ **PHASE 3: Elastic Target Fine-Tuning (Dynamic Decoupling)**

$\theta_t \leftarrow \theta_0$, best_h_score $\leftarrow -1$
for epoch $\in \{1, \ldots,$ FINETUNE_EPOCHS$\}$ **do**
  for mini-batch $(\boldsymbol{x}, \boldsymbol{y}) \in \mathcal{D}_t^{\text{train}}$ **do**
    $\boldsymbol{g} \leftarrow \nabla_{\theta_t} \mathcal{L}_{\text{target}}(\theta_t; \boldsymbol{x}, \boldsymbol{y})$ ▷ Compute raw target gradients
    $\boldsymbol{g}_{\text{mod}} \leftarrow \boldsymbol{E} \odot \boldsymbol{g}$ ▷ Nullify gradients for frozen Core ($E_i = 0$)
    $\theta_t \leftarrow \text{OPTIMIZERSTEP}(\theta_t, \boldsymbol{g}_{\text{mod}})$ ▷ Only plastic Slack updates
  end for
  // Dual-Domain Evaluation (Harmonic Score Optimization)
  $\text{Acc}_{\text{tgt}} \leftarrow \text{EVALUATE}(\theta_t, \mathcal{D}_t^{\text{val}})$
  $\theta_{\text{src_eval}} \leftarrow \text{MASKSLACKDRIFT}(\theta_t, \theta_{\text{src}}, \boldsymbol{E})$ ▷ Restore pristine Core / Mask out Slack drift
  $\text{Acc}_{\text{src}} \leftarrow \text{EVALUATE}(\theta_{\text{src_eval}}, \mathcal{D}_s^{\text{val}})$
  H-Score $\leftarrow \frac{2 \cdot \text{Acc}_{\text{tgt}} \cdot \text{Acc}_{\text{src}}}{\text{Acc}_{\text{tgt}} + \text{Acc}_{\text{src}}}$
  if H-Score $>$ best_h_score **then**
    best_h_score $\leftarrow$ H-Score
    $\theta_{\text{best}} \leftarrow \theta_t$
  end if
end for
return θ_{best}