A Dendritic-Inspired Network Science Generative Model for Topological Initialization of Connectivity in Sparse Artificial Neural Networks

Diego Cerretti[1,2] DIEGOCERRETTI02@GMAIL.COM
Yingtao Zhang[1,2] ZHANGYINGTAO1024@GMAIL.COM
Carlo Vittorio Cannistraci[1,2,3*] KALOKAGATHOS.AGON@GMAIL.COM

[1] *Center for Complex Network Intelligence (CCNI)*[†]
[2] *Dept. of Computer Science,* [3] *Dept. of Biomedical Engineering, Tsinghua University*

Editors: List of editors' names

Abstract

To address the inefficiency of traditional artificial neural networks, we develop the Dendritic Network Model (DNM), a technique for generating sparse, biologically-inspired architectures.

Keywords: Artificial Intelligence, Dendritic Network Model, Dynamic Sparse Training, Brain Network Science, Network Topology

1. Introduction

Artificial neural networks (ANNs) have demonstrated remarkable potential in various fields; however, their size, often comprising billions of parameters, poses challenges for both economic viability and environmental sustainability. Biological neural networks, on the contrary, can efficiently process information using ultra-sparse structures (Drachman, 2005; Walsh, 2013). A central component of the brain's architecture is the dendritic tree, the primary receptive surface of the neuron (Cuntz et al., 2010). Current approaches that aim to integrate the dendritic structure in modern ANNs are often limited to fixed, mathematically defined structures. Therefore, we identify a critical gap in the field: the lack of a flexible, principled framework for generating and testing dendritic topologies.

To address this, we introduce the Dendritic Network Model (DNM), a novel generative technique for creating sparse, biologically-inspired network architectures. The DNM constructs a flexible dendritic-inspired network topology grounded in the principles of network science rather than direct morphological imitation. The model's parametric approach enables the systematic exploration of the relationship between network structure and functionality.

* Corresponding author.
† Research Center in Tsinghua Laboratory of Brain and Intelligence (THBI), Department of Psychological and Cognitive Sciences.

© 2025 D. Cerretti, Y. Zhang & C.V. Cannistraci.

2. The Dendritic Network Model

2.1. Biological Inspiration and Principles

The architecture of the Dendritic Network Model (DNM) is inspired by the structure of biological neurons. In the nervous system, neurons process information through complex, branching extensions called dendrites, which act as the primary receivers of synaptic signals. Inspired by this phenomenon, the DNM shapes each sandwich layer (a bipartite subnetwork that links the input nodes of a layer to their output nodes) in a way that output neurons are connected to distinct groups of input neurons with multiple dendritic branches (Figure 1). In the DNM, an output neuron's connections are organized into distinct clusters called dendrites. Each branch connects the output neuron to a consecutive batch of input neurons. These connected blocks are separated by "inactive spaces", which are groups of input neurons that remain unconnected to that specific output neuron. All of an output neuron's branches must be formed within a localized receptive window, a predefined segment of the input layer. This method results in a structured and clustered connectivity pattern, moving away from random, unstructured sparsity.

2.2. Parametric Specification

By parametrizing the core biological features of DNM, like the number of dendrites for each output neuron, the size of the receptive windows, the distribution of synapses across dendrites, and the degree distribution across output neurons, the DNM provides a flexible framework for generating network topologies that are sparse, structured, and biologically plausible. Figure 4 shows how the connectivity of an MLP is shaped by the DNM.

Sparsity (s) The sparsity parameter (s) defines the percentage of potential connections between the input and output layers that are absent, controlling the trade-off between computational cost and representational power.

Dendritic Distribution The dendritic distribution governs the number of branches that connect each output neuron to the input layer, which can be seen as the number of distinct input regions a neuron integrates information from. The central parameter for this is M, which defines the mean number of dendrites per output neuron. This distribution can be implemented in one of three ways. The simplest is a fixed distribution, where every output neuron has exactly M dendrites. Alternatively, a non-spatial distribution introduces stochasticity by sampling the number of dendrites for each output neuron from a probability distribution (e.g., Gaussian or uniform) with a mean of M. Finally, a spatial distribution makes the number of dendrites for each neuron dependent on its position within the layer. Using a Gaussian or inverted Gaussian profile, this configuration implies that some neurons integrate signals from many distinct input regions (a high dendrite count), while others connect to fewer, more focused regions (a low dendrite count).

Receptive Field Width Distribution The receptive field width distribution determines the size of the receptive field for each output neuron, mirroring the concept of receptive fields in biology. This process is governed by a mean parameter, α, which specifies the average percentage of consecutive input neurons on the input layer from which an output neuron can sample connections. This distribution can be configured in several ways: a

fixed distribution assigns an identical window size α to all output neurons; a non-spatial distribution introduces variability by drawing each neuron's window size from a probability distribution (e.g., Gaussian or uniform) centered on α; and a spatial distribution links the window size to the neuron's position in its layer, allowing for configurations where receptive windows are, for instance, wider at the center and narrower at the edges.

Degree Distribution The degree distribution samples the number of incoming connections for each output neuron. This can be configured using a fixed distribution, where every output neuron is allocated the same degree. To introduce heterogeneity, a non-spatial distribution can be used to sample the degree for each neuron from a probability distribution. Finally, a spatial distribution allows the degree to vary based on the neuron's position, for instance, by creating highly connected, hub-like neurons at the center of the layer. The mean degree is set by the layer size and target sparsity.

Synaptic Distribution Once an output neuron's total degree is determined, the synaptic distribution allocates these connections among its various dendritic branches. The allocation can be fixed, where each dendrite receives an equal number of synapses. Alternatively, a non-spatial distribution can introduce random variability in synapse counts per dendrite. A spatial distribution can also be applied, making the number of synapses dependent on a dendrite's topological location, for example by assigning more connections to central branches versus outer ones. This distribution has a mean of $\frac{N_{in} \cdot (1-s)}{M}$, where N_{in} is the size of the input layer.

Layer Border Wiring Pattern The DNM includes a setting to control how connections are handled at the boundaries of the input layer. The default behavior is a *wrap-around* topology, where the input layer is treated as a ring. This means a receptive window for a neuron near one edge can wrap around to connect to neurons on the opposite edge, ensuring all neurons have a similarly structured receptive field. Alternatively, a *bounded* pattern can be enforced. In this mode, receptive windows are strictly confined within the layer's physical boundaries. If a receptive field extends beyond the first or last input neuron, it is clamped to the edge. This enforces a more stringent locality, which we analyze further in Appendix I.

2.3. Network Topology and Geometric Characterization

The parametric nature of the DNM allows for the generation of a vast landscape of network topologies.[1] In this section, we explore this landscape by systematically varying the model's key hyperparameters and analyzing the resulting structural changes. We employ both visual analysis and quantitative measures grounded in network science to provide a comprehensive geometric characterization of the generated networks.

Figure 2 illustrates this topological diversity by comparing a baseline random network with several DNM configurations in a 3-layered MLP of dimensions $98 \times 196 \times 196$, with 90% sparsity. Each panel displays the network's coalescent embedding (Cacciola et al., 2017) in hyperbolic space, its adjacency matrix, and a direct bipartite graph representation, alongside key network science metrics: characteristic path length (L), modularity (Q),

1. To facilitate an intuitive exploration of this landscape, we have developed an interactive web application where readers can adjust the model's parameters and visualize the resulting network structures. The application is available at: https://dendritic-network-model.streamlit.app/

structural consistency (σ_c), and the power-law exponent of the degree distribution (γ). The coalescent embedding method and the metrics adopted are detailed in Appendix A. The baseline random network (Figure 2a) exhibits a lack of clear structure, which is quantified by low modularity and structural consistency ($Q = 0.14$, $\sigma_c = 0.04$). The high value of γ and the uniform scattering of nodes in the hyperbolic embedding confirm the absence of any scale-free organization. Figure 2b represents a DNM network with $M = 3$, $\alpha = 1$, and all distributions fixed. This model generates highly structured and modular networks, as evidenced by the high modularity ($Q = 0.64$) and structural consistency ($\sigma_c = 0.76$). Similar measures are found when setting a spatial Gaussian synaptic distribution (Figure 2d, $Q = 0.54$, $\sigma_c = 0.74$), because this configuration does not alter the structure of the network much, as highlighted by the adjacency matrix depicted. A key finding is that by setting a spatial Gaussian degree distribution (Figure 2c), the DNM generates a hierarchical structure. Indeed, the power law exponent yielded ($\gamma = 2.30$) falls within the typical range for scale-free networks ($\gamma \leq 3$). This property is clearly evidenced by the emergence of a tree-like structure in the hyperbolic embedding. This analysis shows that DNM is a highly flexible framework that can produce a wide spectrum of network architectures. This ability to controllably generate diverse network geometries is fundamental for analyzing the relationship between network structure and computational function in ANNs. In Appendix F we detail the experiments performed. The results show that DNM can effectively boost the performance of both static and dynamic sparse training methods on various tasks. To understand which network structures are inherently best suited for specific tasks, we analyze the topologies of the top-performing models from our static sparse training experiments. Static sparse training is ideal for this analysis because its fixed topology allows us to link network structure to task performance directly. Figure 3 shows the adjacency matrices of these models, their direct bipartite graph representations, and their key metrics in network science. For image classification on MNIST and Fashion MNIST, the optimal network's topology is identical. This network is scale-free ($\gamma \leq 3$) and exhibits a short characteristic path length. Also on EMNIST, the best-performing network exhibits a power-law degree distribution. Finally, we obtain contrasting results when assessing the network adopted for CIFAR-10 classification. Its high γ parameter indicates that this network lacks hub nodes, possibly hinting that for more complex datasets like CIFAR-10, a more distributed and less hierarchical connectivity pattern is advantageous. Appendix C gives a more detailed analysis of the best parameter combinations for the tests performed. Overall, this analysis reveals a compelling relationship between task complexity and optimal network topology. While simpler, more structured datasets like MNIST and EMNIST benefit from scale-free, hierarchical architectures that can efficiently integrate global features through hub neurons, the more complex CIFAR-10 dataset favors a flatter, more distributed architecture.

3. Conclusion

DNM is a highly flexible tool capable of producing a wide spectrum of network architectures, from modular to hierarchical and scale-free, by systematically adjusting its core parameters. At extreme sparsity levels, DNM consistently outperforms alternative topological initialization methods in both static and dynamic sparse training regimes.

Acknowledgments

This work was supported by the Zhou Yahui Chair Professorship award of Tsinghua University (to C.V.C.), the National High-Level Talent Program of the Ministry of Science and Technology of China (grant number 20241710001, to C.V.C.).

References

Mukund Balasubramanian and Eric L Schwartz. The isomap algorithm and topological stability. *Science*, 295(5552):7–7, 2002.

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

Guillaume Bellec, David Kappel, Wolfgang Maass, and Robert Legenstein. Deep rewiring: Training very sparse deep networks. *arXiv preprint arXiv:1711.05136*, 2017.

Ondřej Bojar, Rajen Chatterjee, Christian Federmann, Yvette Graham, Barry Haddow, Shujian Huang, Matthias Huck, Philipp Koehn, Qun Liu, Varvara Logacheva, et al. Findings of the 2017 conference on machine translation (wmt17). Association for Computational Linguistics, 2017.

Alberto Cacciola, Alessandro Muscoloni, Vaibhav Narula, Alessandro Calamuneri, Salvatore Nigro, Emeran A Mayer, Jennifer S Labus, Giuseppe Anastasi, Aldo Quattrone, Angelo Quartarone, et al. Coalescent embedding in the hyperbolic space unsupervisedly discloses the hidden geometry of the brain. *arXiv preprint arXiv:1705.04192*, 2017.

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.

Mauro Cettolo, C. Girardi, and Marcello Federico. Wit3: Web inventory of transcribed and translated talks. *Proceedings of EAMT*, pages 261–268, 01 2012.

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)*, pages 2921–2926. IEEE, 2017.

Hermann Cuntz, Friedrich Forstner, Alexander Borst, and Michael Häusser. One rule to grow them all: a general theory of neuronal branching and its practical application. *PLoS computational biology*, 6(8):e1000877, 2010.

David A Drachman. Do we have brain to spare?, 2005.

Desmond Elliott, Stella Frank, Khalil Sima'an, and Lucia Specia. Multi30k: Multilingual english-german image descriptions. In *Proceedings of the 5th Workshop on Vision and Language*, pages 70–74. Association for Computational Linguistics, 2016. doi: 10.18653/v1/W16-3210. URL http://www.aclweb.org/anthology/W16-3210.

Paul Erdős and Alfréd Rényi. On the evolution of random graphs. *Publication of the Mathematical Institute of the Hungarian Academy of Sciences*, 5:17–60, 1960.

Utku Evci, Trevor Gale, Jacob Menick, Pablo Samuel Castro, and Erich Elsen. Rigging the lottery: Making all tickets winners. In *International conference on machine learning*, pages 2943–2952. PMLR, 2020.

Siddhant Jayakumar, Razvan Pascanu, Jack Rae, Simon Osindero, and Erich Elsen. Top-kast: Top-k always sparse training. *Advances in Neural Information Processing Systems*, 33:20744–20754, 2020.

Suryam Arnav Kalra, Arindam Biswas, Pabitra Mitra, and Biswajit Basu. Sparse neural architectures via deterministic ramanujan graphs. *Transactions on Machine Learning Research*.

Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

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

Mark Kurtz, Justin Kopinsky, Rati Gelashvili, Alexander Matveev, John Carr, Michael Goin, William Leiserson, Sage Moore, Bill Nell, Nir Shavit, and Dan Alistarh. Inducing and exploiting activation sparsity for fast inference on deep neural networks. In Hal Daumé III and Aarti Singh, editors, *Proceedings of the 37th International Conference on Machine Learning*, volume 119 of *Proceedings of Machine Learning Research*, pages 5533–5543, Virtual, 13–18 Jul 2020. PMLR. URL http://proceedings.mlr.press/v119/kurtz20a.html.

Mike Lasby, Anna Golubeva, Utku Evci, Mihai Nica, and Yani Ioannou. Dynamic sparse training with structured sparsity. *arXiv preprint arXiv:2305.02299*, 2023.

Yann LeCun, Léon Bottou, Yoshua Bengio, and Patrick Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 2002.

Namhoon Lee, Thalaiyasingam Ajanthan, and Philip HS Torr. Snip: Single-shot network pruning based on connection sensitivity. *arXiv preprint arXiv:1810.02340*, 2018.

Linyuan Lü, Liming Pan, Tao Zhou, Yi-Cheng Zhang, and H Eugene Stanley. Toward link predictability of complex networks. *Proceedings of the National Academy of Sciences*, 112 (8):2325–2330, 2015.

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

Adam Marcus, Daniel A Spielman, and Nikhil Srivastava. Interlacing families i: Bipartite ramanujan graphs of all degrees. In *2013 IEEE 54th Annual Symposium on Foundations of computer science*, pages 529–537. IEEE, 2013.

Decebal Constantin Mocanu, Elena Mocanu, Peter Stone, Phuong H Nguyen, Madeleine Gibescu, and Antonio Liotta. Scalable training of artificial neural networks with adaptive sparse connectivity inspired by network science. *Nature communications*, 9(1):2383, 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):1615, 2017.

Alessandro Muscoloni, Ilyes Abdelhamid, and Carlo Vittorio Cannistraci. Local-community network automata modelling based on length-three-paths for prediction of complex network structures in protein interactomes, food webs and more. *BioRxiv*, page 346916, 2018.

Alessandro Muscoloni, Umberto Michieli, Yingtao Zhang, and Carlo Vittorio Cannistraci. Adaptive network automata modelling of complex networks. 2022.

Mark EJ Newman. Modularity and community structure in networks. *Proceedings of the national academy of sciences*, 103(23):8577–8582, 2006.

Vithursan Thangarasa, Abhay Gupta, William Marshall, Tianda Li, Kevin Leong, Dennis DeCoste, Sean Lie, and Shreyas Saxena. Spdf: Sparse pre-training and dense fine-tuning for large language models. In *Uncertainty in Artificial Intelligence*, pages 2134–2146. PMLR, 2023.

Christopher A Walsh. Peter huttenlocher (1931–2013). *Nature*, 502(7470):172–172, 2013.

Duncan J Watts and Steven H Strogatz. Collective dynamics of 'small-world' networks. *Nature*, 393(6684):440–442, 1998.

Han Xiao, Kashif Rasul, and Roland Vollgraf. Fashion-mnist: a novel image dataset for benchmarking machine learning algorithms. *arXiv preprint arXiv:1708.07747*, 2017.

Geng Yuan, Xiaolong Ma, Wei Niu, Zhengang Li, Zhenglun Kong, Ning Liu, Yifan Gong, Zheng Zhan, Chaoyang He, Qing Jin, et al. Mest: Accurate and fast memory-economic sparse training framework on the edge. *Advances in Neural Information Processing Systems*, 34:20838–20850, 2021.

Yingtao Zhang, Jialin Zhao, Wenjing Wu, and Alessandro Muscoloni. 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*, 2024a.

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*, 2024b. 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. *arXiv preprint arXiv:2501.19107*, 2025.

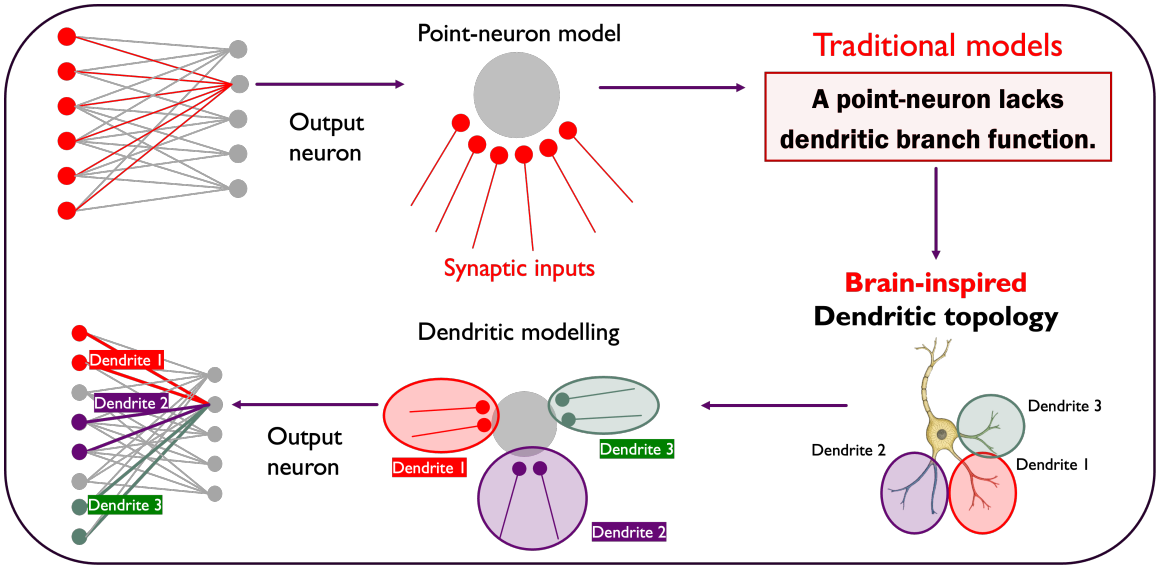


Figure 1: **The Dendritic Network Model.** Traditional models (top) treat neurons as simple integrators, ignoring the structure of synaptic inputs. In contrast, our brain-inspired approach (bottom) organizes connections into dendritic branches, creating a structured topology where distinct groups of inputs are processed locally.

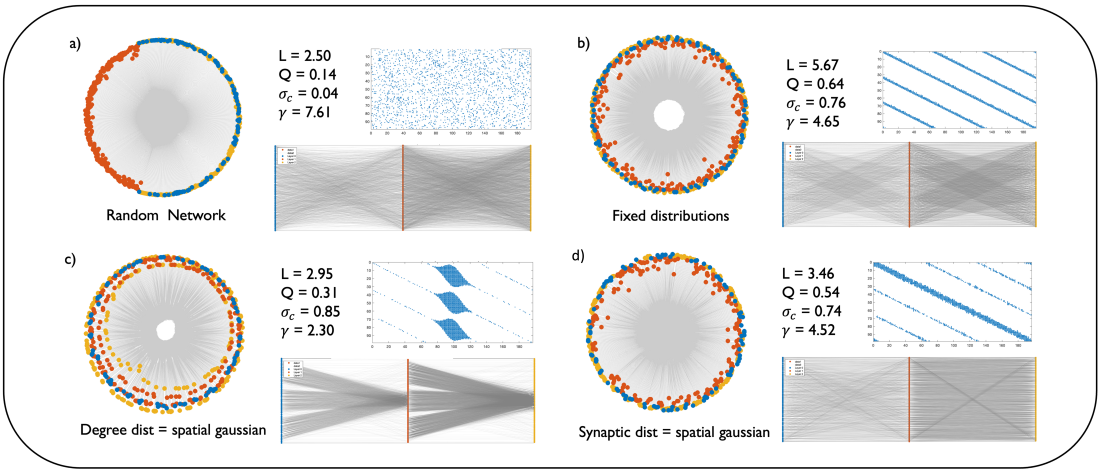


Figure 2: A random network (a) compared with various DNM configurations (b-d) for an MLP of size $98 \times 196 \times 196$ with 90% sparsity. Each panel shows an embedding in hyperbolic space (left), the first layer's adjacency matrix (top right), a bipartite graph representation (bottom right), and key network science metrics: characteristic path length (L), modularity (Q), structural consistency (σ_c), and the power law exponent of the degree distribution (γ).

Appendix A. Glossary of Network Science

In this section, we introduce the basic notions of network science mentioned in this article.

Table 1: Image classification accuracy of statically trained, 99% sparse MLPs with different initial network topologies, compared to the fully-connected (FC) model. The scores are averaged over 3 seeds ± their standard errors. Bold values denote the best performance amongst initialization methods different from CSTI and SNIP.

	Static Sparse Training			
	MNIST	**Fashion MNIST**	**EMNIST**	**CIFAR10**
FC	98.78±0.02	90.88±0.02	87.13±0.04	62.85±0.16
CSTI	98.07±0.02	88.52±0.14	84.66±0.13	52.64±0.30
SNIP	97.59±0.08	87.85±0.22	84.08±0.08	61.81±0.58
Random	96.72±0.04	87.34±0.11	82.66±0.08	55.28±0.09
BSW	97.32±0.02	88.18±0.18	82.94±0.06	56.54±0.15
BRF	96.85±0.01	87.41±0.13	82.98±0.02	54.73±0.07
Ramanujan	96.51±0.17	86.45±0.15	81.80±0.13	55.05±0.40
DNM	**97.82±0.03**	**89.19±0.01**	**84.76±0.13**	**61.63±0.18**

Table 2: Image classification on MNIST, Fashion MNIST, EMNIST, and CIFAR-10 of the SET model on MLPs with 99% sparsity over various topological initialization methods, compared to the fully-connected (FC) model. The scores indicate the accuracy of the models, averaged over 3 seeds ± their standard errors. Bold values denote the best performance amongst initialization methods different from CSTI and SNIP. Sparse models that surpass the fully connected network are marked with "*".

	SET			
	MNIST	**Fashion MNIST**	**EMNIST**	**CIFAR10**
FC	98.78±0.02	90.88±0.02	87.13±0.04	62.85±0.16
SNIP	98.31±0.05	89.49±0.05	86.44±0.09	64.26±0.14
CSTI	98.47±0.02	89.91±0.11	86.66±0.06	65.05±0.14
Random	98.14±0.02	89.00±0.09	86.31±0.08	62.70±0.11
BSW	98.25±0.02	89.25±0.03	86.26±0.08	63.66±0.07*
BRF	98.25±0.01	89.44±0.01	86.20±0.04	62.64±0.10
Ramanujan	97.96±0.05	89.16±0.07	85.80±0.03	62.58±0.20
DNM	**98.40±0.02**	**89.78±0.09**	**86.52±0.02**	**65.67±0.18***

Scale-Free Network A Scale-Free Network (Barabási and Albert, 1999) is characterized by a highly uneven distribution of degrees amongst the nodes. A small number of nodes, called hubs, have a very high degree, and a large number of nodes have very few connections. The degree distribution of scale-free networks follows a power law trend $P(k) \sim k^{\gamma}$, where γ is a constant smaller than 3. In contrast, nodes in random networks are distributed following a Binomial distribution.

Table 3: Image classification on MNIST, Fashion MNIST, EMNIST, and CIFAR-10 of the RigL model on MLPs with 99% sparsity over various topological initialization methods, compared to the fully-connected (FC) model. The scores indicate the accuracy of the models, averaged over 3 seeds ± their standard errors. Bold values denote the best performance amongst initialization methods different from CSTI and SNIP. Sparse models that surpass the fully connected network are marked with "*".

	RigL			
	MNIST	**Fashion MNIST**	**EMNIST**	**CIFAR10**
FC	98.78±0.02	90.88±0.02	87.13±0.04	62.85±0.16
SNIP	98.75±0.04	89.89±0.01	87.34±0.03	63.79±0.23
CSTI	98.76±0.02	90.14±0.10	87.41±0.01	60.52±0.31
Random	98.61±0.01	89.53±0.04	86.99±0.09	64.24±0.07*
BSW	98.71±0.06	89.85±0.07	87.18±0.07*	65.06±0.19*
BRF	98.65±0.03	89.76±0.09	87.08±0.05	64.33±0.16*
Ramanujan	98.42±0.03	89.80±0.07	87.09±0.04	64.31±0.08*
DNM	**98.75±0.04**	**90.08±0.09**	**87.28±0.28***	**65.68±0.01***

Table 4: Image classification on CIFAR-10 of the CHTs and CHTss models on MLPs with 99% sparsity, compared to the fully-connected (FC) model. The scores indicate the accuracy of the models, averaged over 3 seeds ± their standard errors. Bold values denote the best performance amongst initialization methods different from CSTI.

	CSTI	**Random**	**BSW**	**BRF**	**Ramanujan**	**DNM**
FC			62.85±0.16			
CHTs	69.77±0.06	66.96±0.24	66.96±0.24	64.96±0.17	67.19±0.17	**68.76±0.11**
CHTss	71.29±0.14	66.89±0.23	66.89±0.23	64.96±0.17	67.37±0.12	**68.50±0.21**

Watts-Strogatz Model and Small-World Network A Small-World Network (Watts and Strogatz, 1998) is characterized by a small average path length. This property implies that any two nodes can communicate through a short chain of connections. The Watts-Strogatz (Watts and Strogatz, 1998) model is well known for its high clustering and short path lengths. This network is modelled by a parameter β between 0 and 1 that can determine its level of clustering. When β takes low values ($\beta \approx 0$), the WS network is a highly clustered lattice. On the other hand, when β approaches 1, the network becomes a random small-world graph. Intermediate values of β can generate a clustered network that maintains small-world connectivity.

Formally, a network is small-world when the path of length L between two randomly chosen nodes grows proportionally to the logarithm of the number of nodes (N) in the network, that is:

$$L \propto logN. \tag{1}$$

Table 5: Image classification on MNIST, Fashion MNIST, and EMNIST of the CHTs and CHTss models on MLPs with 99% sparsity over various topological initialization methods, compared to the fully-connected (FC) model. The scores indicate the accuracy of the models, averaged over 3 seeds ± their standard errors. Bold values denote the best performance amongst initialization methods different from CSTI. The performances that surpass CSTI are marked with "*".

	MNIST		**Fashion MNIST**		**EMNIST**	
	CHTs	**CHTss**	**CHTs**	**CHTss**	**CHTs**	**CHTss**
FC	98.78±0.02		90.88±0.02		87.13±0.04	
CSTI	98.81±0.04	98.83±0.02	90.93±0.03	90.81±0.11	87.82±0.04	87.52±0.04
Random	98.57±0.04	98.61±0.03	90.42±0.03	90.30±0.10	87.12±0.13	87.20±0.09
BSW	98.57±0.04	98.61±0.04	90.46±0.06	90.46±0.06	87.12±0.13	87.20±0.09
BRF	98.47±0.03	98.47±0.03	90.04±0.12	90.04±0.12	87.03±0.07	87.03±0.07
Ramnujan	98.57±0.03	98.57±0.03	89.82±0.14	98.78±0.08	87.24±0.08	87.24±0.08
DNM	**98.66±0.03**	**98.90±0.01***	**90.68±0.09**	**90.88±0.03***	**87.40±0.04**	**87.55±0.01***

Table 6: Performance comparison on Multi30k en-de and IWSLT en-de translation tasks with varying sparsity levels (95% and 90%). BLEU scores (higher is better) are averaged over 3 seeds ± standard error. Bold indicates best performance for given sparsity and initialization; * denotes scores surpassing the fully connected model.

	Multi30k		IWSLT	
	0.95	0.90	0.95	0.90
FC	31.38±0.38		24.48±0.30	
CHTs^{B}	28.94±0.57	29.81±0.37	21.15±0.10	21.92±0.17
CHTs^{D}	**30.54±0.42**	**31.45±0.35***	**22.09±0.14**	**23.52±0.24**
CHTss^{B}	32.03±0.29*	32.86±0.16*	**24.51±0.02***	**24.31±0.04**
CHTss^{D}	**32.62±0.28***	**33.00±0.31***	24.43±0.14	24.20±0.07

^{D}DNM initialization. ^{B}BRF initialization.

Structural Consistency Structural consistency (Lü et al., 2015) is an index based on the first-order matrix perturbation of the adjacency matrix, which represents the predictability of the network structure. A perturbation set ΔE is randomly sampled from the original link set E. Identifying as E^L the links ranked as the top L according to the structural perturbation method (Lü et al., 2015), with $L = |\Delta E|$, the structural consistency σ_c is calculated as:

$$\sigma_c = \frac{|E^L \cap \Delta E|}{\Delta E}. \tag{2}$$

Modularity Modularity (Newman, 2006) quantifies the tendency of nodes in the network to form distinct communities (or clusters). This measure ranges from -1 to 1. A high modularity score (close to 1) hints at the presence of dense connections between nodes within

Table 7: Performance comparison on machine translation tasks across the WMT en-de dataset with varying final sparsity levels (95% and 90%). Contrary to the BRF model, the DNM model's parameters were transferred from the best-performing combinations of previous tests, avoiding any parameter search. Entries are BLEU scores (higher is better), averaged over 3 seeds ± standard error. Bold values denote the best performance for a given sparsity and initialization.

Model	WMT	
	0.95	0.90
FC	25.52	
CHTs[B]	20.94±0.63	22.40±0.06
CHTs[D]	**21.34±0.20**	**22.56±0.14**
CHTss[B]	23.73±0.43	**24.61±0.14**
CHTss[D]	**24.52±0.12**	24.40±0.15

[D]DNM initialization. [B]BRF initialization.

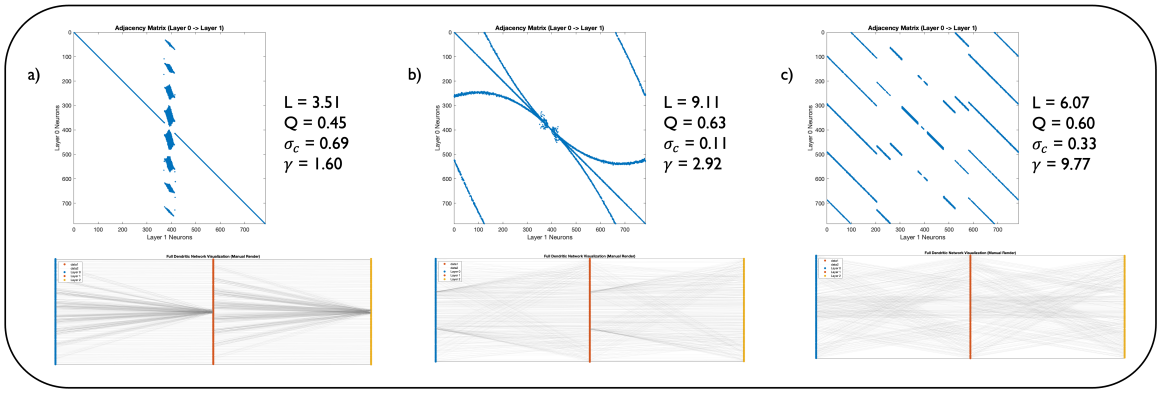
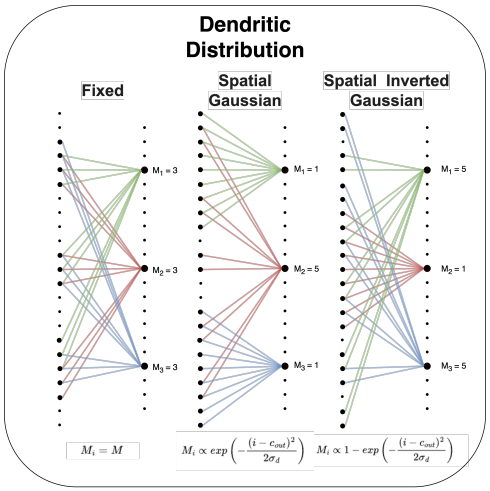
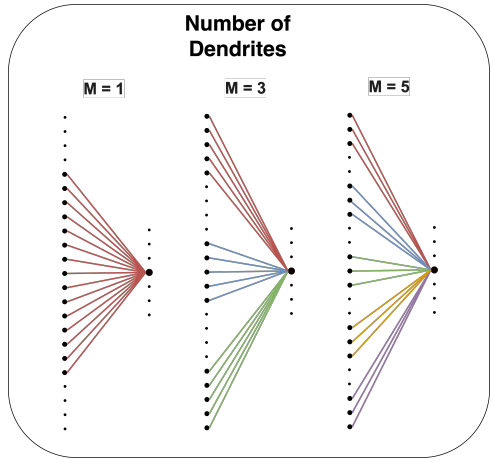


Figure 3: **Representation of the best performing DNM models on image classification.** The figure compares the best performing DNM architectures on MNIST and Fashion MNIST (a), EMNIST (b), and CIFAR10 (c). Each panel shows the network's adjacency matrix (top) and the network's layerwise representation (bottom). Furthermore, each panel exhibits the network's topological measures: characteristic path length L, modularity (Q), structural consistency (σ_c), and the power law exponent of the degree distribution (γ).

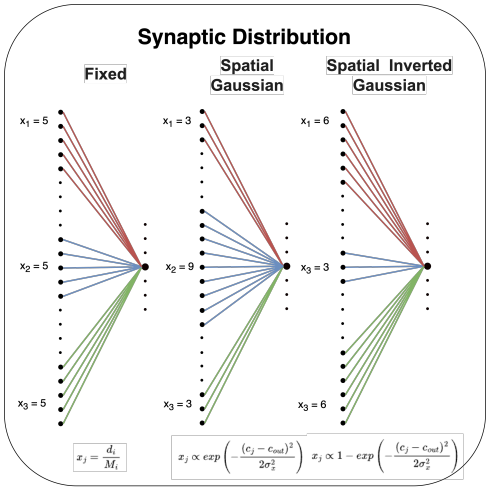
communities, but sparse connections between nodes belonging to different communities. A modularity score close to 0, in contrast, suggests that the network lacks any community organization and the interaction between nodes is essentially uniform. When modularity approaches -1, the network exhibits an anticommunity structure. This means that nodes are strongly connected across the network, and there is little differentiation into separate groups. In other words, a negative modularity represents a cohesive network. The formula



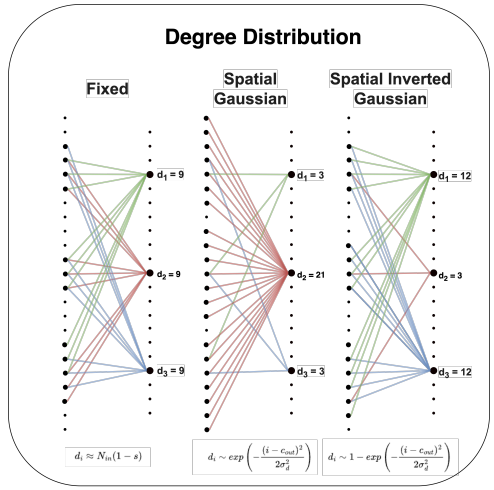
(a) Effect of the dendritic distribution on the network topology.



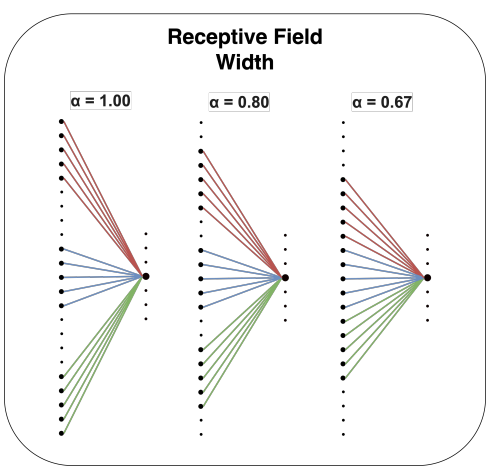
(b) Effect of the number of dendrites on the network topology.



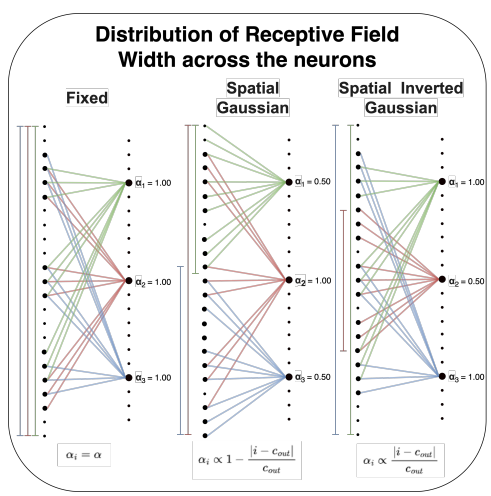
(c) Effect of the synaptic distribution on the network topology.



(d) Effect of the degree distribution on the network topology.

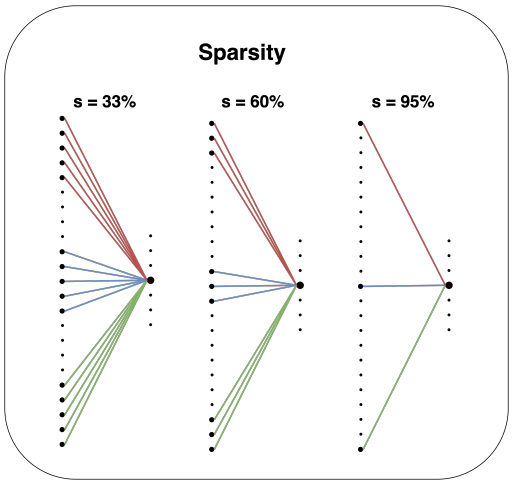


(e) Effect of the receptive field width on the network topology.

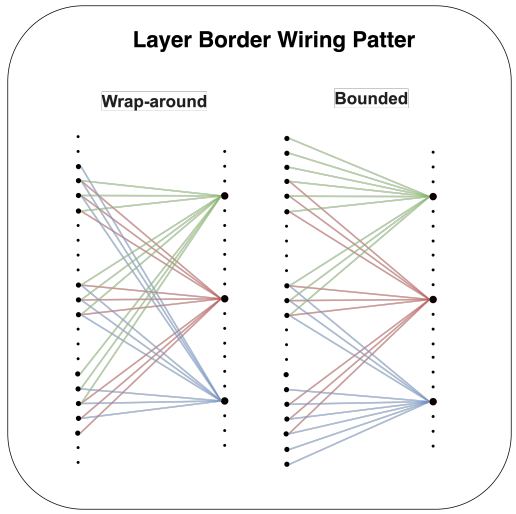


(f) Effect of the receptive field width distribution on the network topology.

Figure 4: Representations of the network's topology obtained by varying a DNM parameter while keeping all others fixed.



(a) Effect of the dendritic distribution on the network topology.



(b) Effect of the layer border setting on the network topology.

Figure 5: Representations of the network's topology obtained by varying a DNM parameter while keeping all others fixed.

to compute the modularity (Q) is:

$$Q = \frac{1}{2m} \sum_{ij} \left[A_{ij} - \frac{k_i k_j}{2m} \right] \delta(c_i, c_j), \tag{3}$$

where A represents the network's adjacency matrix, and k_i and k_j are the degrees of nodes i and j, respectively. $\delta(c_i, c_j)$ is the Kronecker delta function, which equals one if i and j are in the same community, else it equals 0.

Characteristic path length The characteristic path length is computed as the average node-pairs length in the network; it is a measure associated with the network's small-worldness (Cannistraci and Muscoloni, 2022). The characteristic path length (L) is derived by:

$$L = \frac{1}{n(n-1)} \cdot \sum_{i,j} d(i,j), \tag{4}$$

where n is the number of nodes in the network, and d(i,j) is the shortest path length between node i and node j.

Coalescent Embedding Coalescent embedding (Muscoloni et al., 2017) is a class of machine learning algorithms used for unsupervised dimensionality reduction and embedding complex networks in a geometric space, often hyperbolic. This method maps high-dimensional information on a low-dimensional embedding while maintaining the essential topological features of the network. This embedding reveals latent structures of the system, like hierarchical and scale-free structures. In this article, coalescent embedding maps the networks that have latent hyperbolic geometry onto the two-dimensional hyperbolic

space. The approach involves 4 steps: 1) links are pre-weighted with topological rules that approximate the underlying network geometry (Cannistraci and Muscoloni, 2022); 2) non-linear dimensionality reduction; 3) generation of angular coordinates; 4) generation of radial coordinates.

This process is illustrated in Figure 2, which showcases the results of applying a specific coalescent embedding pipeline to four different synthetic networks. The embeddings shown were generated without any initial link pre-weighting (step 1). For the non-linear dimensionality reduction (step 2), the Isomap (Balasubramanian and Schwartz, 2002) algorithm was used. Finally, the angular coordinates (step 3) were determined using Equidistant Adjustment (EA), a process that preserves the relative order of the nodes while arranging them at perfectly uniform angular intervals.

A.1. Sparse Topological Initialization Methods

As mentioned, in dynamic sparse training (DST), a neural network is trained with a sparse topology that evolves throughout the learning process. The initial arrangement of the connections is critically important. This starting structure determines the initial pathways for information flow and acts as the foundational scaffold upon which the network learns and evolves. A well-designed initial topology can significantly improve a model's final performance and training efficiency, whereas a poor starting point can severely hinder its ability to learn effectively. The principal topological initialization approaches for dynamic sparse training are grounded in network science theory, where three basic generative models for monopartite sparse artificial complex networks are the Erdős-Rényi (ER) model (Erdős and Rényi, 1960), the Watts-Strogatz (WS) model (Watts and Strogatz, 1998), and the Barabási-Albert (BA) model (Barabási and Albert, 1999). Since the standard WS and BA models are not directly designed for bipartite networks, they were recently extended into their bipartite counterparts and termed as Bipartite Small-World (BSW) and Bipartite Scale-Free (BSF) (Zhang et al., 2024b), respectively. BSW generally outperforms BSF for dynamic sparse training (Zhang et al., 2024b). The Correlated Sparse Topological Initialization (CSTI) (Zhang et al., 2024a) is a physics-informed topological initialization method that considers the links with the strongest Pearson correlations between nodes and features in the input layer. The Bipartite Receptive Field (BRF) network model (Zhang et al., 2025) generates networks with brain-like receptive field connectivity. This is the first attempt to mimic the structure of brain connections in a sparse network initialization model.

Appendix B. Hyperparameter Settings and Implementation Details

Our experimental setup is designed to replicate the conditions in Zhang et al. (2025). Configurations are assessed on validation sets before being tested on separate test sets. All reported scores are the average of three runs using different random seeds, presented with their corresponding standard errors.

B.1. MLP for Image Classification

Models are trained for 100 epochs using Stochastic Gradient Descent (SGD) with a learning rate of 0.025, a batch size of 32, and a weight decay of 5×10^{-4}. All sparse models are trained at a 99% sparsity level. For dynamic methods, we used SET, CHTs and CHTss. The regrowth strategy for CHTs and CHTss is CH2_L3n (Muscoloni et al., 2018). The CHTss model begins with an initial sparsity of 50% and gradually increases to the final 99% target using an s-shaped scheduler. For our DNM, we conduct a grid search over its key hyperparameters. We tested a mean dendrite count (M) of 3, 7, and 21. For the dendritic, degree, window width, and synaptic distributions, we searched across fixed, spatial Gaussian, and spatial inverted Gaussian options. The mean window width (α) was fixed at 1.0. For the BSW baseline, the rewiring probability is searched in the set {0.0, 0.2, 0.4, 0.6, 0.8, 1.0}. For the BSF baseline, we searched the randomness parameter r over the same set of values, and also tested both fixed and uniform degree distributions.

B.2. Transformer for Machine Translation

We use a standard 6-layer Transformer architecture with 8 attention heads and a model dimension of 512. The dimension of the feed-forward network is set to 1024 for Multi30k and 2048 for IWSLT14 and WMT17. All models are trained using the Adam optimizer (Kingma and Ba, 2014) with the inoam learning rate schedule. Dataset-specific training parameters are as folows:

- Multi30k: Trained for 5,000 steps with a learning rate of 0.25, 1000 warmup steps, and a batch size of 1024.
- IWSLT14: Trained for 20,000 steps with a learning rate of 2.0, 6000 warmup steps, and a batch size of 10240.
- WMT17: Trained for 80,000 steps with a learning rate of 2.0, 8000 warmup steps, and a batch size of 12000.

We evaluated models at final sparsity levels of 90% and 95%. The CHTss models started from a denser state (50% sparsity) and decayed to the target sparsity. For Multi30k and IWSLT14, we performed a comprehensive hyperparameter search. The search for IWSLT14 included a mean dendrite count $M \in \{3, 7, 21\}$ and various combinations of fixed and spatial distributions for all DNM parameters. For WMT17, to assess generalization, we did not perform a new search. Instead, we directly applied the best-performing DNM configuration identified from the IWSLT14 experiments. This configuration used M=7 with a fixed dendritic distribution, alongside spatial Gaussian or inverse-Gaussian distributions for degree, window width, and synapses.

Appendix C. Sensitivity Tests

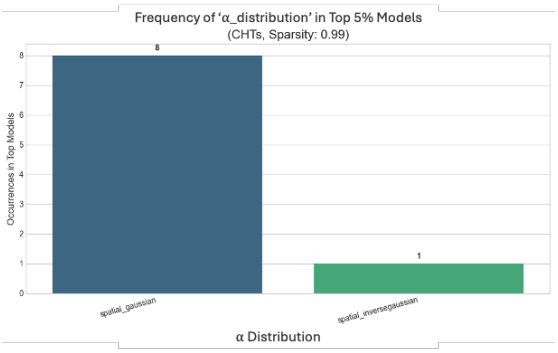
We provide sensitivity tests for DNM hyperparameters. First, we focus on the analysis of CHTs and CHTss on MLPs for image classification at 99% sparsity. Next, we study the parametric configurations for the CHTs and CHTss model on the Multi30k translation task at 90% sparsity. For each task, we vary one parameter at a time, keeping the others fixed

to a specific configuration. We calculate the coefficient of variation (CV) of the scores to quantify the sensitivity of the model to each parameter. A low CV indicates that the model's performance is relatively stable across different settings of that parameter, suggesting low sensitivity. Conversely, a high CV suggests that the model's performance is more variable and sensitive to changes in that parameter. We average each parameter's CV across various parametric configurations to obtain a robust measure of sensitivity. Next, we analyse the top 5% best-performing configurations for each task to understand the commonalities in the optimal settings. This method not only helps us understand which parameters are most influential but also guides future configurations for similar tasks.

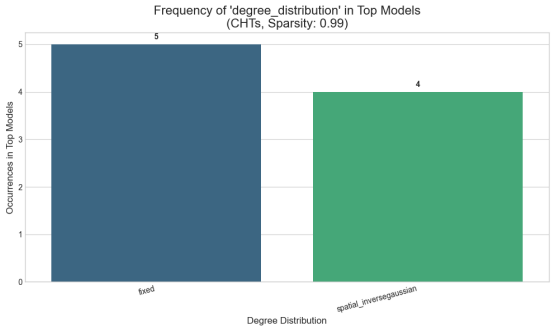
Table 8: Sensitivity Analysis Results for CHTs and CHTss on MNIST, EMNIST, Fashion MNIST, CIFAR-10 for Image Classification. The table presents the average coefficient of variation (CV) for each DNM parameter across different configurations at 99% sparsity level. A higher CV indicates greater sensitivity of the model's performance to changes in that parameter.

	MNIST		**EMNIST**		**Fashion MNIST**		**CIFAR-10**	
Parameter	**CHTs**	**CHTss**	**CHTs**	**CHTss**	**CHTs**	**CHTss**	**CHTs**	**CHTss**
Degree Dist	0.001538	0.001538	0.001419	0.009172	0.003379	0.017639	0.017639	0.017639
Rec Field Width Dist	0.000506	0.000506	0.001585	0.006154	0.001205	0.005381	0.005381	0.005381
Dendritic Dist	0.000525	0.000525	0.001265	0.003620	0.001136	0.004983	0.004983	0.004983
Synaptic Dist	0.000464	0.000464	0.000989	0.002538	0.000964	0.004306	0.004306	0.004306

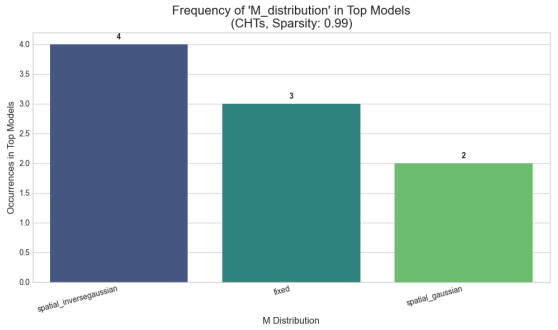
Dynamic Sparse Training for Image Classification We analyse the sensitivity of DNM parameters for the initialization of CHTs and CHTss models on MLPs for image classification at 99% sparsity. The analysis is performed over MNIST, Fashion MNIST, EMNIST, and CIFAR-10. The sensitivity analysis, summarized in Table 8, evaluates the impact of DNM initialization parameters for CHTs and CHTss models at 99% sparsity. A key finding is the significantly heightened sensitivity of the CHTss model compared to CHTs, a disparity that amplifies with increasing dataset complexity from MNIST to CIFAR-10. Across all benchmarks, the Degree Distribution consistently emerges as the most critical parameter, highlighting the paramount importance of initial network connectivity. Following in descending order of influence are the Receptive Field Width, Dendritic, and Synaptic distributions. These results underscore that while both architectures show similar robustness on simpler tasks, the effective deployment of the CHTss model on more challenging problems is critically dependent on the precise calibration of its initial topological structure. Analyzing the top 5% best-performing configurations, we observe similar trends across various datasets (Figure 7). The most relevant findings is that a spatial gaussian receptive field width distribution is constantly preferred.



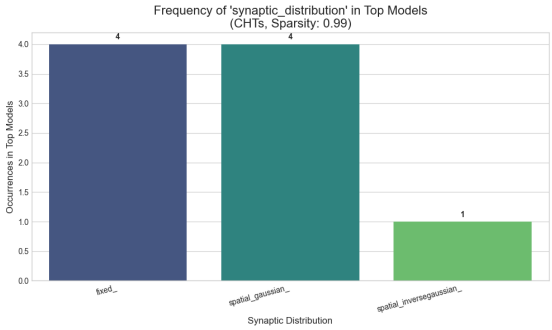
(a) Sensitivity analysis of the receptive field width distribution



(b) Sensitivity analysis of the degree distribution



(c) Sensitivity analysis of the dendritic distribution



(d) Sensitivity analysis of the synaptic distribution

Figure 6: Sensitivity analysis of various DNM hyperparameter distributions.

Dynamic Sparse Training for Machine Translation We focus on the analysis of DNM for the initialization of the CHTss model on Multi-30k for machine translation. Both at 90% and 95% sparsity, the calculated coefficients of variation do not surpass 0.01. This indicates that the model's performance is relatively stable across different settings of each parameter, suggesting low sensitivity (Table 9). At both sparsity levels, the dendritic distribution appears to be the most sensitive, whereas M and α are the most stable. Analyzing the top 5% best-performing configurations, we observe that spatial distributions generally outperform fixed distributions (Figure 7).

Appendix D. Modelling Dendritic Networks

The Dendritic Network Model (DNM) generates the sparse connectivity matrix of sandwich layers by iteratively building connections for each output neuron based on the principles of dendritic branching and localized receptive fields. The generation process can be broken down into the following steps:

1. Determine the degree of each output neuron in the layer based on one of the three distribution strategies (fixed, non-spatial, spatial). A probabilistic rounding and ad-

Table 9: Sensitivity Analysis Results for CHTss on Multi30k for Machine Translation. The table presents the average coefficient of variation (CV) for each DNM parameter across different configurations at 90% and 95% sparsity levels. A higher CV indicates greater sensitivity of the model's performance to changes in that parameter.

Parameter	**Average Coefficient of Variation (CV)**	
	90%	**95%**
Dendritic distribution	0.010643	0.009340
Degree distribution	0.010270	0.008436
Receptive Field Width Distribution	0.009830	0.009103
Synaptic distribution	0.009663	0.009310
M	0.008959	0.007806
α	0.008210	0.007461

Note: Higher CV indicates greater impact on performance.

justment mechanism ensures that no output neuron is disconnected and the sampled degrees sum precisely to the target total number of connections of the layer.

2. Next, for each output neuron j, define a receptive field. This is done by topologically mapping the output neuron's position at a central point in the input layer and establishing a receptive window around this center. The size of this window is controlled by the parameter $\alpha_j \in [0, 1]$, which determines the fraction of the input layer that the neuron can connect to. α_j itself can be fixed or sampled from a spatial or non-spatial distribution.

3. For each output neuron, determine the number of dendritic branches, M_j to be used to connect it to the input layer. Again, M_j is determined based on one of the three configurations (fixed for all neurons, or sampled from a distribution that could depend on the neuron's position in the layer).

4. Place the M_j dendrites as dendritic centers within the neuron's receptive window, spacing them evenly across the window.

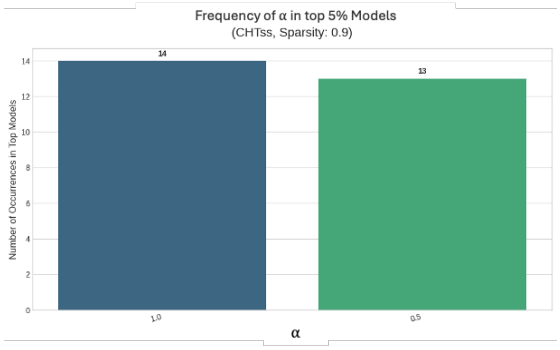
5. The neuron's total degree, obtained from step 1, is distributed across its M_j dendrites according to a synaptic distribution (fixed, spatial, or non-spatial). For each dendrite, connections are formed with the input neurons that are spatially closest to its center.

6. Finally, the process ensures connection uniqueness and adherence to the precise degree constraints.

Appendix E. Dynamic Sparse Training (DST)

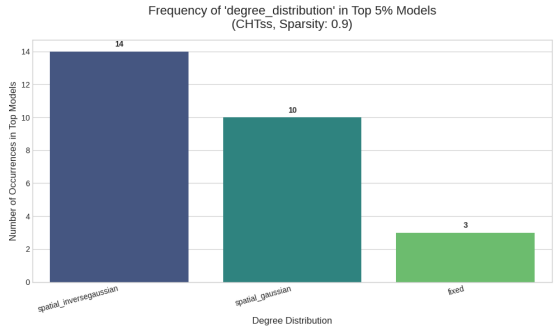
E.1. Dynamic Sparse Training

Dynamic sparse training (DST) is a subset of sparse training methodologies that allows for the evolution of the network's topology during training. Sparse Evolutionary Training (SET) (Mocanu et al., 2018) is the pioneering method in this field, which iteratively

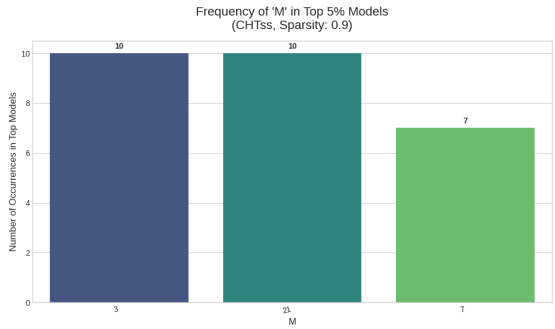


(a) Sensitivity analysis of α.

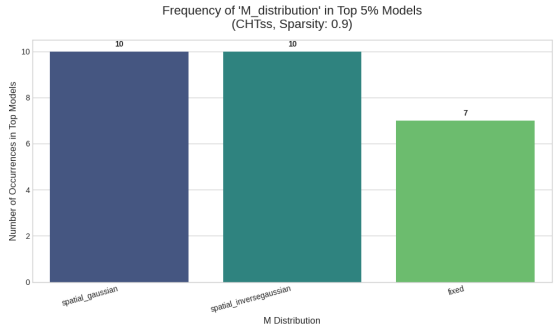
(b) Sensitivity analysis of the receptive field width distribution.



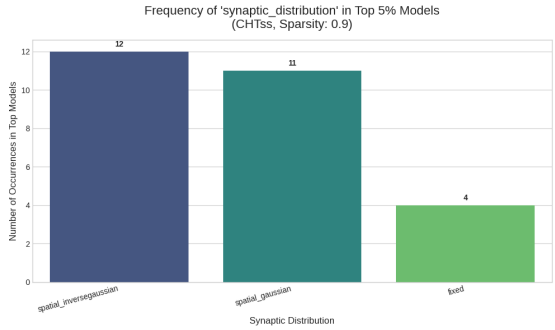
(c) Sensitivity analysis of the degree distribution.



(d) Sensitivity analysis of M.



(e) Sensitivity analysis of the dendritic distribution.



(f) Sensitivity analysis of the synaptic distribution.

Figure 7: Sensitivity analysis of the DNM parameters for CHTss initialization over Transformer models for machine translation on Multi30k at 90% sparsity.

removes links based on the absolute magnitude of their weights and regrows new connections randomly. Subsequent developments have expanded upon this method by refining the pruning and regrowth steps. One such advancement was proposed by Deep R (Bellec et al., 2017), a method that evolves the network's topology based on stochastic gradient updates

combined with a Bayesian-inspired update rule. RigL (Evci et al., 2020) advanced the field further by leveraging the gradient information of non-existing links to guide the regrowth of new connections. MEST (Yuan et al., 2021) is a method that exploits information on the gradient and the weight magnitude to selectively remove and randomly regrow new links, similarly to SET. MEST introduces the EM&S technique that gradually decreases the density of the network until it reaches the desired sparsity level. Top-KAST (Jayakumar et al., 2020) maintains a constant sparsity level through training, iteratively selecting the top K weights based on their magnitude and applying gradients to a broader subset of parameters. To avoid the model being stuck in a suboptimal sparse subset, Top-KAST introduces an auxiliary exploration loss that encourages ongoing adaptation of the mask. A newer version of RigL, sRigL (Lasby et al., 2023), adapts the principles of the original model to semi-structured sparsity, speeding up the training from scratch of vision models. CHT (Zhang et al., 2024b) is the state-of-the-art (SOTA) dynamic sparse training framework that adopts a gradient-free regrowth strategy that relies solely on topological information (network shape intelligence). This model suffers from two main drawbacks: it has time complexity $\mathcal{O}(N \cdot d^3)$ (N node network size, d node degree), and it rigidly selects top link prediction scores, which causes suboptimal link removal and regrowth during the early stages of training. For this reason, this model was evolved into CHTs (Zhang et al., 2025), which adopts a flexible strategy to sample connections to remove and regrow, and reduces the time complexity to $\mathcal{O}(N^3)$. The same authors propose a sigmoid-based gradual density decay strategy, namely CHTss (Zhang et al., 2025), which proves to be the state-of-the-art dynamic sparse training method over multiple tasks. CHTs and CHTss can surpass fully connected MLP, Transformer, and LLM models over various tasks using only a small fraction of the networks' connections.

Appendix F. Experiments

F.1. Experimental Setup

To test the basic advantage of DNM over other initialization methods, we evaluate it for static sparse training using MLPs for image classification tasks on the MNIST (LeCun et al., 2002), Fashion MNIST (Xiao et al., 2017), EMNIST (Cohen et al., 2017), and CIFAR10 (Krizhevsky, 2009) datasets. To further validate our approach, we replicate the tests on CHTs (Zhang et al., 2025) and CHTss, which are the state-of-the-art dynamic sparse training (DST) methods. We use the SET (Mocanu et al., 2018) dynamic sparse training method as a baseline, which regrows new connections randomly. Finally, we test DNM for initialization of the CHTs and CHTss, on Transformers for machine translation tasks on the Multi30k en-de (Elliott et al., 2016), IWSLT14 en-de (Cettolo et al., 2012), and WMT17 (Bojar et al., 2017) datasets. For MLP training, we sparsify all layers except the final layer, as it would lead to disconnected output neurons, and connections in the final layer are relatively fewer compared to the previous layers. For Transformers, we apply DST to all linear layers, excluding the embedding and final generator layer. The hyperparameter settings of the experiments are detailed in Appendix B. Furthermore, we conduct a series of sensitivity tests on the hyperparameters of the DNM in Appendix C.

Baseline methods We compare the DNM network with the baseline approaches in the literature. On static sparse training, we compare DNM with the Erdős-Rényi (ER) model (Erdős and Rényi, 1960), the Bipartite Small World (BSW) (Zhang et al., 2024b) and Bipartite Receptive Field (BRF) (Zhang et al., 2025) initialization techniques. We use CSTI (Zhang et al., 2024a) as the baseline model, although this comparison is inherently unfair due to its feature-informed nature. For dynamic sparse training, we compare DNM with BRF, BSW, and CSTI. Finally, for tests on Transformer models, we compare DNM with the BRF initialization.

F.2. MLP for Image Classification

Static sparse training As an initial evaluation of DNM's performance, we compare it to other topological initialization methods for static sparse training for image classification tasks. On all benchmarks, DNM outperforms the baseline models, as shown in Table 1. Analyzing the best-performing DNM networks is crucial to understanding the relationship between network topology and task performance. This aspect is assessed in Section 2.3.

Dynamic sparse training We first test DNM on the baseline dynamic sparse training method, SET, which regrows links randomly. As shown in Table 2, DNM outperforms the other sparse initialization methods of MLPs (99% sparsity) over all the datasets tested. The model's performance is comparable to the input-informed CSTI, highlighting that DNM's high degree of freedom can match a topology induced by data features.

Tables 4 and 5 repeat the previous tests on the state-of-the-art DST methods, CHTs and CHTss. The results confirm our conclusion: not only DNM exhibits high performance for this task, but it can also surpass the input-informed CSTI method.

F.3. Transformer for Machine Translation

We assess the Transformer's performance on a machine translation task across three datasets. We take the best performance of the model on the validation set and report the BLEU score on the test set. Beam search, with a beam size of 2, is employed to optimize the evaluation process. On the Multi30k and IWSLT datasets, we conducted a thorough hyperparameter search to find the best settings for our DNM model. For the WMT dataset, however, we simply used the best settings found in the previous tests. This approach was to verify that DNM performs well even without extensive, dataset-specific tuning. DNM markedly improves the performance of the CHTs algorithm (Table 10, 11). However, its impact on CHTss is similar to the BRF baseline. This is expected, as CHTss's gradual decay mechanism starts from a dense 50% state, where the initial network structure is less critical and the topological advantages of DNM are less pronounced.

Appendix G. Baseline Methods

We describe in detail the models compared in our experiments.

Table 10: Performance comparison on Multi30k en-de and IWSLT en-de translation tasks with varying sparsity levels (95% and 90%). BLEU scores (higher is better) are averaged over 3 seeds ± standard error. Bold indicates best performance for given sparsity and initialization; * denotes scores surpassing the fully connected model.

	Multi30k		IWSLT	
	0.95	0.90	0.95	0.90
FC	31.38±0.38		24.48±0.30	
$CHTs^B$	28.94±0.57	29.81±0.37	21.15±0.10	21.92±0.17
$CHTs^D$	**30.54±0.42**	**31.45±0.35***	**22.09±0.14**	**23.52±0.24**
$CHTss^B$	32.03±0.29*	32.86±0.16*	**24.51±0.02***	**24.31±0.04**
$CHTss^D$	**32.62±0.28***	**33.00±0.31***	24.43±0.14	24.20±0.07

^{D}DNM initialization. ^{B}BRF initialization.

Table 11: Performance comparison on machine translation tasks across the WMT en-de dataset with varying final sparsity levels (95% and 90%). Contrary to the BRF model, the DNM model's parameters were transferred from the best-performing combinations of previous tests, avoiding any parameter search. Entries are BLEU scores (higher is better), averaged over 3 seeds ± standard error. Bold values denote the best performance for a given sparsity and initialization.

Model	WMT	
	0.95	0.90
FC	25.52	
$CHTs^B$	20.94±0.63	22.40±0.06
$CHTs^D$	**21.34±0.20**	**22.56±0.14**
$CHTss^B$	23.73±0.43	**24.61±0.14**
$CHTss^D$	**24.52±0.12**	24.40±0.15

^{D}DNM initialization. ^{B}BRF initialization.

G.1. Sparse Network Initialization

Bipartite Scale-Free (BSF) The Bipartite Scale-Free network (Zhang et al., 2024b) is an extension of the Barabási-Albert (BA) model (Barabási and Albert, 1999) to bipartite networks. We detail the steps to generate the network. 1) Generate a BA monopartite network consisting of $m + n$ nodes, where m and n are the numbers of nodes of the first and second layer of the bipartite network, respectively. 2) Randomly select m and n nodes to assign to the two layers. 3) Count the number of connections between nodes within the same layer (frustrations). If the two layers have an equal number of frustrations, match each node in layer 1 with a frustration to a node in layer 2 with a frustration, randomly. Apply a single rewiring step using the Maslov-Sneppen randomization (MS) procedure for every matched pair. If the first layer counts more frustrations, randomly sample a subset of layer 1 with the same number of frustrations, and repeat step 1. For each remaining

frustration in layer 1, sequentially rewire the connections to the opposite layer using the preferential attachment method from step 1. If the second layer has more frustrations than the first, apply the opposite procedure. The resulting network will be bipartite and exhibit a power-law distribution with exponent $\gamma = 2.76$.

Bipartite Small-World (BSW) The Bipartite Small-World network (Zhang et al., 2024b) is an extension of the Watts-Strogatz model to bipartite networks. It is modelled as follows: 1) Build a regular ring lattice with a number of nodes $N = \#L_1 + \#L_2$, with $L_1 > L_2$, where L_1 and L_2 represent the nodes in the first and second layers of the bipartite network, respectively. 2) Label the N nodes in a way such that for every L_1 node positioned in the network, $\#L_1/\#L_2$ nodes from L_2 are placed at each step. Then, at each step, establish a connection between an L_1 node and the $K/2$ closest L_2 neighbours in the ring lattice. 3) For every node, take every edge connecting it to its $K/2$ rightmost neighbors, and rewire it with a probability β, avoiding self-loops and link duplication. When $\beta = 1$, the generated network corresponds to a random graph.

Correlated Sparse Topological Initialization (CSTI) The Correlated Sparse Topological Initialization (CSTI) (Zhang et al., 2024b) initializes the topology of the layers that interact directly with the input features. The construction of CSTI follows four steps. 1) **Vectorization:** Denoting as n the number of randomly sampled input data from the training set and as M the number of valid features with variance different from zero among these samples, we build an $n \times M$ matrix. 2) **Feature selection:** We perform feature selection by calculating the Pearson Correlation for each feature. Hence, we construct a correlation matrix. 3) **Connectivity selection:** Next, we construct a sparse adjacency matrix, with entries "1" corresponding to the top-k% values from the correlation matrix (where the value of k depends on the desired sparsity level). A scaling factor $\times$ determines the dimension of the hidden layer. 4) **Assembling topologically hubbed network blocks:** Finally, the selected adjacency matrix masks the network to form the initialized topology for each sandwich layer.

SNIP SNIP (Lee et al., 2018) is a static sparse initialization method that prunes connections based on their sensitivity to the loss function. The sensitivity of a connection is defined as the absolute value of the product of its weight and the gradient of the loss with respect to that weight, evaluated on a small batch of training data. Connections with the lowest sensitivity are pruned until the desired sparsity level is reached. This method allows for the identification of important connections before training begins, enabling the training of sparse networks from scratch.

Ramanujan Graphs Ramanujan Graphs (Lubotzky et al., 1988) are a class of optimal expander graphs that exhibit excellent connectivity properties. They are characterized by their spectral gap, which is the difference between the largest and second-largest eigenvalues of their adjacency matrix. A larger spectral gap indicates better expansion properties, meaning that the graph is highly connected and has a small diameter. Ramanujan graphs are constructed using deep mathematical principles from number theory and algebraic geometry. They are known for their optimal expansion properties, making them ideal for applications in network design, error-correcting codes, and computer science. In this article, we use bipartite Ramanujan graphs as a sparse initialization method for neural networks.

In our experiments, we built bipartite Ramanujan graphs as a theoretically-grounded initialization method, following the core principles outlined by Kalra et al.. Drawing inspiration from the findings of Marcus et al. (2013), which prove the existence of bipartite Ramanujan graphs for all degrees and sizes, we constructed these graphs as follows:

- Generate d random permutation matrices, where d is the desired degree of the graph. Each permutation matrix represents a perfect matching, which is a set of edges that connects each node in one layer to exactly one node in the other layer without any overlaps.
- Iteratively combine these matchings. In each step, deterministically decide whether to add or subtract the successive matching to the current adjacency matrix. This decision is made by minimizing a *barrier function* that ensures that the eigenvalues remain within the Ramanujan bounds.

Bipartite Receptive Field (BRF) The Bipartite Receptive Field (BRF) Zhang et al. (2025) is the first sparse topological initialization model that generates brain-network-like receptive field connectivity. The BRF directly generates sparse adjacency matrices with a customized level of spatial-dependent randomness according to a parameter $r \in [0, 1]$. A low value of r leads to less clustered topologies. As r increases towards 1, the connectivity patterns tend to be generated uniformly at random. Specifically, when r tends to 0, BRF builds adjacency matrices with links near the diagonal (adjacent nodes from the two layers are linked), whereas when r increases 1, this structure tends to break.

Mathematically, consider an $N \times M$ bipartite adjacency matrix $M_{i,j_{i=1,\ldots,M,\ j=1,\ldots,N}}$, where M represents the input size and n represents the output size. Each entry $m_{i,j}$ of the matrix is set to 1 if node i from the input layer connects to node j of the output layer, and 0 otherwise. Define the scoring function

$$S_{i,j} = d_{ij}^{\frac{1}{1-r}}, \tag{5}$$

where

$$d_{ij} = min\{|i - j|, |(i - M) - j|, |i - (j - N)|\} \tag{6}$$

is the distance between the input and output neurons. S_{ij} represents the distance of an entry of the adjacency matrix from the diagonal, raised to the power of $\frac{1-r}{r}$. When the parameter r tends to zero, the scoring function becomes more deterministic; when it tends to 1, all scores $S_{i,j}$ become more uniform, leading to a more random adjacency matrix.

The model is enriched by the introduction of the degree distribution parameter. The Bipartite Receptive Field with fixed sampling (BRFf) sets the degree of all output neurons to be fixed to a constant value. The Bipartite Receptive Field with uniform sampling, on the other hand, samples the degrees of output nodes from a uniform distribution.

G.2. Dynamic Sparse Training (DST)

SET (Mocanu et al., 2018) At each training step, SET removes connections based on weight magnitude and randomly regrows new links.

RigL (Evci et al., 2020) At each training step, RigL removes connections based on weight magnitude and regrows new links based on gradient information.

CHTs and CHTss (Zhang et al., 2025) Cannistraci-Hebb Training (CHT)(Muscoloni et al., 2022) is a brain-inspired gradient-free link regrowth method. It predicts the existence and the likelihood of each nonobserved link in a network. The rationale is that in complex networks that have a local-community structure, nodes within the same community tend to activate simultaneously ("fire together"). This co-activation encourages them to form new connections among themselves ("wire together") because they are topologically isolated. This isolation, caused by minimizing links to outside the community, creates a barrier that reinforces internal signaling. This strengthened signaling, in turn, promotes the creation of new internal links, facilitating learning and plasticity within the community. CHTs enhances this gradient-free regrowth method by incorporating a soft sampling rule and a node-based link-prediction mechanism. CHTss further refines this process by integrating a sigmoid gradual density decay strategy, which lowers the sparsity of a network through epochs.

Appendix H. Impact of initial sparsity on DNM performance

As demonstrated in Subsection **??**, DNM's impact on CHTss is similar to the BRF baseline. This is expected, since CHTss's gradual decay mechanism starts from a dense 50% state, where the initial network structure is less critical and the topological advantages of DNM are less pronounced. To prove that at higher sparsities DNM can largely outperform BRF, we tested both models on the Multi30k translation task at 95% final sparsity and various initial sparsity levels (Table 12, Figure 8). As expected, when starting from a sparser state, DNM outperforms BRF by a larger margin. This suggests that DNM is particularly effective when the initial network structure plays a more significant role in determining performance, as is the case at higher sparsity levels. To further validate this, we fixed DNM's parametric configuration across initial sparsity levels, using the best performing parametric configuration found from the experiment conducted in Subsection **??**. The results (Table 13, Figure 9) confirm that DNM consistently outperforms BRF across all tested initial sparsity levels, with the performance gap widening as the initial sparsity increases. This reinforces the conclusion that DNM's structured approach to sparse connectivity is particularly advantageous in scenarios where the initial network topology is crucial for learning efficacy.

Table 12: Comparison of DNM and BRF initializations on CHTss for machine translation on the Multi-30k en-de dataset at 95% final sparsity. The results for both DNM and BRF correspond to those of the best-performing parametric configurations of the models for each initial sparsity. The initial sparsity level is varied, taking values 0.50, 0.60, 0.75, 0.90. The entries represent the BLEU scores, averaged over 3 seeds ± the standard error.

With parameter search				
	0.50	**0.60**	**0.75**	**0.90**
DNM	32.71±0.35	**32.71±0.43**	**32.28±0.29**	**31.45±0.35**
BRF	**32.86±0.16**	32.16±0.14	31.20±0.36	29.81±0.37

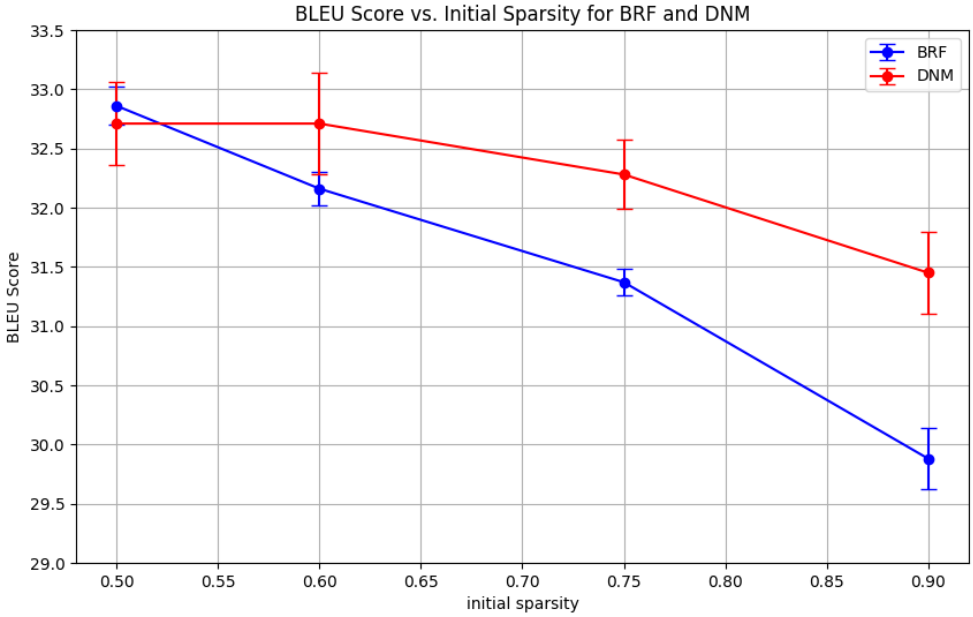


Figure 8: **CHTss with varying initial sparsity and fixed DNM parameters.** The figure compares the performance of DNM and BRF initializations for CHTss over various initial sparsity levels for machine translation tasks on Multi-30k at 90% final sparsity. The results for both DNM and BRF correspond to those of the best performing parametric configurations of the models for each initial sparsity. The entries represent the BLEU scores, averaged over 3 seeds.

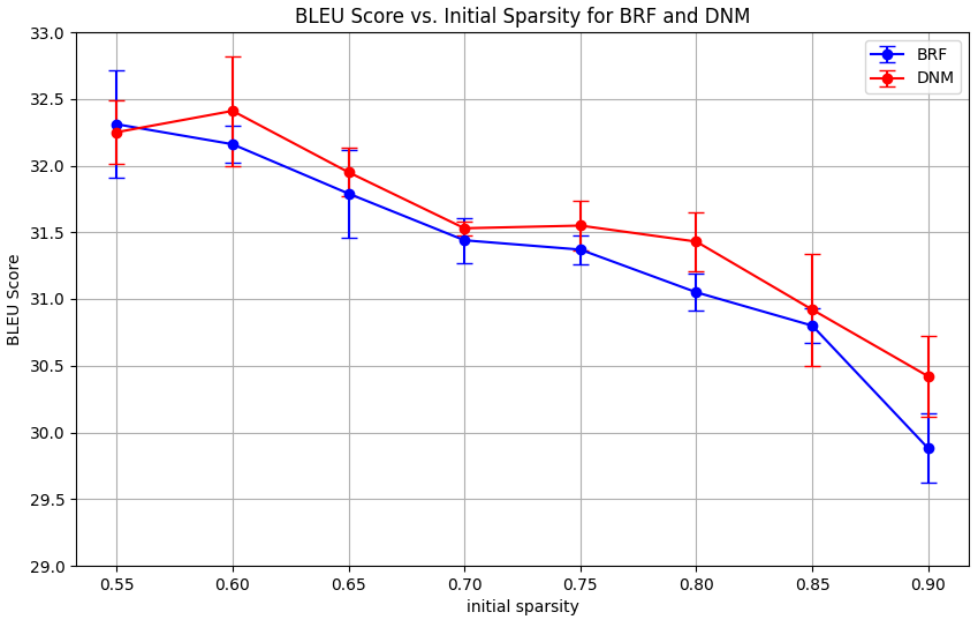


Figure 9: **CHTss with varying initial sparsity.** The figure compares the performance of DNM and BRF initializations for CHTss over various initial sparsity levels for machine translation tasks on Multi-30k at 90% final sparsity. DNM's parametric configuration is fixed across initial sparsity levels, using the best performing parametric configuration found from the experiment performed in Subsection **??**. BRF's performances are chosen taking the best performing parametric configurations. The entries represent the BLEU scores, averaged over 3 seeds.

Table 13: Comparison of DNM and BRF initializations on CHTss for machine translation on the Multi-30k en-de dataset at 95% final sparsity. DNM's parametric configuration is fixed across initial sparsity levels, using the best performing parametric configuration found from the experiment performed in Subsection **??**. BRF's performances are chosen taking the best performing parametric configurations. The initial sparsity level is varied at steps of 0.05 between 0.55 and 0.90. The entries represent the BLEU scores, averaged over 3 seeds.

No DNM parameter search								
	0.55	**0.60**	**0.65**	**0.70**	**0.75**	**0.80**	**0.85**	**0.90**
DNM	32.25	**32.41**	**31.95**	**31.53**	**31.55**	**31.43**	**30.92**	**30.42**
BRF	**32.31**	32.16	31.79	31.44	31.37	31.05	30.80	29.88

Appendix I. Analysis of the Bounded Layer Border Wiring Pattern

To investigate the impact of the layer border wiring pattern setting, we compare the default wrap-around topology with the bounded topology. We tested the bounded configuration on the same image classification tasks as the default, at 99% sparsity using static sparse training (Table 14), SET (Table 15), CHTs (Table 16), and CHTss (Table 17). Overall, the performance of the bounded model is comparable to that of the default wrap-around model, with some variations across datasets and training methods. The bounded model outperforms the default on Fashion MNIST and CIFAR10 when using static sparse training and SET, while the default has a slight edge on MNIST and EMNIST. For CHTs and CHTss, the results are mixed, with each model excelling in different datasets. These findings suggest that while strict locality can be beneficial in certain contexts, the flexibility of wrap-around connections may provide advantages in others. The choice of wiring pattern should thus be informed by the specific characteristics of the task and dataset at hand.

Table 14: Comparison of wrap-around and bounded topology performances for static sparse training of MLPs. The entries represent the accuracy for image classification over different datasets at 99% sparsity, averaged over 3 seeds ± their standard error.

Static Sparse Training				
	MNIST	**EMNIST**	**Fashion_MNIST**	**CIFAR10**
Bounded	97.64±0.10	84.00±0.06	**89.19±0.01**	**61.63±0.18**
Wrap-around	**97.82±0.03**	**84.76±0.13**	88.47±0.03	59.04±0.17

Appendix J. Transferability of Optimal Topologies

A critical question for our generative model of network topology is whether its principles are generalizable across different tasks. To investigate this, we conducted a transfer learning experiment to assess if an optimal topology discovered on a simpler task could be effectively applied to more complex ones. This tests the hypothesis that the DNM can capture fun-

Table 15: Comparison of wrap-around and bounded topology performances for SET initialization on MLPs. The entries represent the accuracy for image classification over different datasets at 99% sparsity, averaged over 3 seeds ± their standard error.

SET				
	MNIST	**EMNIST**	**Fashion_MNIST**	**CIFAR10**
Bounded	**98.40±0.02**	**86.52±0.02**	**89.78±0.09**	**65.67±0.18**
Wrap-around	98.36±0.05	86.50±0.06	89.75±0.04	64.81±0.01

Table 16: Comparison of wrap-around and bounded topology performances for CHTs initialization on MLPs. The entries represent the accuracy for image classification over different datasets at 99% sparsity, averaged over 3 seeds ± their standard error.

CHTs				
	MNIST	**EMNIST**	**Fashion_MNIST**	**CIFAR10**
Bounded	**98.66±0.03**	87.35±0.00	**90.68±0.09**	68.03±0.14
Wrap-around	98.62±0.01	**87.40±0.04**	90.62±0.16	**68.76±0.11**

Table 17: Comparison of wrap-around and bounded topology performances for CHTss initialization on MLPs. The entries represent the accuracy for image classification over different datasets at 99% sparsity, averaged over 3 seeds ± their standard error.

CHTss				
	MNIST	**EMNIST**	**Fashion_MNIST**	**CIFAR10**
Bounded	98.66±0.03	87.37±0.04	90.68±0.09	68.03±0.14
Wrap-around	**98.90±0.01**	**87.55±0.01**	**90.88±0.03**	**68.50±0.21**

damental structural priors beneficial for a class of problems, such as image classification, thereby reducing the need for extensive hyperparameter searches on every new dataset.

Experimental Design We identified the best-performing DNM hyperparameter configuration from the static sparse training experiments on the MNIST dataset, as analyzed in Section 2.3. This configuration, which generates a scale-free, hierarchical network, was then used directly to initialize MLP models for the more challenging EMNIST, Fashion MNIST, and CIFAR-10 datasets. We then compared its performance against the other baseline initialization methods and against the DNM models whose hyperparameters were specifically tuned for each target task ("DNM (Task-Specific Best)").

Results The results, summarized in Table 18, demonstrate the remarkable effectiveness of the transferred topology. On EMNIST and CIFAR10, the DNM (Transferred) model significantly outperforms the random, BSW, BRF, and Ramanujan initializations. On Fashion MNIST, however, BSW slightly outperforms it. Its performance is only marginally lower than that of the task-specific DNM.

Table 18: Performance of the transferred DNM topology on static sparse training at 99% sparsity. The "DNM (Transferred)" model uses the single best hyperparameter configuration found on MNIST for all three target tasks. Its performance is compared to baselines and to the best task-specific DNM configuration. Scores are the accuracy averaged over 3 seeds ± standard error. Bold values denote the best performance.

	Static Sparse Training		
	Fashion MNIST	**EMNIST**	**CIFAR10**
FC	90.88±0.02	87.13±0.04	62.85±0.16
CSTI	88.52±0.14	84.66±0.13	52.64±0.30
SNIP	87.85±0.22	84.08±0.08	61.81±0.58
Random	87.34±0.11	82.66±0.08	55.28±0.09
BSW	88.18±0.18	82.94±0.06	56.54±0.15
BRF	87.41±0.13	82.98±0.02	54.73±0.07
Ramanujan	86.45±0.15	81.80±0.13	55.05±0.40
DNM (Fine-tuned)	**89.19±0.01**	**84.76±0.13**	**61.63±0.18**
DNM (Transferred)	87.98±0.06	83.67±0.06	58.70±0.42

Discussion This experiment strongly suggests that the structural principles identified by DNM as optimal for MNIST serve as a powerful and generalizable prior for other image classification tasks. The ability to transfer a high-performing topology with minimal performance loss has significant practical implications, as it can drastically reduce the computational cost associated with architecture search for new applications. This finding reinforces the idea that bio-inspired, structured initialization is not merely a task-specific trick but a robust strategy for building efficient sparse networks.

Appendix K. Reproduction statement

All experiments were conducted on NVIDIA A100 80GB GPUs. MLP and Transformer models were trained using a single GPU. The code to reproduce the experiments will be made publicly available upon publication.

Appendix L. Claim of LLM Usage

The authors declare that Large Language Models (LLMs) were used in the writing process of this manuscript. However, the core idea and principles of the article are entirely original and were not generated by LLMs.

Appendix M. Limitations and Future Challenges

This study may be limited by the lack of widespread adoption of the hardware needed to speed up sparse training with unstructured sparsity. Nonetheless, several companies are releasing devices that support unstructured sparsity in training (Thangarasa et al., 2023;

Kurtz et al., 2020). For future work, we aim to develop methods for building a physics-informed DNM model that can automatically detect the best parameter setting based on the input information. This would allow us to extend the rationale of CSTI to Transformers and LLMs.