УДК 004.032.26
DOI: 10.18523/2617-3808.2025.8.4-14

S. Medvid

MORPHONAS-BENCH: A BENCHMARK SUITE FOR MORPHOGENETIC NEURAL NETWORK GENERATION

We present MorphoNAS-Bench, a benchmark and toolkit for neural architecture search (NAS) using a generative, developmentally inspired design space. Unlike current NAS benchmark datasets (NAS-Bench-101, NATS-Bench) that use static graph encodings of networks, in MorphoNAS-Bench networks are simple, compact genomes that drive morphogenetic development, allowing for a variety of richly defined, spatially embedded recurrent architectures that emerge through different forms of deterministic growth. The following local developmental rules are used in MorphoNAS to grow genomes: morphogen diffusion, cell division, differentiation, and axon guidance as key mechanisms. The seed benchmark dataset presented in this work consists of 1,000 genome-architecture pairs, taken from a pool of over 50,000 generation attempts using the following quality thresholds: a minimum 5 neurons, 3 edges, and 70% out-degree coverage. The dataset was constructed using Latin Hypercube Sampling (LHS) with orthogonal array design to ensure comprehensive parameter space coverage. The attempts were conducted using both fully stratified parameter sampling and a biologically inspired Genome.random() sampling method, ensuring a reasonable level of coverage of the search space while being plausible. Each sample includes detailed annotations of graph entropy, hierarchy scores, core-periphery structure, transitivity, reciprocity, and structural balance metrics. We share an analysis of the emergent properties like size, modularity, grouping, and efficiency, demonstrating that both generation strategies can produce structured networks that are rich in their nontriviality. The provided Python toolkit provides the means of investigation to test how genomes develop into neural networks, with associated structural analysis, framing MorphoNAS-Bench as a reproducible and biologically inspired testbed for any research studies exploring architecture diversity, evolution, and emergent structure in NAS.

Keywords: neural networks, developmental encoding, morphogenetic development, neural architecture search, benchmark toolkit, emergent modularity, indirect encoding.

1. Introduction

Neural Architecture Search (NAS) attempts to automate the design of neural networks. Typically, NAS has searched a fixed set of architectures that are normally explicitly encoded as graphs or modules. Benchmarks like NAS-Bench-101 and NATS-Bench have been critical for reproducible research, but still use a top-down framework that constrains variability and is completely different from the biological processes generating real neural systems.

MorphoNAS-Bench is a benchmark that addresses this gap by being grounded in a generative search space. MorphoNAS-Bench does not directly encode networks, but instead encodes a compact genome for networks that develop into a neural architecture through simulated morphogenetic development, inspired by biological embryogenesis. Each morphogenetic genome specifies local morphogen diffusion, cell division, differentiation, and axon guidance, and as a result a diverse population of spatially embedded, recurrent neural networks arise through deterministic simulation.

© *S. Medvid*, 2025

This paper will present MorphoNAS-Bench as a dataset and toolkit, including a stratified sample of 1,000 seed architecture genomes generated for several significant developmental parameters, including fully stratified sampling and a biologically plausible Genome.random() method. Each network genome generates a spatially embedded neural network with recorded metrics based on node number, degree distributions, clustering, and spatial organization. Also, we include Python scripts and utility programs for the creation, development, and evaluation of genome neural networks.

Our project fosters reproducible experimentation with developmental NAS methods by introducing additional populations of neural architecture designs that are biologically grounded for additional algorithmic exploration. MorphoNAS-Bench can provide the context to evaluate emergent properties of the resulting networks, as well as design NAS algorithms for a space where architectures can develop, as opposed to the creation of neural networks using a space defined by a human designer.

We built the benchmark on the theoretical framework we described in more detail previously in [6]. It outlined the theoretical intent and generative approach based on morphogenetic growth, inspired by the Free Energy Principle. While that paper primarily deals with the foundational aspects of MorphoNAS-Bench, this work provides a benchmark to investigate structural diversity and architecture design potential of genomically grown networks.

2. Related Work

2.1. Neural Architecture Search and Benchmarks

Neural Architecture Search (NAS) has become an important framework for automating the design of neural networks by exploring large search spaces using optimization techniques such as reinforcement learning, evolutionary strategies, and differentiable approaches. One notable advance has been the manual development of the NAS benchmarks that facilitate reproducible and fair comparisons among algorithms.

NAS-Bench-101 [15] was the first detailed tabular benchmark which consists of over 423,000 cell-based architectures and is evaluated on CIFAR-10. Each architecture was trained using a standardized protocol, with a record of performance metric values in a lookup table that allowed comparative performance checks with little latency. Following that study, **NAS-Bench-201** [4] developed a smaller and more controlled search space and observed the performance across CIFAR-10, CIFAR-100, and ImageNet-16. This expanded and allowed for generalization analysis and also provided an increase in analytical efficiency for testing NAS methods.

NATS-Bench [3] added on to the paradigm by also considering macro and micro architectures, and allowing training-free evaluations and weight-sharing methods. These benchmarks have allowed for a shift in the community towards more rigorous, transparent, and standardized evaluation of NAS architectures.

Other approaches have pioneered the use of differentiable NAS, such as **DARTS** [8], where architecture weights are optimized with model parameters, providing a fast one-shot training method. While these approaches represent advances, they typically only consider very constrained, manually encoded architecture spaces.

One common attribute in current NAS benchmark studies is that they all rely on **explicitly encoded architectures** as directed acyclic graphs (DAGs) or operation lists. As such, the graphs are generic, static, and result in non-generative architecture spaces, which limit researchers' exploration of open-ended or biologically inspired architecture spaces.

2.2. Developmental and Generative Approaches to Architecture Design

There is increasing interest in the composition of **developmental** and **indirectly encoded** neural systems outside of the classical NAS paradigm, where the architecture arises from a compact generative set of rules rather than a directly specified list of components.

There are experimental examples of this emerging approach beginning with NEAT [13], in which both topologies and weights are evolved from direct mutations and crossover of graph-type structures, and **HyperNEAT** [12], which used **Compositional Pattern Producing Networks (CPPNs)** to indirectly encode relationships between elements in a connectivity pattern based on geometric distances. This approach demonstrated the potential of indirect encodings in terms of potentially producing binary representations for traditional regular scalable networks.

Generative systems have started emerging only recently, in areas such as neural tissue simulation [10], modular robot morphogenesis [14], and procedural graph generation [2].

Most of such frameworks still remain largely domain specific, and there does not yet exist a more generalized evaluation platform that permits systematic algorithmic comparisons across tasks and domains. There are some benchmarks (e.g. PCG Benchmark [7], Evolution Gym [1]) that give a little bit of structure within a single domain, but there is still no broadly applicable platform that allows for head-to-head evaluation of generative algorithms across datasets that include neural simulation, robot morphology, and graph-based generation tasks [5].

2.3. Positioning of MorphoNAS-Bench

MorphoNAS-Bench is a step forward with a benchmark for neural architectures designed by developmentally generating them. It combines a genome encoding inspired by biology that defines morphogen dynamics and axonal growth, a deterministic simulator that generates spatially embedded recurrent graphs, and a curated database of valid genomes and their structural metadata. Unlike previous NAS benchmarks [3, 4, 15], MorphoNAS-Bench provides a model for a generative architecture space where topology and function emerge through simple local interaction rules and can be employed to explore evolvability/compactness/biological realism, which none of the existing tabular datasets can do. MorphoNAS-Bench can contribute to the neuroevolution, NAS, and developmental computation communities by providing a vehicle for looking at architectures that grow organically, as opposed to just by refinement. It is an additional, open-ended avenue for architecture search.

3. MorphoNAS-Bench Overview

This section provides descriptions of the components of the MorphoNAS-Bench search space, the operational genome encoding format, and the resulting networks' properties.

3.1. Genome Encoding

The genome representation in MorphoNAS-Bench defines morphogen diffusion, cell fate thresholds, and axon guidance rules that drive network development. Broadly, these mechanisms follow the generative model in a separate theoretical paper [6]. The specification of the genome structure, parameter definitions, and configuration options are available on the MorphoNAS-Bench GitHub repository https://github.com/sergemedvid/MorphoNAS-Bench.

3.2. Morphogenetic Development Process

The network growth is implemented using a deterministic simulation of developmental dynamics based on the genome encoding provided above. A detailed structure and model of these dynamics, including biological motivation, is provided in the supplementary theoretical paper [6].

3.3. Comparison with Traditional NAS Spaces

MorphoNAS-Bench introduces a **complementary design space** for NAS, which enables research of the evolutionary and generative methods that are often incompatible with standard benchmarks.

Table 1. **Comparison of MorphoNAS-Bench with traditional NAS spaces**

Property	MorphoNAS-Bench	Traditional NAS Benchmarks
Encoding Type	Indirect (genome → process)	Direct (graph / op list)
Topology Generation	Emergent via simulation	Static, predefined graph space
Network Size	Variable	Fixed or bounded
Biological Plausibility	High	Low
Interpretability	High (local rules → global form)	Medium
Reproducibility	Deterministic	Deterministic
Search Space Diversity	Very high	Limited by architecture schema

4. Benchmark Dataset and Evaluation Tasks

MorphoNAS-Bench includes a curated, developmentally-grounded benchmark dataset designed to characterize and uncover the search space of architectures generated from within the MorphoNAS framework. The benchmark is simply a starting point for search, discovery, and exploration of diversity within a biologically-inspired generative space, rather than for comparison of fitness value against predefined graph targets. This section describes the dataset construction, evaluation tasks, and supporting metrics to study emergent neural architectures.

4.1. Dataset Construction

The construction of the MorphoNAS-Bench dataset was accomplished via large numbers of simulated genomes, using stratified sampling [11], as well as filtering and post hoc analysis after simulated growth was complete. All genomes simulated through the morphogenetic growth process, generate a directed, weighted recurrent neural network, and, with it the structural and functional properties can then be analyzed.

4.2. Genome Generation

Genome sampling involved a combination of Latin Hypercube Sampling (LHS) methodology [9] and orthogonal array design to maximize coverage and statistical quality of core parameters. Two sampling strategies were used. The first was a fully stratified sampling of all queried parameter ranges, and the second, a biologically-informed Genome.random(), are available at GitHub https://github.com/sergemedvid/MorphoNAS-Bench. The stratified sampling maximizes space coverage by sampling parameters independently of each other, which could allow for some impossible combinations, while the biologically-informed sampling also stratifies core traits, while the additional fields are filled using domain-informed constraints (i.e., normalized diffusion and inhibition matrices, morphogen probabilities) to ensure biologically plausible combinations. Each Genome JSON is fully traceable in the generation_metadata.json file, where all parameters, random seed, CLI flags, and code version used to generate that genome are recorded.

4.3. In-Loop Quality Filtering

Real-time filtering occurs after the growth process is finished: only networks with a minimum of five neurons, at least three edges, and no less than 70% of neurons with an outgoing connection, are included in the dataset. We have additional thresholds on weak edge connectivity and density to eliminate trivial and degenerate topologies to ensure only structurally meaningful architectures enter the dataset. The pipeline will also keep track of the successful and failed generations to corroborate further analysis of the generative space within the framework.

By default, we use LHS over the parameter space with random morphogen-to-rule mappings (i.e., Genome.random()) set, while if the --no-genome-random flag is used, the mappings become deterministic with sampling from the rule table.

5. Toolkit and Implementation Resources

The MorphoNAS-Bench toolkit and resources, including genome generation scripts, morphogenetic development engine, visualization tools, and evaluation pipelines, is openly accessible at https://github.com/sergemedvid/MorphoNAS-Bench. Within the MorphoNAS-Bench codebase, the README contains important documentation, configuration examples, and Jupyter-compatible workflows for reproducing results and expanding the benchmark suite. We encourage users to explore the README and API documentation if they would like to integrate the MorphoNAS-Bench codebase into their own NAS pipelines or other developmental modeling experiments.

6. Baseline Results and Use Cases

To explore the structure, diversity, and functional viability of MorphoNAS-Bench, we completed baseline evaluations. This section outlines examples that highlight the diversity of the generative space and demonstrates the structural expressiveness of the underlying model, as well as provides summary use cases for researchers that work in the area of neural architecture search (NAS) and/or generative graph modeling.

6.1. Generation Metadata and Sample Efficiency

While both generation methods used the same post-growth filtering, requiring at least five neurons, weak connectivity, and 70% out-degree coverage, there are significant differences in sample efficiency for the two generation strategies. In the fully stratified method, for the generation of 1000 filtered genomes, there were 46822 rejected based on insufficient node count and 2610 rejected based on disconnectedness. In the Genome.random() method, for the generation of 1000 filtered genomes, there were 23817 rejected based on insufficient node count and another 2626 rejected based on disconnectedness. This indicates that the Genome.random() method resulted in a higher yield of valid architectures, and generated overall fewer genomes to arrive at each individual valid architecture.

This comparison shows the benefits of biologically inspired genome construction; it preserves diversity and expressiveness in the search space while being able to generate functionally plausible and structurally correct networks more quickly. The results, in the end, confirm both the generative richness of MorphoNAS, and the need for filtering to arrive at viable meaningful neural architectures.

6.2. Structural Metrics and Emergent Patterns

We conducted an extensive structural characterization of **1000 networks** sampled from the benchmark. The structural analysis shows the emergence of a range of topological properties and is shown in **Figures 1 and 2** (*advanced network metrics analyses*).

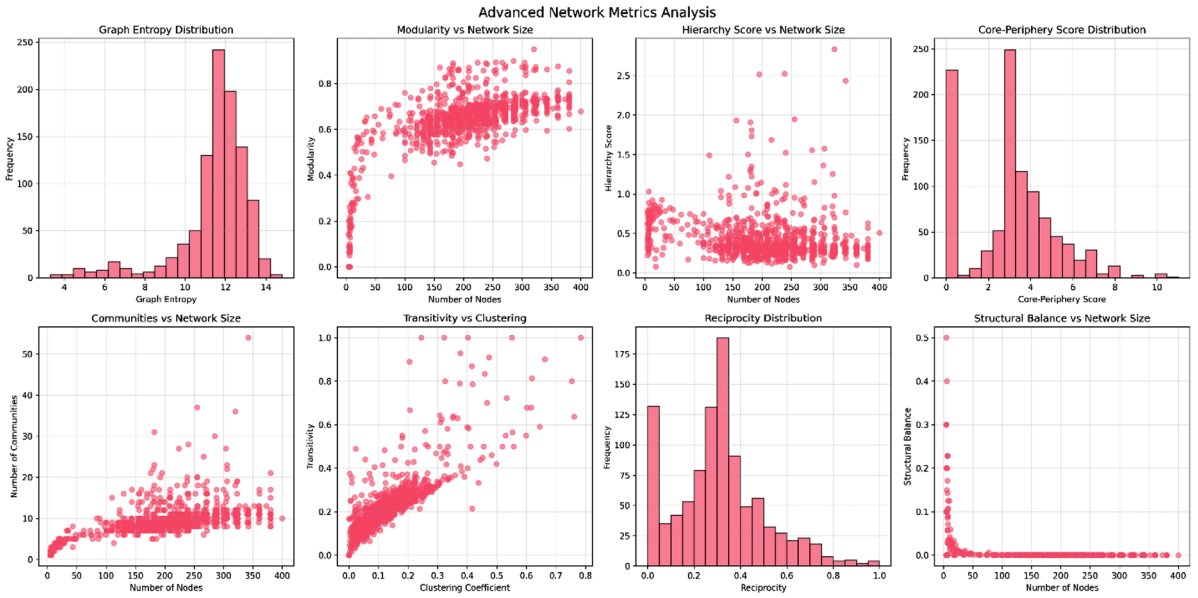

Figure 1. Advanced network metrics analysis for networks generated with Genome.random()

Both generation strategies produce networks that range in size and types of topology, although subtle structural distributional differences are indicated in Table 1.

Table 2. **Comparison of advanced network metrics for Genome.random() and fully stratified sampling (1,000 Samples)**

Metric / Pattern	Genome.random (biologically plausible)	Fully Stratified
Graph Entropy	More peaked, shifted higher (more complex)	Broader, more low-entropy outliers
Modularity (vs. Size)	Higher modularity, esp. at large sizes	More moderate modularity at large size
Hierarchy Score	More high-hierarchy outliers, esp. large	Mostly low-moderate, fewer outliers
Core-Periphery Score	More uniformly low-moderate	Pronounced secondary mid-range peak
Communities (vs. Size)	Tighter, higher at large sizes	More spread, fewer at large sizes
Transitivity vs. Clustering	More high-transitivity outliers	Tighter, lower overall
Reciprocity Distribution	Narrower, centered at mid-range	Broader, more low/high outliers
Structural Balance (vs. Size)	Mostly low, very few outliers	More outliers at small sizes

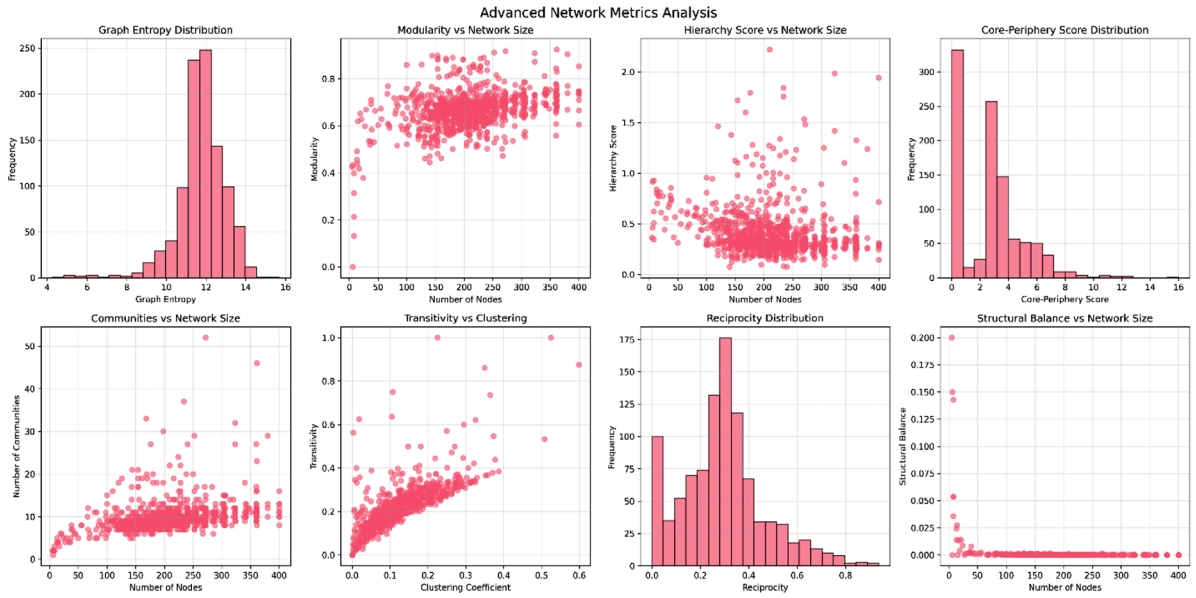

Figure 2. Advanced network metrics analysis for networks generated with the fully stratified method

6.3. Parameter Influence and Developmental Constraints

In order to understand how genome parameters influence the final architectures, we correlated the genome settings with network-level outcomes. **Figures 3 and 4** (genome parameters vs. network properties) depict these relationships.

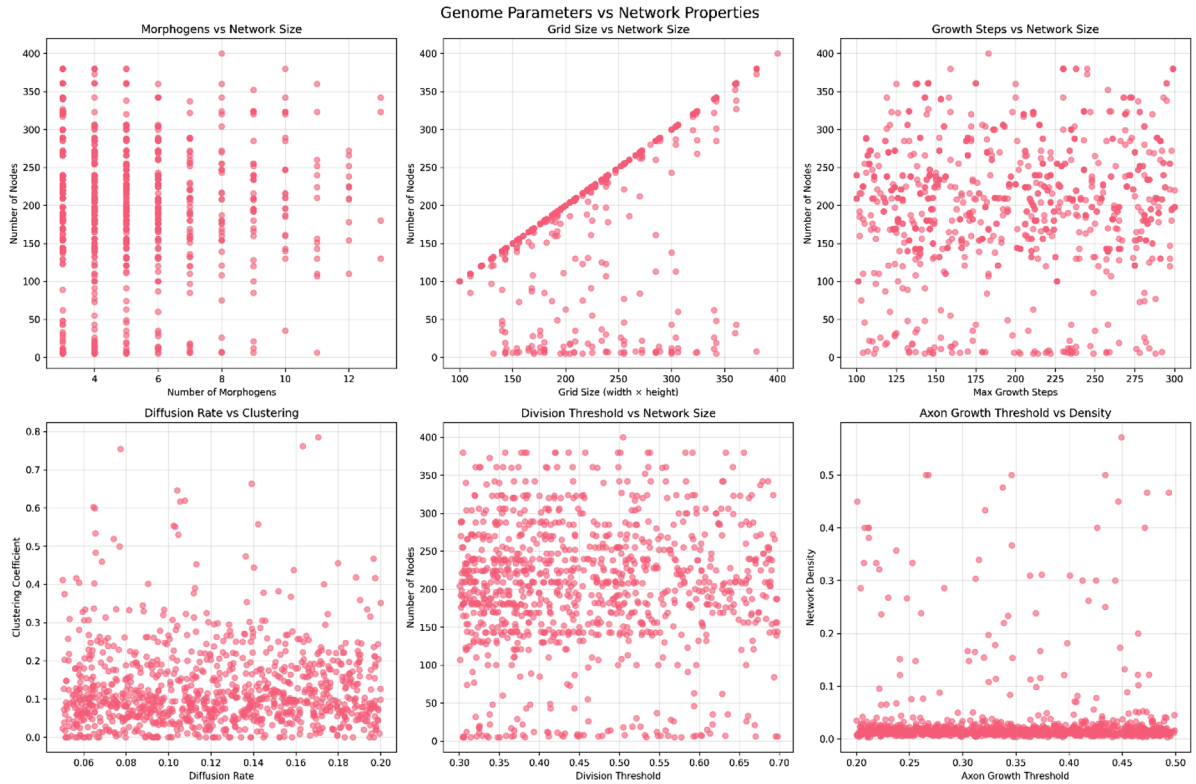

Figure 3. Genome parameters vs. network properties for genomes generated with Genome.random() method

Both generation strategies offer a broad exploration of how genome parameters shape network properties, although subtle differences in their variability and outlier behavior distinguish the two distinct methods, and are summarized further in Table 2.

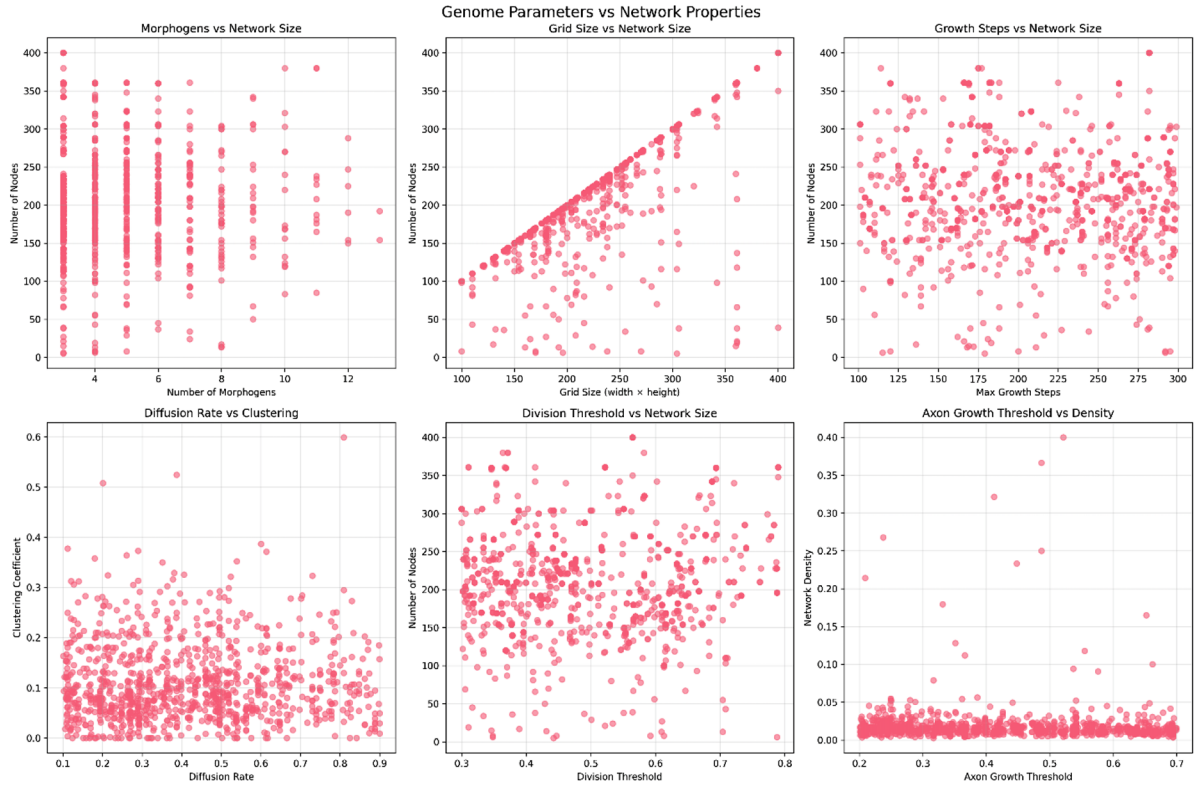

Figure 4. Genome parameters vs. network properties for genomes generated with the fully stratified method

Table 3. **Comparison of genome parameter effects on network properties for Genome.random() and fully stratified sampling (1,000 samples)**

Metric / Relationship	Genome.random (biologically plausible)	Fully Stratified
Morphogens vs. Network Size	Broad spread of network sizes at each morphogen count; more vertical variability across range	Similar broad spread, with slightly more high-size outliers at some counts
Grid Size vs. Network Size	Strong positive correlation, some dispersion; a few outlier networks above/below main diagonal	Very tight correlation, minimal dispersion; nearly all networks tightly follow grid area
Growth Steps vs. Network Size	Little direct correlation; wide vertical spread at all values	Same: step count does not predict size, wide spread across the range
Diffusion Rate vs. Clustering	Mostly low clustering; a few higher outliers up to 0.6	Mostly low clustering, but a few very high outliers (up to 0.8); wider spread at low diffusion rates
Division Threshold vs. Network Size	No strong trend; wide range of sizes at each threshold	No trend, but slightly broader vertical spread in network sizes at all thresholds
Axon Growth Threshold vs. Density	Density concentrated at low values, with a few moderate outliers; no strong relationship	Also concentrated at low density, but outliers reach higher density (up to ~0.5), especially at lower thresholds

Both methods provide a broad exploration of the parameter-to-architecture mapping of networks, although the **fully stratified sampling** indicates even more extreme outliers for clustering and density when compared to **Genome.random**, which produces a marginally more regular and reasonable distribution. Network size tended to dominate any reasonable number of genome parameters that modulated a range of structural properties, with considerable variability indicating the richness and complexity of the proposed generative developmental process.

6.4. Benchmark-Wide Structural Profiles

Figures 5 and 6 (*network structural analysis*) summarize general distributional properties across the 1,000-network benchmark.

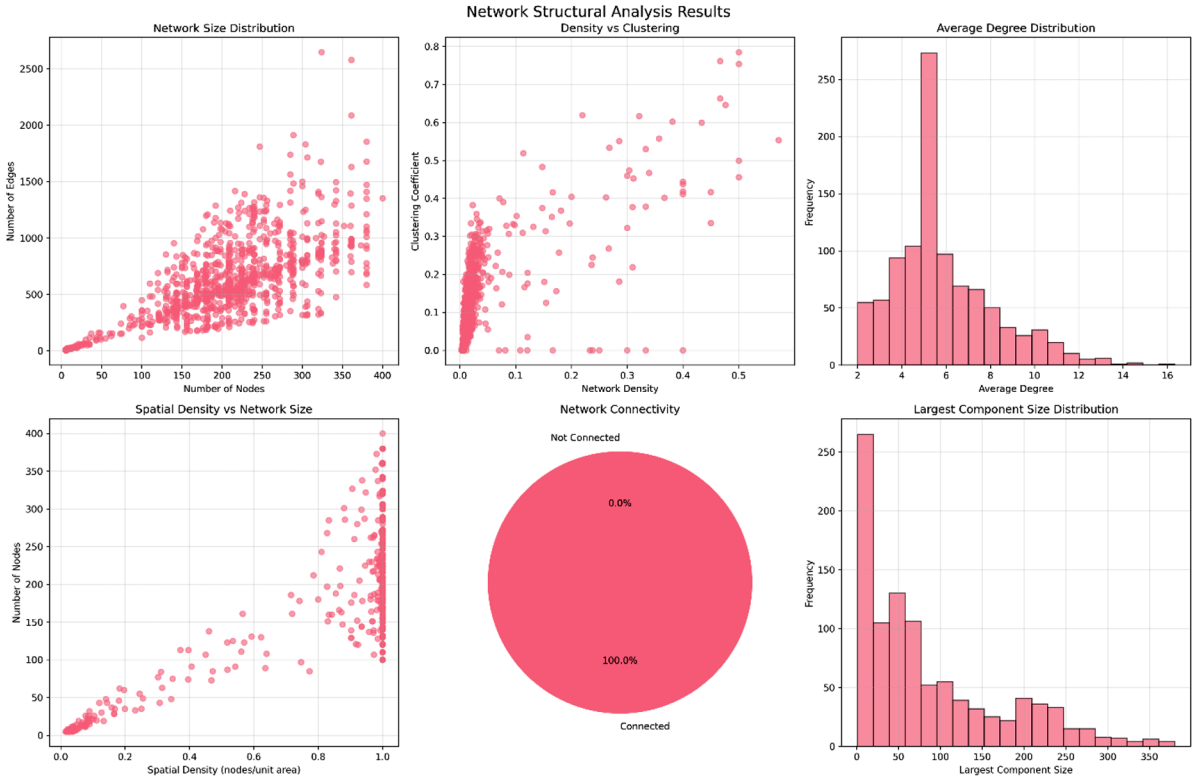

Figure 5. Network structural analysis results for genomes generated with Genome.random() method

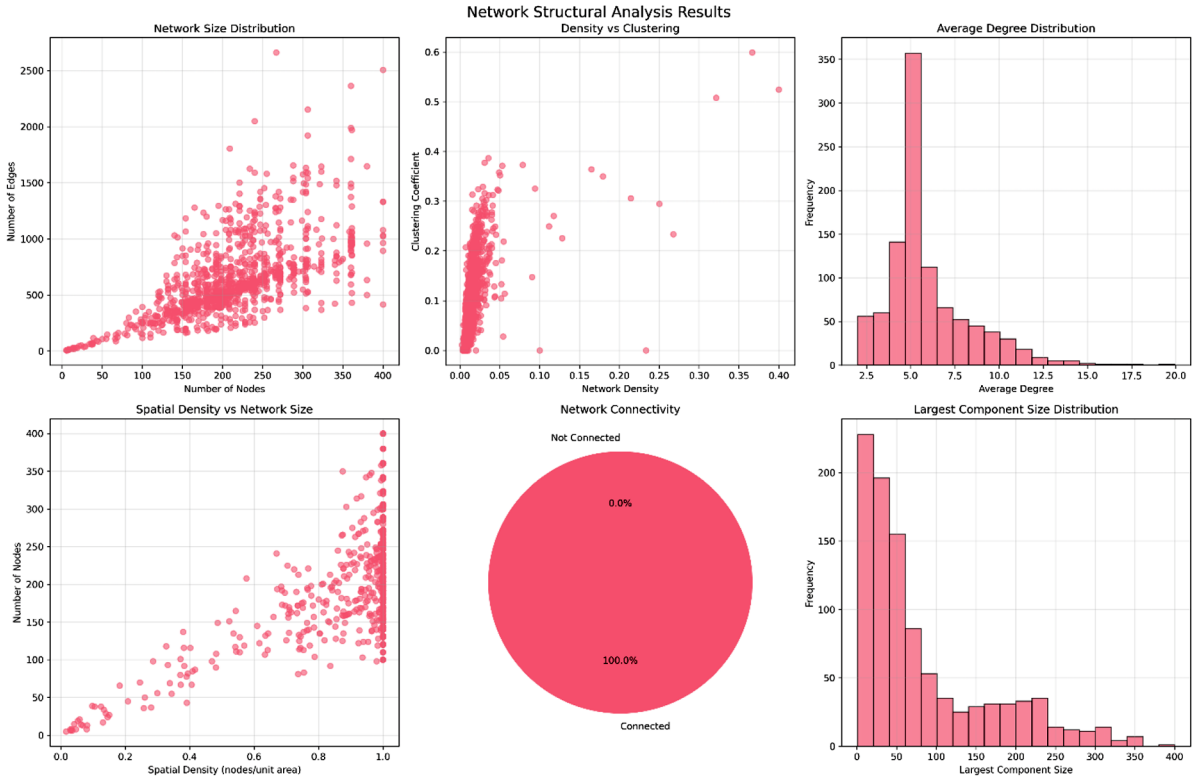

Figure 6. Network structural analysis results for genomes generated with the fully stratified method

In practical terms, **the Genome.random method presents optimal NAS research for applicable conditions**, as the search space is diverse and populated with functional, meaningful architectures requiring little filtering or alteration following generation.

In contrast, those wishing to assess structural diversity without bounds, or to benchmark NAS algorithms under the most extreme and unpredictable conditions, would likely **benefit from completely stratified sampling** since this allows exposure through the search process to a range of network topologies that included even the edge cases. For the purposes of establishing a recommendation, we suggest the Genome. random method as the primary approach for experimental evaluation, and use fully stratified sampling as a supplementary technique for assessing diversity and conducting ablation analyses.

6.5. Summary

This section has indicated that MorphoNAS-Bench is structurally rich, viable for assessing NAS search space, quantitatively diverse across genome and network features, and maintains transparent filtering and reproducibility. By bringing together developmental encoding, statistical sampling, and structural annotation, MorphoNAS-Bench provides a generative, biologically-based search space that can be investigated, analyzed, and benchmarked, filling an identifiable knowledge gap in NAS research.

7. Conclusion and Future Work

MorphoNAS-Bench presents a new methodology for neural architecture search benchmarks. In this regard, we have transitioned from static graph encodings to networks that are developed and shaped through a biologically inspired growth process. This enables us to produce a tight, generative, reproducible search space that is rich in structural and functional variation. Through careful parameter sampling, quality filtering, and analysis, we have shown that our benchmark samples form a larger, controllable architecture space, with sampled networks that are structurally complex and functionally able. We discovered that basic NAS and evolutionary algorithms are able to explore the architecture space without difficulties, establishing its accessibility, yet also its challenging nature. Our methodology differs from previous benchmarks, by not restricting the search space format to a set of cells or sequences of operations. The design of MorphoNAS-Bench opens a pathway for researchers to examine, among others, indirect encodings, morphogenetic processes, developmental constraints, evolvability, modularity, and variation in a biologically plausible manner, as well as apply metrics for evaluating search strategies on the basis of performance, coverage, structural novelty, and generative robustness.

Future Work

Future work is possible towards several promising opportunities to extend MorphoNAS-Bench. **Extending functional tasks** will be accomplished by adding new, reiterative reinforcement learning environments, supervised learning challenges, and evaluations of transfer or multi-task generalization. **Integrating a search strategy** is about developing benchmarks and competitions for differentiable, gradient-free and/or meta-learning NAS methods, with baseline comparisons of sample efficiency, diversity, and convergence. **Genotype-phenotype mapping** may involve studies related to redundancy, robustness, locality, and the neutral networks and mutational neighborhoods structure. We plan to **expand the benchmark** to become increasingly large with tiered datasets consisting of over 100,000 genomes, creating intentional parameters that allow only configurations for low-resource, diversity-based, or task-based evaluations. Finally, there is a rich opportunity for additional **tooling and visualization** within the structure of MorphoNAS-Bench; for example, bringing web-based viewers (that showcase a demographic of developmental simulations and network exploration) and surrogate predictors trained from the neural architecture benchmark data to support the genome-based NAS.

Список літератури

1. Bhatia J. Evolution gym: A large-scale benchmark for evolving soft robots / J. Bhatia, H. Jackson, Y. Tian, et al. // Advances in Neural Information Processing Systems. — 2021. — Vol. 34. — Pp. 2201–2214.
2. Deitke M. ProcTHOR: Large-Scale Embodied AI Using Procedural Generation / M. Deitke, E. VanderBilt, A. Herrasti, et al. // Advances in Neural Information Processing Systems. — 2022. — Vol. 35. — Pp. 5982–5994.
3. Dong X. Nats-bench: Benchmarking nas algorithms for architecture topology and size / X. Dong, L. Liu, K. Musial, B. Gabrys // IEEE transactions on pattern analysis and machine intelligence. — 2021. — Vol. 44, no.7. — Pp. 3634–3646.
4. Dong X. Nas-bench-201: Extending the scope of reproducible neural architecture search / X. Dong, Y. Yang // arXiv preprint arXiv:2001.00326. — 2020.

5. Geng H. RoboVerse: Towards a unified platform, dataset and benchmark for scalable and generalizable robot learning / H. Geng, F. Wang, S. Wei, et al. // arXiv preprint arXiv:2504.18904. — 2025.
6. Glybovets M. MorphoNAS: Embryogenic Neural Architecture Search Through Morphogen-Guided Development / M. Glybovets, S. Medvid // arXiv preprint arXiv:2507.13785. — 2025. — https://arxiv.org/abs/2507.13785.
7. Khalifa A. The Procedural Content Generation Benchmark: An Open-source Testbed for Generative Challenges in Games / A. Khalifa, R. Gallotta, M. Barthet, et al. — 2025.
8. Liu H. Darts: Differentiable architecture search / H. Liu, K. Simonyan, Y. Yang // arXiv preprint arXiv:1806.09055. — 2018.
9. McKay M. Comparison of Three Methods for Selecting Values of Input Variables in the Analysis of Output from a Computer Code / M. McKay, R. Beckman, W. Conover // Technometrics. — 1979. — Vol. 21, no. 2. — Pp. 239–245.
10. Migliore M. Parallel network simulations with NEURON / M. Migliore, C. Cannia, W. W. Lytton, et al. // Journal of computational neuroscience. — 2006. — Vol. 21, no. 2. — Pp. 119–129.
11. Neyman J. On the Two Different Aspects of the Representative Method: The Method of Stratified Sampling and the Method of Purposive Selection / J. Neyman // Journal of the Royal Statistical Society. — 1934. — Vol. 97, no. 4. — Pp. 558–625.
12. Stanley K. O. A Hypercube-Based Encoding for Evolving Large-Scale Neural Networks / K. O. Stanley, D. B. D'Ambrosio, J. Gauci // Artificial Life. — 2009. — https://doi.org/10.1162/ARTL.2009.15.2.15202.
13. Stanley K. O. Evolving Neural Networks through Augmenting Topologies / K. O. Stanley, R. Miikkulainen // Evolutionary Computation. — 2002. — https://doi.org/10.1162/106365602320169811.
14. Veenstra F. Evolution and Morphogenesis of Simulated Modular Robots: A Comparison Between a Direct and Generative Encoding / F. Veenstra, A. Faina, S. Risi, K. Stoy ; Squillero G., Sim K. — Cham : Springer International Publishing, 2017. — ISBN 978-3-319-55849-3.
15. Ying C. Nas-bench-101: Towards reproducible neural architecture search / C. Ying, A. Klein, E. Christiansen, et al. — 2019.

References

Bhatia, J., Jackson, H., Tian, Y., Xu, J., & Matusik, W. (2021). Evolution gym: A large-scale benchmark for evolving soft robots. *Advances in Neural Information Processing Systems*, *34*, 2201–2214.

Deitke, M., VanderBilt, E., Herrasti, A., Weihs, L., Ehsani, K., Salvador, J., Han, W., Kolve, E., Kembhavi, A., & Mottaghi, R. (2022). ProcTHOR: Large-Scale Embodied AI Using Procedural Generation. *Advances in Neural Information Processing Systems*, *35*, 5982–5994.

Dong, X., Liu, L., Musial, K., & Gabrys, B. (2021). Nats-bench: Benchmarking nas algorithms for architecture topology and size. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *44* (7), 3634–3646.

Dong, X., & Yang, Y. (2020). Nas-bench-201: Extending the scope of reproducible neural architecture search. *arXiv Preprint arXiv:2001.00326*.

Geng, H., Wang, F., Wei, S., Li, Y., Wang, B., An, B., Cheng, C. T., Lou, H., Li, P., Wang, Y.-J., & others. (2025). RoboVerse: Towards a unified platform, dataset and benchmark for scalable and generalizable robot learning. *arXiv Preprint arXiv:2504.18904*.

Glybovets, M., & Medvid, S. (2025). MorphoNAS: Embryogenic Neural Architecture Search Through Morphogen-Guided Development. *arXiv Preprint arXiv:2507.13785*. https://arxiv.org/abs/2507.13785

Khalifa, A., Gallotta, R., Barthet, M., Liapis, A., Togelius, J., & Yannakakis, G. N. (2025). The Procedural Content Generation Benchmark: An Open-source Testbed for Generative Challenges in Games. *Proceedings of the 20th International Conference on the Foundations of Digital Games*, 1–12.

Liu, H., Simonyan, K., & Yang, Y. (2018). Darts: Differentiable architecture search. *arXiv Preprint arXiv:1806.09055*.

McKay, M., Beckman, R., & Conover, W. (1979). Comparison of Three Methods for Selecting Values of Input Variables in the Analysis of Output from a Computer Code. *Technometrics*, *21* (2), 239–245.

Migliore, M., Cannia, C., Lytton, W. W., Markram, H., & Hines, M. L. (2006). Parallel network simulations with NEURON. *Journal of Computational Neuroscience*, *21* (2), 119–129.

Neyman, J. (1934). On the Two Different Aspects of the Representative Method: The Method of Stratified Sampling and the Method of Purposive Selection. *Journal of the Royal Statistical Society*, *97* (4), 558–625.

Stanley, K. O., D'Ambrosio, D. B., & Gauci, J. (2009). A Hypercube-Based Encoding for Evolving Large-Scale Neural Networks. *Artificial Life*. https://doi.org/10.1162/ARTL.2009.15.2.15202.

Stanley, K. O., & Miikkulainen, R. (2002). Evolving Neural Networks through Augmenting Topologies. *Evolutionary Computation*. https://doi.org/10.1162/106365602320169811.

Veenstra, F., Faina, A., Risi, S., & Stoy, K. (2017). Evolution and Morphogenesis of Simulated Modular Robots: A Comparison Between a Direct and Generative Encoding. In G. Squillero & K. Sim (Eds.), *Applications of Evolutionary Computation* (pp. 870–885). Springer International Publishing. https://doi.org/10.1007/978-3-319-55849-3_56.

Ying, C., Klein, A., Christiansen, E., Real, E., Murphy, K., & Hutter, F. (2019). *Nas-bench-101: Towards reproducible neural architecture search*, 7105–7114. https://proceedings.mlr.press/v97/ying19a.html.

Медвідь С. О.

MORPHONAS-BENCH: БЕНЧМАРК ДЛЯ МОРФОГЕНЕТИЧНОЇ ГЕНЕРАЦІЇ НЕЙРОННИХ МЕРЕЖ

У роботі представлено MorphoNAS-Bench — бенчмарк і набір інструментів для пошуку нейронних архітектур (Neural Architecture Search, NAS), який оснований на генеративному, розвитково-натхненому просторі пошуку. На відміну від сучасних NAS-бенчмарків, які використовують статичні кодування графів, що представляють нейронні мережі, MorphoNAS-Bench

характеризується компактними геномами. Ці геноми контролюють процес морфогенетичного розвитку, що дозволяє створювати різні просторові рекурентні архітектури, які виникають внаслідок різних типів детермінованого зростання, які при цьому визначаються локальними правилами розвитку.

Початковий набір даних бенчмарку містить 1000 пар «геном-архітектура», які були обрані з більш ніж 50 000 спроб генерації. Цей набір був створений шляхом використання як повністю стратифікованого відбору параметрів, так і за біологічно-натхненним методом Genome.random(). Застосування випадкового підходу забезпечує адекватне охоплення площі пошуку та реалістичність результатів. Кожне знайдене рішення містить детальну анотацію структурних показників. Ми проводимо аналіз таких структурних характеристик, як розмір, модульність, групування та ефективність. Ми показуємо, що обидві стратегії генерації здатні утворювати як структуровані, так і нетривіальні мережі.

Наданий інструментарій на Python дозволяє вивчати процеси розвитку геномів нейронних мереж разом із відповідним структурним аналізом. Таким чином, MorphoNAS-Bench виступає як повторювана і біологічно обґрунтована платформа для досліджень різноманітності, еволюціонування, та емерджентні структури для NAS.

Ключові слова: нейронні мережі, розвиткове кодування, морфогенетичний розвиток, пошук нейронних архітектур, бенчмарк-інструментарій, емерджентна модульність, непряме кодування.

Матеріал надійшов 21.07.2025

Creative Commons Attribution 4.0 International License (CC BY 4.0)