GEOMETRIC SUPERCHARGER: DATA-INFORMED HYPERBOLIC INITIALIZATION OF MLP-BASED SPARSE NEURAL NETWORKS

Anonymous authors
Paper under double-blind review

ABSTRACT

Sparse neural networks can substantially reduce computational and memory costs, yet their performance remains highly sensitive to the choice of sparse topology, particularly at extreme sparsity, where they struggle to outperform fully connected models. Existing sparse initialization methods are predominantly structure-driven and rarely conditioned on the data, motivating a central question: how can sparse connectivity be organized in a principled way that reflects the structure of the data rather than relying on predefined or heuristic topologies? Network science has shown that complex network topology can be characterized by an underlying latent geometry, helping to explain processes such as network routing, navigation, and dismantling. This suggests that, rather than imposing a predefined sparse topology, one can extract the latent geometry of the data and use it to organize network connectivity. Motivated by this connection, we introduce **Geometric Supercharger (GS)**, a data-informed framework for hyperbolic initialization of MLP-based sparse neural networks. GS extracts pairwise feature relationships using Pearson correlation and constructs a macro-scale geometric embedding through shortest-path-based pre-weighting and dimensionality reduction. The resulting geometry is then reused to reweight the network and refine the embedding, thereby *supercharging* it. This supercharged embedding is followed by meso-scale refinement based on community detection and angular reordering. The final angular coordinates encode data-informed node similarity and are used to generate a sparse hyperbolic topology. Building on previous findings that arbitrary, data-agnostic hyperbolic hierarchies can hinder sparse neural network training, we show that making the hierarchy data-informed turns the same geometric prior into a performance advantage. We evaluate Geometric Supercharger across four diverse tasks: image classification, physics-informed neural networks, reinforcement learning, and stock-market prediction. GS achieves state-of-the-art performance among all evaluated static sparse initialization methods in the ultra-sparse 99% regime and outperforms fully connected networks in the majority of experiments at 90% and 99% sparsity. These findings establish data-informed latent geometry as an important organizing principle for sparse MLP initialization and motivate a broader class of geometry-driven sparse neural networks.

1 INTRODUCTION

The increasing size of neural networks has made computational cost, memory consumption, and training efficiency central challenges in modern machine learning. Sparse neural networks offer a way to reduce these costs by replacing dense connectivity with a much smaller set of active connections. In particular, static sparse training initializes the network sparsely and maintains its sparse computation throughout training, reducing both training and inference costs. Beyond computational efficiency, recent theoretical work provides a broader motivation for sparse neural architectures. Poggio & Fraser (2024) argue that learnable functions can exhibit compositional sparsity, allowing complex functions to be represented as compositions of simpler functions depending on only small subsets of inputs. Subsequent work (Poggio, 2025) shows that efficiently computable functions admit compositionally sparse representations and corresponding sparse neural approximations.

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.

The main difficulty is that sparse neural networks are highly sensitive to *which* connections are retained. At very high sparsity, small differences in network organization can strongly affect performance. Existing static sparse initialization methods rely on a range of structural priors, including uniformly random connectivity, small-world and scale-free networks (Zhang et al., 2024), mathematically structured topologies such as Ramanujan graphs and RadiX-Nets (Lubotzky et al., 1988; Kepner & Robinett, 2019), and biologically inspired receptive-field or dendritic architectures (Zhang et al., 2025; Chavlis & Poirazi, 2025; Cerretti et al., 2025). This problem is further complicated by strong sensitivity to optimization hyperparameters as sparsity changes, as highlighted by SμPar (Dey et al., 2024). In addition, most sparse initialization methods remain largely *data-agnostic*: the same structural prior is generated independently of the particular dataset on which the network will be trained. Data-informed alternatives remain limited: CSTI (Zhang et al., 2024) selects connections from feature correlations, while SNIP (Lee et al., 2018) uses gradient-based connection saliency. Both use data to score individual connections, but neither explicitly models the global geometric organization of feature relationships.

Network science provides a framework for addressing this limitation. Complex networks can be represented in a latent geometric space, where connectivity reflects relationships between nodes on a latent manifold or metric space (Boguna et al., 2021). Latent geometry has emerged as an important principle for explaining network properties such as small-worldness, degree heterogeneity, and clustering, as well as processes including efficient information flow (Cannistraci & Muscoloni, 2022) and network dismantling (Adler et al., 2026b). Hyperbolic geometry is particularly suited to heterogeneous and hierarchical networks, with radial coordinates encoding hierarchy or popularity and angular coordinates encoding similarity (Krioukov et al., 2010). Many hyperbolic embedding methods have been developed to infer this latent geometry from observed networks (Muscoloni et al., 2017; García-Pérez et al., 2019; Balogh et al., 2025). This suggests a different approach to sparse neural initialization: instead of imposing a predefined topology, the relationships contained in the training data can first be represented geometrically, and this latent geometry can then be used to organize sparse neural connectivity.

Previous results reveal an apparent contradiction. Cannistraci-Hebb dynamic sparse training (CHTs) starts from a uniformly random sparse topology and, during training, progressively develops a hyperbolic hierarchical structure while achieving an *ultra-sparse advantage* (Zhang et al., 2024). In contrast, nBPSO imposes a data-agnostic hyperbolic hierarchical structure directly at initialization, which was found to perform worse than a uniformly random initialization (Adler et al., 2026a). This suggests that hyperbolic structure is not beneficial by itself; rather, its usefulness depends on how that structure is organized and aligned with the task.

As a result, we introduce **Geometric Supercharger (GS)**, a data-informed framework for hyperbolic initialization of MLP-based sparse neural networks. GS constructs a sparse feature network from pairwise correlations in the training data and uses its global structure to infer a latent geometric organization.

GS combines macro- and meso-scale embedding. At the macro scale, shortest-path-based preweighting (SPP) and non-centered Minimum Curvilinear Embedding (ncMCE) (Cannistraci et al., 2013) infer an initial angular geometry, which is reused to reweight the network and perform a second *supercharged* embedding. At the meso scale, community detection refines the angular organization. The resulting data-informed angular coordinates are supplied to nBPSO (Adler et al., 2026a) to generate hierarchical sparse bipartite layers.

Our four innovations are as follows. First, we formulate sparse MLP initialization as a problem of extracting data-informed latent geometry and using it to organize connectivity in sparse MLP-based neural networks. Second, we introduce **Geometric Supercharger (GS)**, a data-informed hyperbolic initialization framework. GS includes a multi-scale embedding procedure that combines shortest-path-based pre-weighting, ncMCE, a second geometry-informed *supercharging* pass, and community-based angular refinement. Third, we show that hyperbolic hierarchy is beneficial when its organization is informed by the data, reconciling prior results from dynamic and static sparse training. Fourth, we validate GS across four distinct tasks: image classification, physics-informed neural networks, stock-market prediction, and reinforcement learning. Across these settings, it outperforms all evaluated sparse initialization methods and achieves an ultra-sparse advantage by outperforming fully connected models. These findings establish data-informed latent geometry as

an important organizing principle for sparse MLP initialization and motivate a broader class of geometry-driven sparse neural networks.

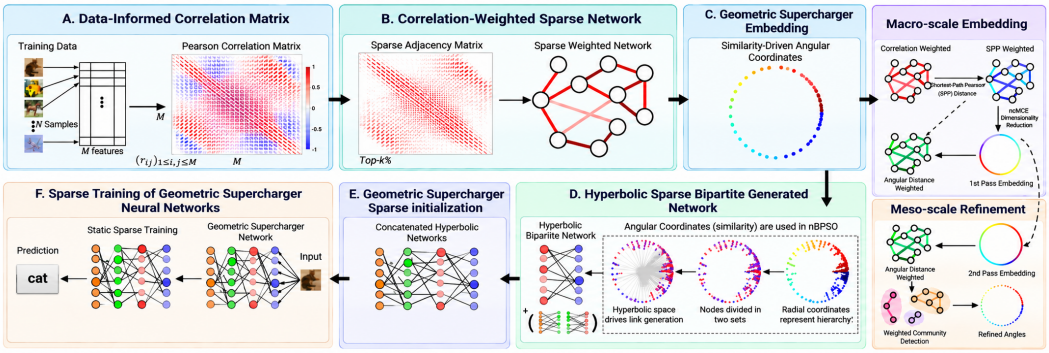

Figure 1: Overview of Geometric Supercharger. A: N calibration samples are represented by M features, and pairwise Pearson correlations are computed to form an $M \times M$ correlation matrix, where r_{ij} denotes the correlation between features i and j. **B**: The correlation matrix is sparsified by retaining the top-$k\%$ entries by absolute correlation magnitude to match the target MLP sparsity, where k is the target density. The retained correlations define a sparse weighted feature network and are converted into distances. **C**: Geometric Supercharger embeds this network through macro-scale embedding and meso-scale refinement. The macro-scale stage computes all-pairs shortest paths, compares shortest-path profiles using Pearson distance, and applies ncMCE to obtain angular coordinates. These coordinates are then used to reweight the network according to angular distance for a second embedding pass, supercharging the macro-scale embedding. The meso-scale stage applies angular-distance-weighted Louvain community detection and CLOVE-based travelling-salesman community ordering. The final angular coordinates encode feature similarity. **D**: These angular coordinates are used as similarity coordinates in nBPSO, while radial coordinates encode hierarchy through the nBPSO growth process. Two node sets are generated and connected according to hyperbolic distance to form a sparse bipartite network. **E**: The resulting bipartite topology is replicated or generated across consecutive layers to construct a multilayer Geometric Supercharger sparse neural network. **F**: The resulting network is then trained.

2 Related Work

Hyperbolic embedding of complex networks. Hyperbolic space provides a natural representation for complex networks because radial coordinates can encode node hierarchy or popularity, while angular coordinates capture node similarity: nodes with similar connectivity patterns tend to occupy nearby angular positions. Recovering the latent angular coordinates from an observed network is a central problem in hyperbolic network embedding. Some approaches are model-based. HyperMap (Papadopoulos et al., 2014) reconstructs the latent hyperbolic coordinates of a network by replaying its growth under the E-PSO model, which is a modification of the PSO model that considers external links, meaning the links that are formed between a new and an existing node. For each arriving node, it selects the coordinates that maximize the likelihood of the observed connectivity. HyperMap-CN (Papadopoulos et al., 2015) extends this idea by additionally considering the number of common neighbors between node pairs. An alternative family of methods avoids direct maximum-likelihood optimization. Laplacian-based Network Embedding (LaBNE) (Alanis-Lobato et al., 2016) uses spectral manifold learning to obtain an efficient data-driven approximation of the latent geometry. Coalescent Embedding (Muscoloni et al., 2017) proposed a class of parameter-free, topology-based machine learning methods for nonlinear dimension reduction. It termed the idea "angular coalescence", which is the idea that node angular coordinates of the hyperbolic model can be directly approximated in the two- or three-dimensional embedding space according to a persistent node aggregation pattern. Mercator (García-Pérez et al., 2019) combines machine-learning-based dimensionality reduction with maximum-likelihood refinement under the PSO model. Other methods explicitly exploit community structure to improve angular organization. LPCS (Wang et al., 2016b) first detects a hierarchy of communities, recursively orders them according to inter-community relationships, assigns each community an angular sector, and places its nodes within that sector. HMCS (Wang et al., 2019) and CHM (Wang et al., 2016a) follow a similar community-sector principle, using measures of community proximity or intimacy to determine the angular ordering of communities before assigning node positions within those sectors. More recently, CLOVE (Balogh et al., 2025) recursively detects communities and arranges them hierar-

chically down to the level of individual nodes; critically, the angular ordering of communities at each level is formulated as a Travelling Salesman Problem (TSP), allowing strongly related communities to be positioned close to one another on the hyperbolic disk.

Geometric network models. Random geometric graphs generate networks defined by a latent geometry. In the bipartite setting, the AB random geometric graph (AB-RGG) model (Iyer & Yogeshwaran, 2012) places the two node sets according to independent point processes in Euclidean space and connects nodes from opposite sets whenever their distance falls below a prescribed threshold. Other latent-space models showed that complex network structure can be well represented in hyperbolic space, with negative curvature and exponential growth of space. There, radial coordinates capture node popularity and angular coordinates encode similarity. Building on this principle, the Popularity-Similarity Optimization (PSO) model (Papadopoulos et al., 2012) introduced a generative mechanism for unipartite networks in hyperbolic space, where connection probabilities depend on hyperbolic distance and naturally produce heterogeneous degree distributions and high clustering. The nonuniform PSO (nPSO) model (Muscoloni & Cannistraci, 2018) extended PSO by sampling angular coordinates from nonuniform distributions, introducing explicit and tunable community organization. PSO was subsequently generalized to bipartite networks (Filho & O'Neale, 2020). The nonuniform Bipartite Popularity-Similarity Optimization (nBPSO) model (Adler et al., 2026a) extends nPSO to the bipartite case and further develops the framework by jointly controlling density, degree-distribution profiles, clustering, and community organization. It additionally introduces mechanisms for controlling community compactness and hierarchical organization and can generate consecutive bipartite layers for sparse neural network initialization. Unlike embedding methods that infer coordinates for an already existing network in a geometric space, PSO, nPSO, and nBPSO are generative models that construct network topology directly from the latent geometry.

Sparse neural network initialization. Static sparse training differs from pruning in that the network is initialized and trained in sparse form from the start, whereas pruning typically begins from a fully connected network and removes connections after training. Consequently, static sparse training can reduce both training and inference costs, while post-training pruning primarily reduces inference costs. Existing sparse initialization methods span several families. Network-science-based approaches include Bipartite Small-World (BSW) and Bipartite Scale-Free (BSF) topologies (Zhang et al., 2024), which adapt the classic Watts-Strogatz (WS) and Barabási-Albert (BA) generative network models to bipartite networks. The BSW network produces a network with clustering and the small-world property, while the BSF network is scale-free (with a power-law degree distribution) and also has the small-world property. Bipartite Receptive Field (BRF) (Zhang et al., 2025) instead mimics the spatially restricted receptive fields in biological brains. Other biologically inspired approaches include Dendritic Artificial Neural Networks (dANN) (Chavlis & Poirazi, 2025) and the Dendritic Network Model (DNM) (Cerretti et al., 2025), which model structured dendritic connectivity and restricted input sampling. Ramanujan graphs (Lubotzky et al., 1988) provide sparse expander topologies with strong spectral properties, while RadiX-Nets (Kepner & Robinett, 2019) use mixed-radix numeral systems and Kronecker products to construct sparse topologies with guaranteed path connectivity and symmetry. In contrast to these largely data-agnostic approaches, Correlated Sparse Topological Initialization (CSTI) (Zhang et al., 2024) is data-informed, selecting sparse connectivity from the strongest pairwise feature correlations. SNIP (Lee et al., 2018) uses weight-gradient saliency on calibration data to identify important connections before training and can therefore be used as a sparse initialization method. However, whether SNIP should be regarded as a strictly sparse initialization method is debatable, since constructing the mask still requires evaluating the initially dense network on calibration data and computing gradients with respect to all candidate connections.

3 GEOMETRIC SUPERCHARGER

Our first innovation, inspired by network science and complex systems theory, is to extract the latent geometry of the data and use it to initialize sparse MLP-based sparse neural networks. Building on this idea, we introduce our second innovation, **Geometric Supercharger (GS)**, a data-informed hyperbolic initialization framework. GS constructs a sparse topology by first extracting a latent geometric organization from the training data and then using this geometry to generate sparse bipartite neural layers. The procedure is detailed in Figure 1 and consists of four stages: (i) construction of a

data-informed feature network, (ii) macro-scale geometric embedding, (iii) meso-scale community refinement, and (iv) generation of the final hyperbolic sparse topology. The entire initialization is performed once before training. Detailed ablation experiments for each component are presented in Appendix A and Table 5.

3.1 DATA-INFORMED FEATURE NETWORK

Given N training samples represented by M features, we compute the Pearson correlation coefficient between every pair of features, producing an $M \times M$ correlation matrix (Zhang et al., 2024). Since both positive and negative correlations indicate a relationship between features, we use their absolute values. Self-correlations are removed, and the strongest correlations are retained according to the target sparsity of the final network. The retained correlations define a sparse weighted feature network, where each node represents a feature and each edge represents a strong relationship between two features. The retained correlation weights are subsequently converted into distances for geometric embedding.

3.2 MACRO-SCALE EMBEDDING

The macro-scale component aims to recover the global similarity organization of the feature network. It combines shortest-path Pearson pre-weighting (SPP) with non-centered Minimum Curvilinear Embedding (ncMCE) (Cannistraci et al., 2013; 2010).

Shortest-path Pearson pre-weighting. We first compute the shortest-path distances between all pairs of nodes in the sparse feature network. Each node i is therefore represented by a shortest-path profile containing its distances to every other node. We then compare these profiles using Pearson distance. Two nodes are considered geometrically similar when they have similar distance relationships to the rest of the network. The resulting shortest-path Pearson distances are mapped back onto the existing edges of the sparse network as new edge weights. We refer to this procedure as shortest-path Pearson pre-weighting (SPP). This step incorporates global network information into each existing connection. The weight of an edge reflects both the original pairwise relationship between two nodes and how similarly the two nodes are positioned relative to the entire network.

ncMCE embedding. The SPP-weighted network is then embedded using ncMCE, which estimates nonlinear relationships by considering curvilinear distances along the minimum spanning tree of the data, providing a parameter-free approximation of nonlinear distances (Cannistraci et al., 2013). Following the angular-coalescence principle of Coalescent Embedding (Muscoloni et al., 2017), the two-dimensional ncMCE coordinates are converted into angular coordinates. In the hyperbolic representation, these angular coordinates encode node similarity: nodes positioned closer in angle are interpreted as more similar.

Supercharged embedding. The first embedding is subsequently fed back into the procedure. We compute the angular distance between connected nodes and use these distances to reweight the existing network. SPP and ncMCE are then applied a second time to this geometry-informed network. We refer to this second pass as *supercharging*: the geometry inferred by the first embedding is reused to reweight the network before the subsequent embedding. The term is inspired by a mechanical supercharger, which uses energy from the engine itself to increase the density of the incoming air for the next combustion cycle. Analogously, GS reuses the geometry produced by the first embedding to reweight the network before the second pass, concentrating the representation around geometrically consistent relationships and reducing the influence of potentially spurious links.

3.3 MESO-SCALE REFINEMENT

The macro-scale embedding captures global relationships between nodes, but does not explicitly enforce the mesoscale organization associated with network communities. We therefore refine the second-pass embedding using the community structure of the angular-distance-weighted network. Weighted Louvain community detection is applied to the angular-distance-weighted network. We then adopt the TSP-based community ordering of CLOVE (Balogh et al., 2025), arranging strongly related communities closer in angular space. Unlike standard CLOVE, community detection is performed on the geometry-weighted network, motivated by previous evidence that geometry-informed

weighting improves community detection (Muscoloni et al., 2017). The resulting refinement enables the final angular coordinates to incorporate both macro-scale network relationships and meso-scale community structure.

3.4 HYPERBOLIC SPARSE TOPOLOGY GENERATION

The final GS angular coordinates are used as data-informed similarity coordinates for the nonuniform Bipartite Popularity-Similarity Optimization (nBPSO) model (Adler et al., 2026a). In the hyperbolic disk, angular coordinates determine similarity, while radial coordinates represent the hierarchical or popularity component introduced by the nBPSO growth process. Connectivity between nodes from opposite sets is determined according to their hyperbolic distance.

nBPSO generates sparse bipartite networks at the requested density, which are used as connectivity masks for selected fully connected weight matrices. We use $T = 0$ by default, such that edge formation follows the inferred hyperbolic geometry as strongly as possible by favoring the geometrically closest nodes. We replicate the generated topology across layers because the ablation in Table 6 showed no meaningful benefit from independent generation. This is expected since all layers share the same data-informed angular coordinates, while only the radial coordinates vary slightly, leaving the underlying geometric organization largely unchanged. The degree-distribution exponent is fixed to $\gamma = 3$. Values in the range $2 < \gamma < 3$ are characteristic of scale-free networks with power-law degree distributions, with smaller γ producing heavier tails and stronger hub formation. Although the weaker hierarchical organization at $\gamma = 3$ was found to be optimal, the hierarchy remained important: as shown in Table 7, rewiring either one or both bipartite node sets to disrupt the power-law degree distribution consistently reduced performance. Across all experiments and initialization methods, the final output layer remains dense.

4 ULTRA-SPARSE ADVANTAGE OF GEOMETRIC SUPERCHARGER

Our first key observation, which motivates GS and constitutes our third innovation, is shown in Figure 2: data-agnostic nBPSO performs better as the hyperbolic hierarchy weakens, whereas data-informed GS performs better as the network more closely follows the underlying hyperbolic hierarchical geometry. In nBPSO, performance improves as the temperature T increases and the topology becomes less constrained by the underlying hyperbolic geometry, indicating that a data-agnostic hyperbolic hierarchy is detrimental to static sparse training. GS reverses this trend: performance decreases as T increases, with the best results obtained when the generated topology most closely follows the inferred data-informed geometry. This is the first indication that the benefit of hyperbolic organization depends on aligning the geometry with the data.

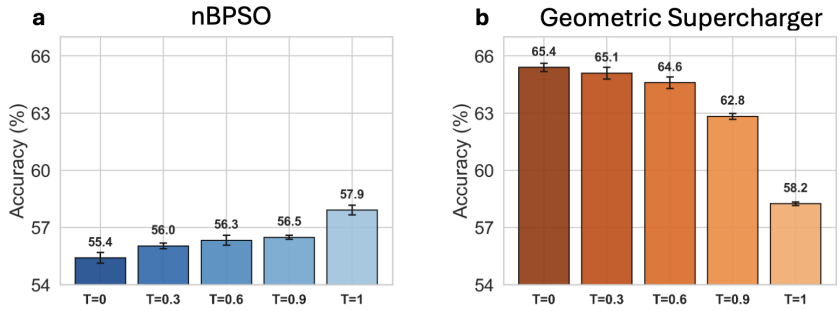

Figure 2: Test accuracy on CIFAR-10 for MLPs (99% sparsity) initialized with nBPSO (blue) and Geometric Supercharger (GS) (orange) at varying temperatures T and trained using static sparse training. Both methods use $\gamma = 3$; nBPSO uses $C = 0$ (no explicit communities). Colors range from dark blue/orange for initializations with a lower temperature (respecting the underlying geometry) to light blue/orange for initializations with a higher temperature (more random). Each bar is the mean over three seeds; error bars show the standard error.

4.1 GEOMETRIC SUPERCHARGER OUTPERFORMS ALL OTHER SPARSE INITIALIZATION METHODS.

Table 1 compares GS with a fully connected network and a broad set of data-agnostic and data-informed sparse initialization methods at 99% sparsity under static sparse training. GS achieves

the highest test accuracy on all three image classification benchmarks. Relative to the strongest competing sparse method, CSTI, GS improves accuracy by 3.45 percentage points (p.p.) on CIFAR-10, 3.56 p.p. on CIFAR-100, and 2.05 p.p. on ImageNet32-200. GS also exceeds the fully connected baseline by 3.34, 7.50, and 3.85 p.p., respectively, demonstrating an ultra-sparse advantage across all three datasets. This is the key result for our fourth innovation.

Importantly, the fully connected baseline was given an additional hyperparameter search over batch size and learning rate, whereas all sparse methods used the same fixed training configuration. Despite this additional tuning, the fully connected model remained below GS on every dataset.

Table 1: Image classification test accuracy on CIFAR-10, CIFAR-100, and ImageNet32-200 under 99%-sparse static training. Each classifier is a four-layer MLP with three hidden layers of 3072 neurons. A parameter search was conducted over batch sizes $\in \{32, 64, 128\}$ and learning rates $\in \{0.025, 0.01, 0.001\}$ only for the fully connected network, with the best hyperparameters selected based on validation performance. All sparse methods use a batch size of 32 and a learning rate of 0.025. Each value is the mean over three seeds $\pm$ SEM; bold denotes the best-performing model for each dataset.

	Static Sparse Training		
	CIFAR-10	**CIFAR-100**	**ImageNet32-200**
FC	62.05 ± 0.13	30.68 ± 0.16	9.71 ± 0.40
nBPSO	57.63 ± 0.22	30.37 ± 0.08	10.47 ± 0.16
Uniformly Random	57.65 ± 0.11	30.45 ± 0.23	10.15 ± 0.06
BSW	59.52 ± 0.16	33.17 ± 0.11	10.61 ± 0.20
BRF	55.60 ± 0.40	27.66 ± 0.17	8.87 ± 0.16
Ramanujan	57.39 ± 0.16	30.79 ± 0.04	9.49 ± 0.27
RadiX-Nets	58.15 ± 0.40	31.94 ± 0.19	9.98 ± 0.02
dANN-R	53.99 ± 0.26	27.41 ± 0.12	8.08 ± 0.10
DNM	58.02 ± 0.13	30.78 ± 0.36	10.19 ± 0.13
SNIP	55.73 ± 0.37	28.43 ± 0.39	10.85 ± 0.14
CSTI	61.94 ± 0.08	34.62 ± 0.29	11.51 ± 0.11
GS (Ours)	$\mathbf{65.39 \pm 0.22}$	$\mathbf{38.18 \pm 0.16}$	$\mathbf{13.56 \pm 0.30}$

In addition, Figure 3 shows that the effect of hyperbolic hierarchy becomes clearer as task complexity increases. The figure reports $\Delta\text{Accuracy} = \text{Acc}_{T=0} - \text{Acc}_{T=1}$, such that negative values indicate an advantage for the more randomized $T = 1$ initialization, whereas positive values indicate an advantage for the geometry-preserving $T = 0$ initialization. For data-agnostic nBPSO, $T = 0$ increasingly underperforms $T = 1$ from MNIST to CIFAR-100, producing increasingly negative differences. GS shows the opposite trend: the data-informed $T = 0$ initialization increasingly outperforms $T = 1$, producing increasingly positive differences as task complexity increases. The differences are statistically significant on CIFAR-10 and CIFAR-100, but negligible and not statistically significant on MNIST and EMNIST. This indicates that the effect of the hyperbolic hierarchy, and particularly whether it is aligned with the data, becomes increasingly important on more complex tasks.

We use 99% sparsity for image classification because the comparatively large networks still retain enough active connections at this extreme sparsity level. For the PINN experiments, although the networks are smaller, the low-dimensional coordinates are first transformed into higher-dimensional Fourier-feature representations, which encode and distribute the original coordinate information across a richer and more redundant feature space. This allows the network to remain effective under much stronger sparsification, and we therefore also use 99% sparsity for PINNs. In contrast, the stock-prediction and reinforcement-learning models are networks operating directly on low-dimensional inputs. We therefore use 90% sparsity for these tasks to avoid reducing the absolute number of active connections to a level that substantially degrades performance.

4.2 Physics-Informed Neural Networks (PINN)

We next evaluate whether the advantage of GS extends beyond image classification to physics-informed neural networks, further testing the generality of our approach. At 99% sparsity, GS outperforms the fully connected baseline on Helmholtz and inverse Brinkman (Table 2), showing that the benefit of data-informed geometric initialization can extend to both forward and inverse physics-informed learning.

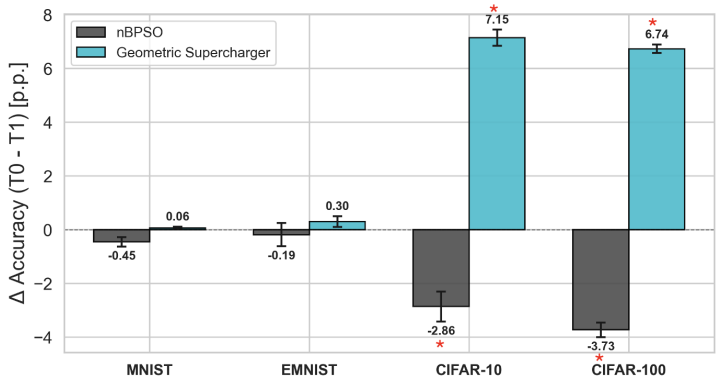

Figure 3: Average test accuracy difference between MLPs (99% sparsity) initialized with $T = 0$ and $T = 1$, where $\Delta Accuracy = Acc_{T=0} - Acc_{T=1}$ in percentage points (p.p.), for nBPSO ($\gamma = 2$) and Geometric Supercharger (GS) ($\gamma = 3$) across image-recognition tasks ordered by complexity. All models were trained using static sparse training. Each bar is the mean over six seeds; error bars show the standard error. Asterisks indicate statistical significance ($p < 0.01$) based on a one-sided permutation test.

Table 2: Physics-informed learning on the two PDE tasks: forward solution of the two-dimensional Helmholtz equation and parameter inference for the one-dimensional inverse Brinkman problem. Each task uses two fixed Fourier-feature branches followed by a shared three-hidden-layer MLP and a dense scalar output. Per branch, the architecture is $2m \to w \to w \to w$; (m, w) is $(36, 62)$ for Helmholtz and $(25, 22)$ for inverse Brinkman. The two width-w branch outputs are concatenated before the output layer. Values are mean error $\pm$ SEM across five seeds (lower is better). Helmholtz reports relative L_2 solution error; inverse Brinkman reports the mean relative error of the inferred ν_e and K. Bold indicates that Geometric Supercharger at 99% sparsity outperforms the corresponding fully connected (FC) baseline.

	Helmholtz	**Inverse Brinkman**
FC	0.622 ± 0.036	1.190 ± 0.369
GS	$\mathbf{0.020 \pm 0.012}$	$\mathbf{0.284 \pm 0.189}$

4.3 Stock-price prediction

Table 3 evaluates GS on next-day stock-price prediction using 90%-sparse MLPs. GS achieves a lower mean test MSE than the fully connected baseline for 11 of the 17 evaluated securities with only 10% of connections active. This advantage is observed across securities with substantially different historical lengths and price dynamics, indicating that GS can also transfer to temporal regression problems. Prior studies using the same underlying dataset have evaluated statistical, deep learning, recurrent, ensemble, and Transformer-based forecasting approaches for next-day or multi-step stock-price prediction (Li et al., 2023; Wang et al., 2022; Mochurad & Dereviannyi, 2024). Although direct comparison is not possible because of differences in experimental setup, the errors obtained by our 90%-sparse MLPs are of a comparable order to those reported for substantially denser and more complex forecasting models, further supporting our fourth innovation.

4.4 Reinforcement Learning

Finally, we evaluate GS on the CartPole-v1 reinforcement-learning task. As shown in Table 4, the 90%-sparse GS network achieves higher mean evaluation returns than the fully connected model at hidden-layer widths $h = 64$ and $h = 128$. For $h = 64$, the mean return increases from 106.3 for FC to 192.4 for GS, while for $h = 128$ it increases from 140.1 to 262.3.

5 Conclusion

We first introduce the idea of estimating the latent geometry of the data to initialize an advantageous, data-informed sparse topology for MLP-based sparse neural networks before training.

We then introduce **Geometric Supercharger (GS)**, a data-informed framework for hyperbolic initialization of MLP-based sparse neural networks. Rather than imposing a predefined sparse topology, GS extracts feature relationships from the training data and uses them to infer a latent geomet-

Table 3: Next-day closing-price regression on 17 publicly traded stocks. The task uses the preceding 20 standardized daily closing prices to predict the next standardized close. Each regressor is a four-layer MLP with three hidden layers of 40 neurons. Values are mean test MSE ($\times 10^{-2}$) $\pm$ SEM across three seeds for fully connected (FC) and 90%-sparse Geometric Supercharger (GS) models; bold denotes the lower mean for each asset.

Stock	FC	GS
AA	2.28 ± 0.14	$\mathbf{1.57 \pm 0.22}$
ARNC	2.33 ± 0.08	$\mathbf{2.03 \pm 0.44}$
BA	0.27 ± 0.03	$\mathbf{0.26 \pm 0.05}$
CNP	$\mathbf{0.56 \pm 0.02}$	0.79 ± 0.22
DD	0.66 ± 0.00	$\mathbf{0.60 \pm 0.09}$
DIS	1.27 ± 0.14	$\mathbf{0.37 \pm 0.21}$
DXC	0.41 ± 0.01	$\mathbf{0.32 \pm 0.05}$
F	1.80 ± 0.09	$\mathbf{1.62 \pm 0.44}$
GE	3.81 ± 0.95	$\mathbf{1.56 \pm 0.67}$
HON	0.34 ± 0.01	$\mathbf{0.18 \pm 0.02}$
HPQ	$\mathbf{2.59 \pm 0.40}$	2.84 ± 1.86
IP	1.36 ± 0.13	$\mathbf{0.65 \pm 0.17}$
JNJ	2.15 ± 0.30	$\mathbf{0.46 \pm 0.14}$
MCD	$\mathbf{0.40 \pm 0.08}$	0.41 ± 0.13
MRO	$\mathbf{1.20 \pm 0.14}$	1.71 ± 0.66
XLP	$\mathbf{3.25 \pm 0.12}$	5.75 ± 0.16
YRCW	$\mathbf{1.20 \pm 0.08}$	2.29 ± 0.66

Table 4: Reinforcement-learning control on the CartPole-v1 environment. From each four-dimensional state, the task is to estimate the values of the two discrete actions and learn to balance the pole. Each action-value network is a three-layer MLP with two hidden layers of width $H \in \{64, 128\}$. The input and output layers are dense; the $H \to H$ layer is fully connected (FC) or initialized by Geometric Supercharger at 90% sparsity (GS). Values are mean evaluation return $\pm$ SEM across ten seeds; higher is better.

	$h = 64$	$h = 128$
FC	106.3 ± 32.9	140.1 ± 43.3
GS	$\mathbf{192.4 \pm 52.1}$	$\mathbf{262.3 \pm 50.1}$

ric organization. A macro-scale embedding combines shortest-path Pearson pre-weighting, ncMCE, and a second geometry-informed *supercharging* pass, while a meso-scale refinement incorporates community structure. The resulting angular coordinates provide data-informed similarity coordinates for nBPSO, which generates the final hierarchical sparse topology.

Using the GS framework, we show that hyperbolic structure alone is insufficient: data-agnostic hyperbolic initialization is detrimental, whereas aligning the geometry with the data reverses this effect, particularly on more complex tasks.

Across four diverse tasks, GS consistently demonstrates the benefit of data-informed geometric initialization. At 99% sparsity, GS achieves the highest accuracy among all evaluated sparse initialization methods on CIFAR-10, CIFAR-100, and ImageNet32-200, while also outperforming the fully connected models. Beyond image classification, GS outperforms the fully connected baseline in both PINN tasks, 11 of 17 stock-price prediction tasks, and both reinforcement-learning network sizes.

Overall, these findings establish data-informed latent geometry as an effective organizing principle for sparse MLP initialization and motivate a broader class of geometry-driven sparse neural networks.

REFERENCES

Thomas Adler, Junda Chen, Wenqi Gu, Ziheng Liao, and Carlo Vittorio Cannistraci. Hyperbolic bipartite network model with communities impacts sparse neural network training. *Preprints*, September 2026a. doi: 10.20944/preprints202609.2187.v1. URL https://doi.org/10.20944/preprints202609.2187.v1.

Thomas Adler, Marco Grassia, Ziheng Liao, Giuseppe Mangioni, and Carlo Vittorio Cannistraci. Latent geometry-driven network automata for complex network dismantling. In *International Conference on Learning Representations*, volume 2026, pp. 104931–104980, 2026b.

Gregorio Alanis-Lobato, Pablo Mier, and Miguel A Andrade-Navarro. Efficient embedding of complex networks to hyperbolic space via their laplacian. *Scientific reports*, 6(1):30108, 2016.

Sámuel G Balogh, Bendegúz Sulyok, Tamás Vicsek, and Gergely Palla. Clove, a travelling salesman's approach to hyperbolic embeddings of complex networks with communities. *Communications Physics*, 8(1):397, 2025.

Albert-László Barabási and Réka Albert. Emergence of scaling in random networks. *science*, 286 (5439):509–512, 1999.

Andrew G Barto, Richard S Sutton, and Charles W Anderson. Neuronlike adaptive elements that can solve difficult learning control problems. *IEEE transactions on systems, man, and cybernetics*, (5):834–846, 1983.

Marian Boguna, Ivan Bonamassa, Manlio De Domenico, Shlomo Havlin, Dmitri Krioukov, and M Ángeles Serrano. Network geometry. *Nature Reviews Physics*, 3(2):114–135, 2021.

Carlo Vittorio Cannistraci and Alessandro Muscoloni. Geometrical congruence, greedy navigability and myopic transfer in complex networks and brain connectomes. *Nature Communications*, 13 (1):7308, 2022.

Carlo Vittorio Cannistraci, Timothy Ravasi, Franco Maria Montevecchi, Trey Ideker, and Massimo Alessio. Nonlinear dimension reduction and clustering by minimum curvilinearity unfold neuropathic pain and tissue embryological classes. *Bioinformatics*, 26(18):i531–i539, 2010.

Carlo Vittorio Cannistraci, Gregorio Alanis-Lobato, and Timothy Ravasi. Minimum curvilinearity to enhance topological prediction of protein interactions by network embedding. *Bioinformatics*, 29(13):i199–i209, 2013.

Diego Cerretti, Yingtao Zhang, and Carlo Vittorio Cannistraci. A dendritic-inspired network science generative model for topological initialization of connectivity in sparse artificial neural networks. In *NeurIPS 2025 Workshop on Symmetry and Geometry in Neural Representations*, 2025.

Spyridon Chavlis and Panayiota Poirazi. Dendrites endow artificial neural networks with accurate, robust and parameter-efficient learning. *Nature communications*, 16(1):943, 2025.

Patryk Chrabaszcz, Ilya Loshchilov, and Frank Hutter. A downsampled variant of imagenet as an alternative to the cifar datasets. *arXiv preprint arXiv:1707.08819*, 2017.

Gregory Cohen, Saeed Afshar, Jonathan Tapson, and Andre Van Schaik. Emnist: Extending mnist to handwritten letters. In *2017 international joint conference on neural networks (IJCNN)*, pp. 2921–2926. IEEE, 2017.

Nolan Dey, Shane Bergsma, and Joel Hestness. Sparse maximal update parameterization: A holistic approach to sparse training dynamics. *Advances in Neural Information Processing Systems*, 37: 33836–33862, 2024.

Demival Vasques Filho and Dion RJ O'Neale. Latent space generative model for bipartite networks. In *International Conference on Network Science*, pp. 3–16. Springer, 2020.

Guillermo García-Pérez, Antoine Allard, M Ángeles Serrano, and Marián Boguñá. Mercator: uncovering faithful hyperbolic embeddings of complex networks. *New Journal of Physics*, 21(12): 123033, 2019.

Wenqi Gu and Carlo Vittorio Cannistraci. Lightweight explainable physics-informed neural networks by learnable activation function and contextual modulation. 2026.

Srikanth K Iyer and Dahandapani Yogeshwaran. Percolation and connectivity in ab random geometric graphs. *Advances in Applied Probability*, 44(1):21–41, 2012.

Keller Jordan, Yuchen Jin, Vlado Boza, Jiacheng You, Franz Cesista, Laker Newhouse, and Jeremy Bernstein. Muon: An optimizer for hidden layers in neural networks, 2024. *URL https://kellerjordan. github. io/posts/muon*, 6(3):4, 2024.

Jeremy Kepner and Ryan Robinett. Radix-net: Structured sparse matrices for deep neural networks. In *2019 IEEE international parallel and distributed processing symposium workshops (IPDPSW)*, pp. 268–274. IEEE, 2019.

Diederik Kinga, Jimmy Ba Adam, et al. A method for stochastic optimization. In *International conference on learning representations (ICLR)*, volume 5, pp. 1. California;, 2015.

Dmitri Krioukov, Fragkiskos Papadopoulos, Maksim Kitsak, Amin Vahdat, and Marián Boguná. Hyperbolic geometry of complex networks. *Physical Review E—Statistical, Nonlinear, and Soft Matter Physics*, 82(3):036106, 2010.

Alex Krizhevsky, Geoffrey Hinton, et al. Learning multiple layers of features from tiny images.(2009), 2009.

Yann LeCun, Léon Bottou, Yoshua Bengio, and Patrick Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998.

Namhoon Lee, Thalaiyasingam Ajanthan, and Philip Torr. Snip: Single-shot network pruning based on connection sensitivity. In *International conference on learning representations*, 2018.

Man Li, Ye Zhu, Yuxin Shen, and Maia Angelova. Clustering-enhanced stock price prediction using deep learning. *World wide web*, 26(1):207–232, 2023.

Lu Lu, Xuhui Meng, Zhiping Mao, and George Em Karniadakis. DeepXDE: A Deep Learning Library for Solving Differential Equations. *SIAM Review*, 63(1):208–228, January 2021. ISSN 0036-1445, 1095-7200. doi: 10.1137/19M1274067. URL https://epubs.siam.org/doi/10.1137/19M1274067.

Alexander Lubotzky, Ralph Phillips, and Peter Sarnak. Ramanujan graphs. *Combinatorica*, 8(3): 261–277, 1988.

L. Mochurad and A. Dereviannyi. An ensemble approach integrating LSTM and ARIMA models for enhanced financial market predictions. *Royal Society Open Science*, 11(9):240699, Sep 2024. doi: 10.1098/rsos.240699.

Alessandro Muscoloni and Carlo Vittorio Cannistraci. A nonuniform popularity-similarity optimization (npso) model to efficiently generate realistic complex networks with communities. *New Journal of Physics*, 20(5):052002, 2018.

Alessandro Muscoloni, Josephine Maria Thomas, Sara Ciucci, Ginestra Bianconi, and Carlo Vittorio Cannistraci. Machine learning meets complex networks via coalescent embedding in the hyperbolic space. *Nature Communications*, 8(1), 2017. ISSN 2041-1723. doi: 10.1038/s41467-017-01825-5.

Jorge Nocedal et al. Updating quasi-newton matrices with limited storage. *Mathematics of computation*, 35(151):773–782, 1980.

Oleh Onyshchak. Stock market dataset, 2020. URL https://www.kaggle.com/dsv/1054465.

Fragkiskos Papadopoulos, Maksim Kitsak, M Ángeles Serrano, Marián Boguná, and Dmitri Krioukov. Popularity versus similarity in growing networks. *Nature*, 489(7417):537–540, 2012.

Fragkiskos Papadopoulos, Constantinos Psomas, and Dmitri Krioukov. Network mapping by replaying hyperbolic growth. *IEEE/ACM Transactions on Networking*, 23(1):198–211, 2014.

Fragkiskos Papadopoulos, Rodrigo Aldecoa, and Dmitri Krioukov. Network geometry inference using common neighbors. *Physical Review E*, 92(2):022807, 2015.

Tomaso Poggio. On efficiently computable functions, deep networks and sparse compositionality. *arXiv preprint arXiv:2510.11942*, 2025.

Tomaso Poggio and Maia Fraser. Compositional sparsity of learnable functions. *Bulletin of the American Mathematical Society*, 61(3), 2024.

Hado Van Hasselt, Arthur Guez, and David Silver. Deep reinforcement learning with double q-learning. In *Proceedings of the AAAI conference on artificial intelligence*, volume 30, 2016.

Chaojie Wang, Yuanyuan Chen, Shuqi Zhang, and Qiuhui Zhang. Stock market index prediction using deep transformer model. *Expert Systems with Applications*, 208:118128, 2022.

Sifan Wang, Hanwen Wang, and Paris Perdikaris. On the eigenvector bias of fourier feature networks: From regression to solving multi-scale pdes with physics-informed neural networks. *Computer Methods in Applied Mechanics and Engineering*, 384:113938, 2021.

Zuxi Wang, Qingguang Li, Fengdong Jin, Wei Xiong, and Yao Wu. Hyperbolic mapping of complex networks based on community information. *Physica A: Statistical Mechanics and its Applications*, 455:104–119, 2016a.

Zuxi Wang, Yao Wu, Qingguang Li, Fengdong Jin, and Wei Xiong. Link prediction based on hyperbolic mapping with community structure for complex networks. *Physica A: Statistical Mechanics and its Applications*, 450:609–623, 2016b.

Zuxi Wang, Lingjie Sun, Menglin Cai, and Pengcheng Xie. Fast hyperbolic mapping based on the hierarchical community structure in complex networks. *Journal of Statistical Mechanics: Theory and Experiment*, 2019(12):123401, 2019.

Jeremy Yu, Lu Lu, Xuhui Meng, and George Em Karniadakis. Gradient-enhanced physics-informed neural networks for forward and inverse PDE problems. *Computer Methods in Applied Mechanics and Engineering*, 393:114823, April 2022. ISSN 00457825. doi: 10.1016/j.cma.2022.114823. URL https://linkinghub.elsevier.com/retrieve/pii/S0045782522001438.

Yingtao Zhang, Jialin Zhao, Wenjing Wu, Alessandro Muscoloni, and Carlo Vittorio Cannistraci. Epitopological learning and cannistraci-hebb network shape intelligence brain-inspired theory for ultra-sparse advantage in deep learning. In *The Twelfth International Conference on Learning Representations*, 2024. URL https://openreview.net/forum?id=iayEcORsGd.

Yingtao Zhang, Diego Cerretti, Jialin Zhao, Wenjing Wu, Ziheng Liao, Umberto Michieli, and Carlo Vittorio Cannistraci. Brain network science modelling of sparse neural networks enables transformers and LLMs to perform as fully connected. In *The Thirty-ninth Annual Conference on Neural Information Processing Systems*, 2025. URL https://openreview.net/forum?id=OM0Qkq9xtY.

A Ablation Test for Geometric Supercharger

We ablate the main components of Geometric Supercharger on CIFAR-10 in Table 5. All tested methods produce angular coordinates that are subsequently used in the nBPSO generative network for sparse network initialization. Mercator (García-Pérez et al., 2019), a strong model-based hyperbolic embedding method, performs poorly in this setting. This is consistent with Mercator inferring coordinates under a predefined hyperbolic network model rather than adapting the geometry directly to the task data.

We next test ncMCE and Laplacian Eigenmaps (LE), two of the best dimensionality-reduction approaches used in Coalescent Embedding (Muscoloni et al., 2017). Among these, ncMCE provides the stronger initialization. We then evaluate the contribution of two additional macro-scale components around ncMCE: shortest-path Pearson (SPP) weighting before ncMCE and angular-distance (AD) reweighting after ncMCE, which results in a second pass. Neither SPP–ncMCE nor ncMCE–AD alone provides the same improvement as their combination. In contrast, SPP–ncMCE–AD produces a substantial increase in performance, indicating that both the initial topology-informed weighting and the subsequent geometry-informed reweighting contribute to the final embedding.

We also evaluate CLOVE with Louvain community detection, which corresponds closely to the mesoscale refinement component of GS. CLOVE performs well independently, but remains below the complete GS pipeline. Finally, we compare GS using either Louvain or Leiden community detection. Both variants outperform the other ablations, with Louvain giving slightly higher performance. Overall, the results indicate that the strongest performance arises from combining the macro-scale SPP–ncMCE–AD embedding with the subsequent community-based mesoscale refinement.

We also tested the angular-adjustment procedure proposed in Coalescent Embedding, which redistributes nodes uniformly around the disk while preserving their angular ordering; however, this did not improve performance.

Table 5: Test accuracy on CIFAR-10 for MLPs (99% sparsity) initialized using different embedding techniques. Mercator (García-Pérez et al., 2019) and CLOVE (Balogh et al., 2025) are hyperbolic embedding methods that use an unweighted network. ncMCE (Non-centered Minimum Curvilinear Embedding) and LE (Laplacian Eigenmaps) are the basic and best-performing dimensionality-reduction methods used in Coalescent Embedding (CE) (Muscoloni et al., 2017); they use the absolute Pearson distances from the sparsified pearson correlation matrix (Zhang et al., 2024). SPP (Shortest Path Pearson) includes the kernel computation before ncMCE. AD (Angular Distance) uses angular distances as weights for another embedding iteration. Supercharger includes SPP–ncMCE–AD; the original sparsified network is then weighted using angular distances from the previous embedding and fed into the weighted community-detection and arrangement steps, using either Louvain or Leiden.

	CIFAR-10
Mercator	60.32 ± 0.16
LE (CE)	60.54 ± 0.28
ncMCE (CE)	63.44 ± 0.25
SPP-ncMCE	62.91 ± 0.17
ncMCE-AD	63.78 ± 0.41
SPP-ncMCE-AD	64.32 ± 0.39
CLOVE - Louvain	64.41 ± 0.43
Supercharger - Leiden	65.13 ± 0.06
Supercharger - Louvain (GS)	$\mathbf{65.39 \pm 0.22}$

B Computational complexity of Geometric Supercharger

The geometric embedding and topology construction performed by Geometric Supercharger (GS) are executed once during initialization. Because this is a one-time setup, the computational cost does not slow down the training process. The computational complexity for each step is outlined below. Let V denote the number of nodes in the network being embedded, E the number of existing edges, and C the number of communities detected during meso-scale refinement.

Macro-scale embedding. The macro-scale embedding consists of two successive embedding passes. Each pass comprises an all-pairs shortest-path computation, construction of a Pearson-distance matrix from the resulting shortest-path profiles, and non-centered Minimum Curvilinear Embedding (ncMCE).

Shortest-path computation. For each node, shortest-path distances to all other nodes are computed. For a sparse graph with non-negative edge weights, performing Dijkstra's algorithm from every node requires approximately

$$O(VE \log V)$$

Pearson distance between shortest-path profiles. Each node is represented by its vector of shortest-path distances to all V nodes. Pearson distance is then computed between every pair of these V-dimensional profiles. There are $O(V^2)$ profile pairs, and evaluating each correlation requires $O(V)$ operations, resulting in

$$O(V^3)$$

ncMCE. The resulting distance matrix is embedded using non-centered Minimum Curvilinear Embedding (ncMCE). The omission of the centering operation allows ncMCE to operate with a time complexity of

$$O(V^2)$$

(Cannistraci et al., 2013).

After obtaining the first macro-scale embedding, angular distances are calculated between connected nodes and used to reweight the existing network. Since only existing edges are considered, this step requires

$$O(E)$$

operations.

The shortest-path, Pearson-distance, and ncMCE procedure is then repeated on this angular-distance-weighted network to obtain the final macro-scale embedding. Repeating the procedure introduces only a constant factor and therefore does not change its asymptotic complexity. Consequently, when the full pairwise Pearson-distance matrix is explicitly constructed, the macro-scale stage is dominated by the $O(V^3)$ shortest-path/profile-comparison operations:

$$T_{\text{macro}} = O\big(T_{\text{APSP}}(V, E) + V^3\big)$$

where T_{APSP} denotes the chosen all-pairs shortest-path algorithm. Under a standard dense implementation, this reduces to

$$T_{\text{macro}} = O(V^3)$$

Meso-scale refinement. Following the macro-scale embedding, GS performs a meso-scale refinement based on the community structure of the angular-distance-weighted network. Computing angular distances for the existing edges requires $O(E)$ operations, where E is the number of edges.

Weighted Louvain community detection is approximately linear in the number of edges per pass, giving a practical complexity of

$$O(kE)$$

where k denotes the number of Louvain passes.

The detected communities are then arranged according to their inter-community relationships using the travelling-salesman-based ordering procedure adopted from CLOVE (Balogh et al., 2025). Let C denote the number of detected communities. Following the simplified complexity analysis of CLOVE, this community-ordering step is treated as having at least quadratic complexity in C,

$$O(C^2)$$

Therefore, the computational complexity of one meso-scale refinement step is

$$T_{\text{meso}}^{(1)} = O(E) + O(kE) + O(C^2)$$

which simplifies to

$$T_{\text{meso}}^{(1)} = O(kE + C^2)$$

Since the meso-scale refinement is repeated p times, its total complexity is

$$T_{\text{meso}} = O\left(p(kE + C^2)\right)$$

Overall complexity. The computational cost of GS is dominated by the macro-scale embedding. When full pairwise shortest-path profiles and their Pearson distances are explicitly evaluated, the macro-scale procedure has complexity

$$O(V^3)$$

Including the repeated meso-scale refinement, the overall one-time computational complexity is therefore

$$T_{\text{GS}} = O(V^3) + O\left(p(kE + C^2)\right)$$

For sparse networks, where $E \ll V^2$, and when the number of Louvain passes k, detected communities C, and refinement iterations p remain moderate, the $O(V^3)$ macro-scale term dominates the meso-scale operations. Thus, the overall asymptotic complexity of GS remains

$$\boxed{O(V^3)}$$

C LAYER REPLICATION

Table 6: Test accuracy on CIFAR-10 for MLPs (99% sparsity) initialized with Geometric Supercharger (GS) ($\gamma = 3$) and statically trained with and without layer replication. Each value is the mean over three seeds, with the standard error reported.

	GS - CIFAR-10		
	$\gamma = 2.0$	$\gamma = 2.5$	$\gamma = 3.0$
No layer replication	57.62 ± 0.08	63.13 ± 0.28	65.40 ± 0.22
Layer replication	57.54 ± 0.39	63.47 ± 0.32	65.39 ± 0.22

In the CSTI framework, a single correlation matrix is used to construct the connectivity masks for all sparsified layers, resulting in identical topological structures throughout the network. We investigated whether replicating a single topology across all layers impacts performance compared to allowing each layer to have its own unique structure.

In our Geometric Supercharger model, the final network topology is determined by both the angular coordinates (θ), derived from the correlation matrix, and the radial coordinates (r), assigned during

the nBPSO network growth process. For the **replicated layers** approach, we execute the nBPSO growth process only once; the resulting sparse connectivity mask is then duplicated across all hidden layers. For the **non-replicated layers** approach, every layer is still initialized with the exact same data-informed angular coordinates (θ). However, we execute a separate nBPSO growth process independently for each layer. Due to the stochastic nature of the nBPSO generative mechanism, this results in unique radial coordinates and distinct wiring patterns for each layer, despite sharing the same underlying angular geometry.

As shown in Table 6, both approaches result in similar accuracies across varying values of γ, with neither method demonstrating a definitive advantage on CIFAR-10. Consequently, for simplicity and to maintain consistency with the CSTI baseline, we adopted the replicated layer approach for all experiments.

D HIERARCHY AND SCALE-FREENESS

In the Geometric Supercharger model, the degree of hierarchy is controlled by the power-law exponent γ used in the nBPSO bipartite generative network model. The degree distribution of a scale-free network follows a power law, $P(k) \sim k^{-\gamma}$. In the nBPSO model, $\beta = 1/(\gamma - 1)$, where β is the popularity-fading parameter that dictates how quickly nodes lose their attractiveness to new connections over time. A network with a power-law degree distribution is typically termed scale-free when $2 \leq \gamma \leq 3$ (Barabási & Albert, 1999); it is characterized by a few highly connected "hub" nodes and a vast majority of low-degree nodes. This structure forms a hierarchy. As γ increases, the distribution becomes less extreme, resulting in a softer hierarchy.

To evaluate the importance of this hierarchical structure, we compared different γ values and assessed the impact of rewiring, with and without layer replication, in Table 7. We used the uniformly random rewiring method from the original nBPSO paper to rewire the outgoing connections of specific sets (left, right, or both). This effectively destroys the scale-free property, replacing the hierarchical power-law distribution with a flat, binomial degree distribution.

The results yield two key observations. First, a weaker hierarchy ($\gamma = 3$) consistently outperforms stronger hierarchies (lower γ values). Second, maintaining the intact topology (no rewiring) strictly outperforms rewiring the left or right node set, or both sets. While the superior performance of $\gamma = 3$ indicates that an overly aggressive hierarchy is suboptimal, the severe performance degradation caused by rewiring confirms that preserving a power-law hierarchy remains critical.

Table 7: Test accuracy on CIFAR-10 for MLPs (99% sparsity) initialized with Geometric Supercharger (GS) and statically trained with varying γ values and rewiring modes, with and without layer replication. Rows represent the topological structures of sets A and B in the bipartite networks. ER denotes a set that has been randomly rewired (binomial degree distribution), and γ denotes a set that retains its hierarchical structure (power-law degree distribution). Each value is the mean over three seeds, with the standard error reported. Bold denotes the best-performing combination of γ and rewiring mode.

GS - CIFAR-10 - No Layer Replication

	$\gamma = 2.0$	$\gamma = 2.5$	$\gamma = 3.0$
γ, γ	57.62 ± 0.08	63.13 ± 0.28	$\mathbf{65.40 \pm 0.22}$
γ, ER	57.95 ± 0.13	58.25 ± 0.43	57.75 ± 0.26
ER, γ	55.51 ± 0.09	56.44 ± 0.34	57.24 ± 0.17
ER, ER		57.41 ± 0.17	

GS - CIFAR-10 - Layer Replication

	$\gamma = 2.0$	$\gamma = 2.5$	$\gamma = 3.0$
γ, γ	57.54 ± 0.39	63.47 ± 0.32	$\mathbf{65.39 \pm 0.22}$
γ, ER	57.83 ± 0.32	58.26 ± 0.45	58.16 ± 0.21
ER, γ	55.21 ± 0.38	56.60 ± 0.08	57.18 ± 0.15
ER, ER		57.41 ± 0.24	

E STATIC SPARSE TRAINING

In static sparse training, the sparse network structure is set before training and remains fixed. During training, only the weights of these initial connections are updated via gradient descent; no connections are added or removed.

F BASELINE NEURAL NETWORK INITIALIZATION METHODS

F.1 DATA-AGNOSTIC METHODS

Fully Connected (FC) Network. The fully connected network is a standard dense network in which all possible connections between adjacent layers are present.

Random (ER-like). The random model is a standard sparse baseline in which connections between layers are initialized uniformly at random, similar to an Erdős-Rényi (ER) graph.

Bipartite Small-World (BSW). The BSW model (Zhang et al., 2024) adapts the Watts-Strogatz model for bipartite networks. It initializes connections based on a regular lattice to ensure high local clustering, then randomly rewires a fraction of them (controlled by a parameter β) to create the small-world property (short characteristic path length). $\beta = 0$ amounts to a regular lattice (no rewiring) and $\beta = 1$ to a random network (all links rewired).

Bipartite Receptive Field (BRF). BRF (Zhang et al., 2025) generates sparse topologies mimicking the spatially restricted receptive fields of biological brains. A parameter $r \in [0, 1]$ controls the spatial randomness: $r \to 0$ enforces strictly localized, diagonal-heavy connections, while $r \to 1$ yields uniformly random connectivity. BRF also enables custom degree distribution profiles.

Ramanujan. Ramanujan graphs (Lubotzky et al., 1988) are optimal expander graphs derived from number theory. Characterized by a maximal spectral gap, they guarantee highly efficient information flow, short network diameters, and optimal connectivity at extreme sparsity levels. We use the adapted bipartite Ramanujan graph construction method from (Cerretti et al., 2025) to initialize our networks.

RadiX-Nets. RadiX-Nets (Kepner & Robinett, 2019) deterministically construct sparse topologies *de novo* using mixed-radix numeral systems and Kronecker products. This mathematical construction guarantees path-connectedness and perfect symmetry (equal paths between input and output nodes).

Dendritic Artificial Neural Network (dANN-R). Inspired by the structural connectivity and restricted sampling properties of biological dendrites, dANN-R (Chavlis & Poirazi, 2025) splits the standard point-neuron into a dendritic layer (which sparsely samples inputs) and a somatic layer (which aggregates dendritic signals). The "R" variant randomly selects input features for each dendrite. This version was shown to be the best-performing dANN variant by (Cerretti et al., 2025).

Dendritic Network Model (DNM). The DNM (Cerretti et al., 2025) is a generative framework that explicitly models the clustered, compartmentalized connectivity of biological dendrites. Rather than using unstructured randomness, it iteratively generates sparse bipartite layers by parametrically controlling node degree, local receptive fields, and synaptic distributions.

F.2 DATA-INFORMED METHODS

Correlated Sparse Topological Initialization (CSTI). CSTI (Zhang et al., 2024) initializes the network topology based on the underlying data distribution by executing the following four steps:

1. **Vectorization:** A subset of n data points is randomly sampled from the training set. Features with zero variance are discarded, and the remaining M valid features are used to construct an $n \times M$ data matrix.

2. **Feature Selection:** The Pearson correlation coefficient is computed for all pairs of valid features to generate an $M \times M$ correlation matrix.
3. **Connectivity Selection:** The correlation matrix is converted into a binary sparse adjacency matrix. Entries corresponding to the top-$k\%$ highest correlation values are set to 1, while the rest are set to 0, where k is the target density. The threshold k is determined by the target network sparsity, and a scaling factor is applied to match the dimensions of the hidden layers.
4. **Assembling Network Blocks:** This final sparse adjacency matrix acts as a structural mask, which is then applied to shape the initial topology of each layer (sandwich layer) throughout the network.

SNIP. Single-Shot Network Pruning (SNIP) (Lee et al., 2018) is a pruning method that identifies crucial connections before training. It evaluates connection sensitivity (the product of a weight and its gradient) on a small batch of data, pruning the least sensitive links to reach a target sparsity level from scratch. This pruning method can essentially be used as a sparse initialization method.

G IMAGE CLASSIFICATION

Data. Below are the image-classification datasets used in this study, ordered by increasing complexity from easiest (MNIST) to hardest (ImageNet32-200).

1. **MNIST** (LeCun et al., 1998) contains 70,000 grayscale images of handwritten digits (0–9). Each image is 28×28 pixels, and the dataset is typically split into 60,000 training images and 10,000 testing images.
2. **EMNIST** (Extended MNIST) (Cohen et al., 2017) expands upon the MNIST dataset to include handwritten uppercase and lowercase letters. It maintains the same 28×28 grayscale format.
3. **CIFAR-10** (Krizhevsky et al., 2009) consists of 60,000 RGB color images of 32×32 pixels, distributed equally across 10 mutually exclusive classes (e.g., airplanes, cars, birds, cats). It is divided into 50,000 training images and 10,000 testing images.
4. **CIFAR-100** (Krizhevsky et al., 2009) has the same 32×32 color image format and total size (60,000 images) as CIFAR-10. However, it is partitioned into 100 fine-grained classes, with each class containing 600 images; these classes are further grouped into 20 broader superclasses.
5. **ImageNet32-200-100** (Chrabaszcz et al., 2017) is a reduced subset of ImageNet32, a 32×32 pixel downsampled version of the original ImageNet dataset. We select 200 classes and sample 100 images per class, resulting in a total of 20,000 RGB images.

Architecture & Training. All experiments utilize a 4-layer MLP initialized with Sparse Weight Initialization (SWI) to preserve variance across sparse layers. For CIFAR-10, CIFAR-100, and ImageNet32-200, the input size is 3072 with three hidden layers of 3072 neurons. For MNIST and EMNIST, the input size is 784 with three hidden layers of 1568 neurons (twice the size of the input layer). Models are trained for 100 epochs using stochastic gradient descent (SGD) with a momentum of 0.9 and a weight decay of 5×10^{-4}. By default, the batch size is 32, and the learning rate decays linearly from 0.025 to 2.5×10^{-4}. The only parameter search was conducted for all methods in Table 1, over batch sizes $\in \{32, 64, 128\}$ and initial learning rates $\in \{0.025, 0.01, 0.001\}$.

Geometric Supercharger initialization. Geometric Supercharger uses only the training inputs to construct the initial sparse topology. Each image is flattened into a feature vector, such that CIFAR-10, CIFAR-100, and ImageNet32-200-100 contain $32 \times 32 \times 3 = 3072$ input features. Across a subset of training images, we compute the Pearson correlation between every pair of features, producing a 3072×3072 correlation matrix. We retain the strongest absolute correlations according to the target sparsity to obtain a sparse weighted network, which is then embedded by Geometric Supercharger to infer the angular coordinates.

Hyperparameters. The main parameter search was conducted for all methods in Table 1, over batch sizes $\in \{32, 64, 128\}$ and initial learning rates $\in \{0.025, 0.01, 0.001\}$. Otherwise, we chose or searched over the following hyperparameters for each method:

- **GS:** Defaults to $T = 0$, no rewiring, and $\gamma = 3$, with replicated layers.
- **nBPSO:** Defaults to $T = 0$, no rewiring, 0 communities ($C = 0$), and non-replicated layers. We additionally searched $C \in \{0, 4\}$ and $\gamma \in \{2, 2.5, 3\}$ for Table 1.
- **BSW:** The rewiring parameter is set to $\beta = 0.5$ to balance local clustering with small-world properties.
- **BRF:** We set $r = 0.25$ and used a fixed degree distribution.
- **dANN-R:** Hyperparameters were directly adopted following the optimal configuration established in the original DNM paper (Cerretti et al., 2025). Specifically, settings were selected to match the parameter count of the other sparse baseline networks:
  - **CIFAR-10:** 416 somas, 8 synapses
  - **CIFAR-100:** 272 somas, 8 synapses
  - **ImageNet32-200:** 272 somas, 8 synapses
- **DNM:** Hyperparameters were taken directly from the suggested configuration for static sparse training in Cerretti et al. (2025): 3 dendrites per neuron, a fixed dendritic distribution, a receptive field width of 1.0, and spatial Inverse Gaussian distributions for the receptive field width, degree, and synaptic placements.

H Physics-Informed Neural Networks (PINN)

Data. We evaluate Geometric Supercharger on two physics-informed learning problems: the two-dimensional Helmholtz equation (Lu et al., 2021) and the one-dimensional inverse Brinkman problem (Yu et al., 2022). We follow the experimental setup from Gu & Cannistraci (2026).

In all experiments, derivatives appearing in the governing equations are computed by automatic differentiation. The training objective is the unweighted sum of the mean-squared residuals associated with the governing equation and the corresponding initial, boundary, or observation constraints.

Fourier-Feature Architecture. All tasks use an MLP preceded by a Fourier-feature coordinate mapping (Wang et al., 2021). Given an input coordinate $\mathbf{x} \in \mathbb{R}^d$, the coordinates are first normalized to the interval $[-1, 1]$. Two fixed random Fourier-feature mappings are then applied at scales $\sigma \in \{1, 10\}$, using sine and cosine projections to represent the input at two spatial-frequency scales. Both Fourier branches are processed by the same three-hidden-layer MLP with $\tanh$ activations; the backbone weights are shared between the two scales. The final representations of the two branches are concatenated and passed through a dense linear output layer.

Table 8: Architecture and training settings for the physics-informed neural network experiments. Here, m denotes the number of Fourier projections per branch and w the shared hidden-layer width.

Setting	Helmholtz	Inverse Brinkman
Input dimension d	2	1
Hidden width w	62	22
Fourier features m	36	25
Fourier scales	$\{1, 10\}$	$\{1, 10\}$
Learning rate	2×10^{-3}	5×10^{-4}
Muon iterations	5,000	20,000
L-BFGS iterations	15,000	15,000

No learning-rate scheduler is used. The first optimization stage uses Muon (Jordan et al., 2024) with momentum 0.95 and weight decay 0.01. Biases, the dense output layer, and other non-matrix parameters are optimized using Adam. For the inverse Brinkman problem, the physical parameters ν_e and K are optimized using Adam without weight decay.

The first-order optimization stage is followed by 15,000 iterations of L-BFGS (Nocedal et al., 1980) over all trainable network and physical parameters. FC and GS models use identical task-specific learning rates, collocation points, Fourier mappings, and optimization budgets for each seed.

Geometric Supercharger initialization. Geometric Supercharger replaces the three trainable fully connected backbone matrices with static sparse matrices, while the Fourier projections, biases, and final output layer remain dense. The sparse topology is constructed before training using only the coordinates available in the training set. Each coordinate is passed through both fixed Fourier encoders. Because the two Fourier branches share the same backbone, the encoded outputs from both branches are stacked as additional observations of the same Fourier features. Pairwise Pearson correlations are then computed between these features.

The absolute Pearson correlations are used to construct a sparse weighted correlation network at the desired sparsity. Geometric Supercharger embeds this network to obtain data-informed angular coordinates, which are then supplied to the hyperbolic network generator. Weights are initialized using Sparse Weight Initialization (SWI).

Evaluation. For the forward Helmholtz problem, performance is evaluated using the global relative L_2 solution error. For the inverse Brinkman problem, the primary metric is the mean relative recovery error of the two unknown physical parameters ν_e and K.

Table 9: Summary of the governing equations, computational domains, initial and boundary conditions, and primary evaluation metrics for the PINN experiments. Here, u^* denotes a reference solution. Helmholtz is an oscillatory-system task, and inverse Brinkman is an inverse-problem task.

Task	Equation	Domain	Conditions	Primary metric
Helmholtz (Lu et al., 2021)	$-\Delta u - (2\pi)^2 u = f$, $u^* = \sin(2\pi x)\sin(2\pi y)$, $f = 4\pi^2 u^*$	$(x, y) \in [0, 1]^2$	$u = 0$ on $\partial\Omega$	Global relative L_2
Inverse Brinkman (Yu et al., 2022)	$-\frac{\nu_e}{0.4} u_{xx} + \frac{10^{-3}}{K} u = 1$; recover ν_e, K, $\nu_e^* = K^* = 10^{-3}$	$x \in [0, 1]$	$u(0) = u(1) = 0$	Mean relative error of ν_e and K

I STOCK-PRICE PREDICTION

Data. We used daily historical stock-market data from the Kaggle Stock Market Dataset (Onyshchak, 2020), originally collected from Yahoo Finance. The dataset provides daily closing-price observations. The selected securities contain between 5,353 and 14,664 trading-day observations, with histories beginning between January 1962 and December 1998 and ending in early April 2020. Each series was divided chronologically into approximately 80% training data and 20% test data, ensuring that the model was trained on earlier observations and evaluated on a later, unseen period. This produced 4,284–11,731 raw training observations and 1,069–2,933 raw test observations per security.

For each split, closing prices were standardized to zero mean and unit variance. Each series was then converted into overlapping windows, where the preceding 20 trading-day closing prices were used to predict the standardized closing price on the following trading day. Thus, each input contained 20 lagged closing prices and the output was a single scalar. Training windows were shuffled during optimization, whereas test windows retained their chronological order.

The 17 assets are AA, ARNC, BA, CNP, DD, DIS, DXC, F, GE, HON, HPQ, IP, JNJ, MCD, MRO, XLP, and YRCW.

Architecture. The network consists of an input layer of size 20, three hidden layers of 40 neurons each, and a scalar regression output:

$$20 \to 40 \to 40 \to 40 \to 1.$$

Dropout was set to zero in all experiments. In the FC baseline, all connections were trainable. In GS, the three hidden weight matrices were sparsified, while their biases and the final $40 \to 1$ output layer remained dense.

Training. Models were trained for 100 epochs using stochastic gradient descent with momentum 0.9, weight decay 5×10^{-4}, a batch size of 32, and an initial learning rate of 0.025. The learning rate was reduced linearly to 1% of its initial value over the first 90 epochs and then held constant.

Geometric Supercharger initialization. Supercharger used only the training inputs to construct the initial sparse topology. For each asset, we computed the Pearson correlation matrix between the 20 lag positions across all training windows. Because both hidden-to-hidden matrices had shape 40×40, a single topology was replicated across them. Because the input mask had rectangular dimensions of 40×20, it was instead generated uniformly at random. We used Sparse Weight Initialization (SWI) to initialize the weights.

Evaluation. Performance was measured using mean squared error on the standardized test targets:

$$\mathrm{MSE} = \frac{1}{N_{\mathrm{test}}} \sum_{i=1}^{N_{\mathrm{test}}} (\hat{y}_i - y_i)^2 .$$

Lower values therefore indicate more accurate next-day closing-price predictions.

J REINFORCEMENT LEARNING

CartPole task. We evaluate fully connected and sparse Geometric Supercharger-initialized (GS) MLPs on CartPole-v1, a standard reinforcement-learning control task (Barto et al., 1983). The agent observes the position and velocity of a cart and the angle and angular velocity of an attached pole. At each step, it selects one of two actions, corresponding to pushing the cart left or right. The agent receives a reward of $+1$ for every step during which the episode continues. An episode terminates if the cart or pole exceeds the standard CartPole limits and is truncated after 500 steps. Thus, the undiscounted episode return measures how long the agent balances the pole and has a maximum value of 500; higher values are better.

Architecture. Both agents use an MLP to estimate the action-value function $Q_\theta(\mathbf{s}, a)$. The network consists of a dense 4-to-H input layer, one H-to-H hidden layer, and a dense H-to-2 output layer. The two outputs represent the estimated values of the left and right actions.

Observations are divided element-wise by the fixed scale vector $(2.4, 2.5, 0.2095, 2.5)$ before being passed to the network. This normalization is identical for both models.

The dense baseline uses all H^2 possible connections in the H-to-H hidden layer. Its total number of trainable parameters is

$$P_{\mathrm{dense}} = H^2 + 8H + 2. \tag{1}$$

Geometric Supercharger initialization. The GS network replaces this hidden layer with a fixed sparse layer generated using Geometric Supercharger. The input and output layers remain dense. Only active sparse edges are stored and optimized. If the GS hidden layer contains E active edges, the total parameter count is

$$P_{\mathrm{GS}} = E + 8H + 2. \tag{2}$$

All parameter counts include weights and biases.

To construct the GS topology, we first collect 2,048 states using a uniformly random policy. These states are passed through the randomly initialized dense input layer and its ReLU activation. The resulting H-dimensional hidden activation vectors are used as the node features.

Training. Both models are trained from scratch using Double Deep Q-learning (Van Hasselt et al., 2016). Training uses an experience replay buffer containing at most 50,000 transitions. The first 1,000 environment steps are used to populate the replay buffer with random actions. The networks are optimized using Adam (Kinga et al., 2015) with a learning rate of 10^{-3}, no weight decay, and the smooth L_1 loss. Exploration follows an ϵ-greedy policy. The exploration probability decreases linearly from 1.0 to 0.05 over the first 5,000 environment steps and remains at 0.05 thereafter. Each model is trained for 500 episodes, with a maximum of 500 steps per episode.

Evaluation. After training, each model is evaluated for 20 episodes using a greedy policy, with no exploratory actions. For training seed k, the evaluation score is the mean undiscounted return across these episodes:

$$\overline{R}_k = \frac{1}{20} \sum_{j=1}^{20} R_{k,j}. \tag{3}$$

K Limitations & Future work

One could extend Geometric Supercharger to explicitly generate community structure, as in the nBPSO model, while allowing the organization of these communities to be informed by the data. Another direction would be to replace Pearson correlation with alternative data-informed connection scores. For example, SNIP (Lee et al., 2018) could be used to construct the weighted sparse network subsequently embedded by Geometric Supercharger. Unlike Pearson correlation, however, SNIP requires propagating a calibration batch through the network and computing gradients of the loss with respect to the weights to obtain connection-sensitivity scores.

L Ethics statement

We do not foresee any ethical issues with our work. Our work is focused on optimizing neural networks using network science principles and complex systems theory, which is a technical contribution to the field of machine learning. We do not anticipate any negative societal impacts or ethical concerns arising from our research.

M Reproducibility statement

All experiments were conducted on NVIDIA A100 80GB GPUs. All experimental details are provided in the paper, and the Geometric Supercharger code is available in the supplementary material.

N AI use statement

In this work, we used generative AI tools to assist with implementing methods in code, designing or providing feedback on research, supporting qualitative and thematic data analysis, and interpreting results. We did not use generative AI tools to develop theoretical models or conceptual frameworks, formulate mathematical claims, propose or refine hypotheses, or design methodology or experiments. Generating synthetic datasets, providing critical ingredients for proving mathematical claims, assisting with writing proofs or translation, and cleaning or reformatting datasets were not applicable to this work. Additionally, we used generative AI tools to suggest experimental parameters, create or edit software code, draft parts of the paper, and improve its readability. We have thoroughly reviewed and checked all AI-assisted work. We take responsibility for the final content of this work, including any text, claims, or artifacts produced with the aid of generative AI. The principles, core logic, and innovations are entirely the authors'.