ChatGPT

# Related Work on Constructive and Dynamic Neural Architectures

Modern neural networks can **grow new units and connections during learning**, a paradigm with roots in the 1980s–90s. One early example is the **Cascade-Correlation** algorithm [1] [2], which begins with a minimal network (only inputs and outputs) and *automatically adds hidden neurons one by one* during training [1]. Each new neuron is trained locally (to maximize its correlation with the residual error) and then frozen, becoming a permanent feature detector [3]. Notably, Cascade-Correlation *does not use backpropagation through the entire network* – each added unit is trained by a local objective and thereafter the existing weights are not altered [4]. Growth in Cascade-Correlation is **deterministic** given the training data order and stops when performance plateaus, so network size is *adaptively determined* rather than fixed *a priori*. However, its architecture forms a shallow "cascade" rather than a deep layered hierarchy (new neurons receive inputs from all previously added units, creating one large hidden layer in effect). It does not impose a multi-dimensional grid or spatial structure on connections, and there is no notion of **2D spatial wiring** or unique source constraints – each new neuron simply connects to all existing units. Variants like the **Upstart algorithm** (Frean 1990) and other *dynamic node creation* methods introduced similar one-by-one growth of hidden units using different criteria (e.g. error-based triggers) [5].

In parallel, the unsupervised learning community developed **self-organizing networks** capable of structural growth. **Kohonen's Self-Organizing Map (SOM)** organizes neurons on a fixed two-dimensional lattice, but extensions such as the **Growing Grid** and **Growing Neural Gas (GNG)** introduced the ability to add neurons over time. **GNG** (Fritzke 1995) begins with a small set of nodes and **incrementally inserts new nodes** to better cover the input space topology [6] [7]. GNG adds a new unit after a fixed number of input signals: it finds the neuron with the largest accumulated quantization error and inserts a new neuron in its vicinity, connecting it to the network [6]. This growth is **triggered by local error measures** and is repeated until a stopping criterion (e.g. desired quantization error or max units) is reached [8]. The process is largely deterministic (given a fixed input ordering) and *unbounded in principle*, though practical implementations set limits or error thresholds. GNG and related methods (e.g. Fritzke's **Growing Cell Structures**) produce a *topology of units with neighbor connections* but usually **lack a layered hierarchy** – the network is a flat graph (often 2D or graph-structured) of prototypes. Notably, GNG employs a form of local Hebbian-like learning for connectivity (creating edges between frequently coactivated units) [9], and it respects spatial organization in that neurons often lie in a 2D embedding of the input distribution. **Unique-source semantics** (where each input source is uniquely assigned) do not explicitly appear in SOM/GNG – instead, all neurons compete for all inputs, though topological neighborhoods impose a form of locality.

Another family of constructive models is the **Adaptive Resonance Theory (ART)** by Grossberg and Carpenter. ART networks (e.g. **Fuzzy ART**) perform **online clustering with the ability to create new category nodes on demand** [10]. As each input arrives, the ART system either fits it into an existing category (if within a vigilance threshold) or **allocates a new category neuron** [11]. This growth is *triggered by a mismatch between the input and all existing learned patterns*. ART's learning is completely local (no error backpropagation); each category unit's weights adapt via simple rules, and new units are added to preserve previously learned categories (avoiding catastrophic forgetting). ART's growth is **deterministic** given a fixed input order and vigilance parameter, and in theory unbounded (it will keep adding categories for novel inputs as needed). Classic ART implementations form a two-layer hierarchy (input layer to category layer, sometimes coupled with a second ART module for supervised mapping as in ARTMAP), but do not typically arrange neurons in a multi-layer *deep* hierarchy. There is also no spatial layout like a 2D grid – category units are abstract nodes. Each input feature dimension has *defined connections to every category node*, rather than unique source assignments (indeed, Fuzzy ART uses complement coding, feeding each feature and its complement to all category nodes [12]). Thus, while GrowNet and ART both emphasize **stable incremental learning**, GrowNet's structured multi-layer growth and spatial wiring differ significantly from ART's flat category expansion.

Beyond these classic systems, there has been renewed interest in **dynamic neural architectures** in modern deep learning. **Progressive Neural Networks** (Rusu et al., 2016) introduce a form of growth across tasks: when a new task arrives, a *new column of network layers is added* while freezing the previous ones [13]. Lateral connections are added from old to new layers to allow knowledge transfer [14]. This results in a growing *multi-column architecture* that is **immune to forgetting** (since old columns are not altered) and leverages prior features [14]. Progressive Nets thus grow *modules (layers/columns)* deterministically for each new task; the growth is bounded by the number of tasks and is manually triggered by task delineation (not autonomously during a single task's training). The architecture is hierarchical (each column is deep), but the growth occurs at the level of duplicating entire layer stacks, rather than inserting neurons within an existing layer. There is no explicit 2D spatial structure or unique-source constraint in Progressive Nets – they typically use standard fully-connected or convolutional layers for each column, with conventional weight sharing where applicable.

A related concept is the **Dynamic Expandable Network (DEN)** for continual learning [15]. In DEN (Yoon et al., 2018), when a new task arrives, the network selectively **adds new neurons to existing layers** if needed to accommodate new knowledge, while reusing neurons that are still relevant. This approach uses a criterion based on sparseness and performance to decide growth and is paired with group sparsity regularization to keep the network compact. Growth in DEN is partially deterministic (depending on hyperparameters and thresholds) and **bounded by design** (it expands only to the extent necessary for each task). Like GrowNet, DEN operates at the neuron level and preserves previously learned features; however, DEN's trigger for growth is task-level performance (global), and it still uses backpropagation for weight updates (with selective freezing of old weights). DEN does not enforce a spatial layout or unique source assignments – it extends a standard MLP or CNN architecture.

In architecture search and **structural optimization**, approaches like **DARTS (Differentiable Architecture Search)** have been proposed to *learn network structure* within a continuous optimization framework [16]. DARTS defines a superset of possible connections/operations and uses gradient descent to adjust continuous parameters that eventually determine the discrete architecture [17]. While DARTS effectively *selects* connections (and can be seen as "growing" some paths while pruning others), it does **not add new neurons or layers beyond the initial search space**. Instead, it optimizes a fixed-capacity super-network. The process is **reproducible** given the same initial random seed and is typically one-shot (not ongoing growth during training, but a search phase followed by final architecture fixing). DARTS yields *fully feed-forward hierarchies* (often cell-based CNNs), and it assumes a predetermined spatial organization (e.g. convolutional cells operating on image grids) but does not introduce any novel spatial wiring beyond what convolution itself provides. The **GrowNet** approach, in contrast, *constructs the architecture incrementally during training* by deterministic rules, rather than searching over a predefined topology with gradient signals.

Another form of dynamic structure is seen in **Capsule Networks** (Sabour et al., 2017). Capsules do not grow new units during training; instead they employ **dynamic routing** of signals between fixed layers of "capsules." In the *routing-by-agreement* mechanism, lower-level capsules iteratively decide which higher-level capsule to send their output to, based on agreement between their prediction and the higher capsule's state [11]. This process dynamically adjusts **connection weights on the fly** for each input, effectively constructing a unique pathway through the network for each example. While not growing the architecture, this introduces a form of *structural plasticity in usage*: connections are input-dependent. Capsule Networks thus model a **hierarchy** (parts to wholes) and use an explicit *spatial* arrangement (e.g. capsules corresponding to positions in an image), with each capsule output typically routed to only one parent in the next layer (somewhat akin to a "unique destination" semantic for each input's output). GrowNet likewise emphasizes a form of local routing ("tracts" and focus windows) but differs by *actually adding new neurons/ layers* over time. Other dynamic routing frameworks, such as **Mixture-of-Experts** with gating (Shazeer et al., 2017) or **Routing Networks** [18], similarly keep the model size fixed but change the active sub-network per input. These methods show that *dynamic connectivity* can improve efficiency and representational power, but they do not address **growth of new units** as GrowNet does.

Recently, there have been explicit attempts to mimic **structural plasticity and neurogenesis** in deep networks. **Neurogenesis Deep Learning (NDL)** (Draelos et al., 2017) is inspired by adult neurogenesis in the brain [19]. NDL trains an autoencoder and **adds new hidden neurons** when the network's reconstruction error on novel data exceeds a threshold, indicating the current network cannot adequately represent a new pattern [20]. Each new neuron is integrated and trained on the new data while freezing the rest of the network to prevent forgetting [21]. This approach addresses the stability-plasticity dilemma by allocating new capacity for new information [22]. Growth in NDL is *triggered by a reconstruction error criterion* and is **deterministic** given that criterion. It is theoretically unbounded (the network can keep growing with endless novel inputs), though in practice one might cap growth. NDL focuses on unsupervised learning (autoencoders) and grows width (neuron count) in existing layers. **Grow-and-Prune** techniques, such as Dai et al. (2019) [23] and other works by Jha and colleagues, also modify network structure during training. These methods often start with a sparse network and **grow new connections or neurons based on gradient signals** (e.g. add a connection where the gradient of the loss would be large if that connection existed) and then **prune low-weight connections** to encourage sparsity [23]. The **NeST tool** [23] exemplifies this: it iteratively alternates **gradient-based growth of neurons/connections** with magnitude-based pruning, yielding compact networks that retain accuracy. Growth here is *driven by global error gradients* (hence not purely local learning) and can be partially random (depending on initialization of new weights). It is typically done in phases and can be reproduced given the same random seed. These approaches do create *layered hierarchies* (since they usually work within a standard feed-forward architecture), but they do not usually incorporate the notion of spatially localized wiring or one-to-one source semantics – new connections are often fully connected or convolutional as in the original architecture. **GrowNet differs** in that it prescribes *deterministic local wiring rules* (e.g. windowed receptive fields and unique source assignments) when new neurons or layers grow, whereas grow-and-prune methods often rely on random initial wiring or dense connectivity before pruning.

Finally, we note there are evolutionary approaches like **NEAT (NeuroEvolution of Augmenting Topologies)** [24] that also produce network growth. NEAT evolves a population of networks, starting from minimal structures and **mutating the topology by adding neurons and connections over generations** [24]. This yields increasingly complex networks that are optimized via genetic algorithms rather than gradient descent. While conceptually similar in growing topology, NEAT's growth is *stochastic* and driven by evolutionary selection, not deterministic rules; it requires maintaining many network variants and speciation to protect innovations [25]. In contrast, GrowNet performs *online, single-network growth* during learning, which is more efficient for continuous learning scenarios. NEAT also typically does not impose structured hierarchical layers or spatial locality – it can form arbitrary graphs – whereas GrowNet enforces a layered regional structure (slot → neuron → layer → region hierarchy) with 2D spatial constraints.

**Positioning Matrix:** Table 1 summarizes key differences among representative prior works and **GrowNet** along axes of growth mechanism, triggers, determinism, hierarchy, and wiring semantics.

## Positioning Matrix

| **Approach** | **Growth Mechanism** (what grows & how) | **Trigger & When** (when/ how growth is initiated) | **Deterministic & Bounded?** | **Hierarchy Modeled?** | **Spatial Wiring or Unique-Source?** |
|---|---|---|---|---|---|
| **Cascade-Correlation** [1] [4] | Adds one hidden **neuron** at a time, trained on residual; weights frozen after addition. | Triggered when existing network can't reduce error further; adds neuron during training until error goal met. | Deterministic growth (given data order); network size *adapts* (not fixed) but stops when error plateaus (bounded by training convergence). | Forms a shallow **cascade** (new neurons feed forward from all prior neurons); not a multilayer *stack* but incremental depth. | **No 2D spatial layout** (fully connected inputs); no unique-source constraint (each new neuron takes all prior as input). |
| **Growing Neural Gas** [6] [8] | Adds new **units** (prototypes) and **connections** in a graph; inserts node between highest-error unit and its neighbor. | Triggered periodically (every N inputs) based on accumulated quantization **error**; runs throughout unsupervised training. | Largely deterministic (for fixed random seed & input order); growth continues until error threshold or max units (unbounded if threshold never met). | **No layered hierarchy** – units form an adaptive topology (graph) in input space. | Often arranged in **2D/metric space** (embedding input topology); no unique source semantics (all inputs connect to all units via distance). |

| Approach | **Growth Mechanism** (what grows & how) | **Trigger & When** (when/ how growth is initiated) | **Deterministic & Bounded?** | **Hierarchy Modeled?** | **Spatial Wiring or Unique-Source?** |
|---|---|---|---|---|---|
| **Fuzzy ART** 11 | Adds new **category neuron** for novel input patterns; retains all prior categories (vigilance-based). | Triggered whenever input does not sufficiently match any existing category (fails vigilance); checked on each new input. | Deterministic (given vigilance and sample order); grows until each input fits a category (unbounded in worst case, bounded if data has finite clusters). | Two-layer (feature → category) **classification hierarchy**; no deep multi-layer features. | **No spatial wiring** (fully connected input-to-category); no unique source (each feature connects to all category units). |
| **Self-Organizing Map (fixed)** | *(No structural growth by default)* – fixed grid of neurons that self-tune weights. (Variations like GSOM add neurons.) | N/A for basic SOM (structure fixed); GSOM triggers growth when map's quantization error exceeds threshold in a region. | N/A (SOM size set by user; GSOM growth deterministic given threshold). | Neurons on a 2D grid (one-layer map); **no multiple layers**. | **2D neighbor connections** on grid; no unique source (every input seen by all neurons). |

| **Approach** | **Growth Mechanism** (what grows & how) | **Trigger & When** (when/ how growth is initiated) | **Deterministic & Bounded?** | **Hierarchy Modeled?** | **Spatial Wiring or Unique-Source?** |
|---|---|---|---|---|---|
| **Progressive Networks** [26] | Adds new **column of layers** for each new task; lateral connections from old to new. | Triggered *per task* in sequential multi-task learning; a new column is allocated when a task begins. | Deterministic given task sequence; growth is bounded by number of tasks (one column per task). | Yes – each column is a deep network (with fixed depth), and **multiple parallel columns** form a higher-level architecture (one per task). | Uses whatever spatial structure base model has (e.g. conv layers with image topology), but *no novel wiring* beyond lateral links; no unique-source semantics (standard dense or conv connections). |
| **DARTS (Arch. Search)** [16] | Optimizes **connection pattern** among fixed candidate ops; *selects* architecture (no new units beyond search space). | Continuous relaxation of arch encoding optimized by gradient; effectively "grows" strongest paths during search phase. | Reproducible given same initialization; architecture size is fixed (search space bounded; no new layers beyond it). | Yes – yields a **layered cell-based network** (e.g. conv cells); hierarchy is defined by chosen connections. | Inherits spatial structure of chosen ops (e.g. conv filters); no concept of unique source (standard multiple inputs to layers as selected). |

| Approach | **Growth Mechanism** (what grows & how) | **Trigger & When** (when/ how growth is initiated) | **Deterministic & Bounded?** | **Hierarchy Modeled?** | **Spatial Wiring or Unique-Source?** |
|---|---|---|---|---|---|
| **Capsule Network** 11 | *No new capsules added* after initialization; instead **dynamic routing** adjusts which higher-level capsule each lower capsule's output goes to (per input). | Triggered each forward pass: an iterative routing algorithm routes signals based on agreement (higher agreement → strengthen route). | Deterministic routing results for a given input (algorithm is fixed); *network topology is static* (bounded), only connection weights vary per input. | Yes – layered **part-whole hierarchy** (e.g. strokes to digits); fixed architecture of capsule layers. | **Spatial grouping** of neurons into capsules; routing enforces a form of **unique target selection** for each lower capsule (it sends output mostly to one parent capsule). |
| **Mixture-of-Experts** (e.g. Shazeer 2017) | *No new experts added at runtime*; **gating network** dynamically selects a subset of expert networks to activate per input. | Gating decision computed for each input, routing it to top-$k$ experts (typically based on learned gating weights). | Stochastic or deterministic gating (depending on design); model size fixed (experts predetermined). | Yes – typically a two-level hierarchy (gate and experts, or experts as parallel sub-layers). | No spatial layout; experts are abstract sub-networks; no unique source (multiple experts can be active, though typically each input uses only a few). |

| Approach | **Growth Mechanism** (what grows & how) | **Trigger & When** (when/ how growth is initiated) | **Deterministic & Bounded?** | **Hierarchy Modeled?** | **Spatial Wiring or Unique-Source?** |
|---|---|---|---|---|---|
| **Neurogenesis Deep Learning** [19] [20] | Adds new **hidden neurons** in an autoencoder's layers to encode new patterns; uses intrinsic replay to retain old info. | Triggered when reconstruction **error spikes** on novel data (indicating existing units insufficient); checked during continuous learning. | Deterministic (rule-based triggers); in continual learning it can grow without bound (but practical limits or decay can be set). | Yes – multi-layer autoencoder; new neurons inserted in existing layers (maintaining overall depth). | No special spatial wiring (fully connected layers); no unique source semantics (new neurons connect into existing layer like others). |
| **Grow-and-Prune (NeST)** [23] | Alternates between **growing connections/ neurons** (based on gradient magnitude or other signals) and **pruning** small-weight connections; starts from sparse seed network. | Triggered periodically during training: grow phase adds where gradient or loss suggests capacity need; prune phase removes low-weight connections; repeat. | Partly stochastic (random initial weights for new connections) but using global criteria; final size is bounded by design (pruning keeps it sparse). | Yes – uses standard feed-forward hierarchy (e.g. LeNet, ResNet) but finds a **sparser sub-network** via grow-prune. | Uses standard layer connectivity (e.g. convolutional kernels) – no novel spatial layout beyond underlying architecture; no unique-source (connections are added in dense fashion then pruned). |

| Approach | **Growth Mechanism** (what grows & how) | **Trigger & When** (when/ how growth is initiated) | **Deterministic & Bounded?** | **Hierarchy Modeled?** | **Spatial Wiring or Unique-Source?** |
|---|---|---|---|---|---|
| **Dynamically Expandable Net** (DEN 2018) | Adds **neurons in existing layers** when new task arrives (if needed); also frees unused capacity via pruning. | Triggered at new task onset: if model's performance on new task is poor, expand layers by Δ neurons (determined by validation loss criteria). | Deterministic given thresholding scheme; bounded by number of tasks and max allowed expansion per task. | Yes – retains the original layered structure, simply wider; old and new neurons form composite hierarchy. | No special spatial or local wiring (follows original architecture); no unique sources (new neurons integrate like any other in layer). |
| **NEAT (Neuro-evolution)** [24] | *Evolutionary:* adds **neurons and connections** via mutations over generations; population-based search optimizes both weights and topology. | Triggered by genetic operations (random mutations) during evolution; over many generations network topology complexifies. | Stochastic (random mutations + selection); not bounded (can keep adding until fitness stops improving or forced limits). | Flexible topology – can form multi-layer or irregular graphs; not explicitly layered unless evolved so. | No inherent spatial embedding (unless encoded); no unique-source constraint; topology is arbitrary within genetic encoding. |

**Table 1:** Comparison of prior methods to **GrowNet** (which **grows new slots, neurons, layers, and regions** deterministically when local capacity is saturated, uses two-phase learning without global backprop, organizes neurons in 2D **grid "regions" with windowed receptive fields**, and enforces **unique-source subscriptions** so that each input signal is received by only one slot in a given layer [27] [28]). GrowNet's growth is *triggered by local failure conditions* (e.g. a sequence of "fallback" events when no unused slot can sufficiently respond) and employs **cooldown periods** to ensure stability [29]. Unlike most prior art, GrowNet's growth is **fully deterministic and reproducible**, producing the same architecture given the same data and hyperparameters (no random mutations or search). It maintains a clear hierarchy (slots within neurons, neurons within layers, layers within regions) and explicitly leverages 2D spatial organization and one-to-one source allocation [27] – a combination of features not found in earlier growing neural networks.

## References (BibTeX)

```
@inproceedings{FahlmanL89,
  author    = {Scott E. Fahlman and Christian Lebiere},
  title     = {The Cascade-Correlation Learning Architecture},
  booktitle = {Advances in Neural Information Processing Systems 2 (NIPS 1989)},
  editor    = {D. S. Touretzky},
  pages     = {524--532},
  year      = {1990},
  publisher = {Morgan Kaufmann},
  url       = {http://papers.nips.cc/paper/207-the-cascade-correlation-learning-
architecture},
}

@inproceedings{Fritzke94,
  author    = {Bernd Fritzke},
  title     = {A Growing Neural Gas Network Learns Topologies},
  booktitle = {Advances in Neural Information Processing Systems 7 (NIPS 1994)},
  editor    = {G. Tesauro and D. S. Touretzky and T. K. Leen},
  pages     = {625--632},
  year      = {1995},
  publisher = {MIT Press},
}

@article{Carpenter91,
  author    = {Gail A. Carpenter and Stephen Grossberg and David B. Rosen},
  title     = {{Fuzzy ART}: Fast Stable Learning and Categorization of Analog
Patterns by an Adaptive Resonance System},
  journal   = {Neural Networks},
  volume    = {4},
  number    = {6},
  pages     = {759--771},
  year      = {1991},
  doi       = {10.1016/0893-6080(91)90056-B},
}

@article{Kohonen90,
  author    = {Teuvo Kohonen},
  title     = {The Self-Organizing Map},
  journal   = {Proceedings of the IEEE},
  volume    = {78},
  number    = {9},
  pages     = {1464--1480},
  year      = {1990},
  doi       = {10.1109/5.58325},
}

@inproceedings{Rusu16,
  author    = {Andrei A. Rusu and Neil~C. Rabinowitz and Guillaume Desjardins
```

```
and Hubert Soyer and James Kirkpatrick and Koray Kavukcuoglu and Razvan Pascanu
and Raia Hadsell},
  title     = {Progressive Neural Networks},
  booktitle = {Proceedings of the 2016 International Conference on Machine
Learning (ICML)},
  year      = {2016},
  url       = {https://arxiv.org/abs/1606.04671},
}

@inproceedings{Liu19,
  author    = {Hanxiao Liu and Karen Simonyan and Yiming Yang},
  title     = {{DARTS}: Differentiable Architecture Search},
  booktitle = {6th International Conference on Learning Representations (ICLR
2019)},
  year      = {2019},
  url       = {https://arxiv.org/abs/1806.09055},
}

@inproceedings{Sabour17,
  author    = {Sara Sabour and Nicholas Frosst and Geoffrey~E. Hinton},
  title     = {Dynamic Routing Between Capsules},
  booktitle = {Advances in Neural Information Processing Systems 30 (NIPS
2017)},
  pages     = {3856--3866},
  year      = {2017},
  url       = {https://arxiv.org/abs/1710.09829},
}

@article{Draelos17,
  author    = {Timothy~J. Draelos and Nadine~E. Miner and Christopher~C. Lamb
and Jonathan~A. Cox and Craig~M. Vineyard and Kristofor~D. Carlson and
William~M. Severa and Conrad~D. James and James~B. Aimone},
  title     = {Neurogenesis Deep Learning: Extending Deep Networks to
Accommodate New Classes},
  journal   = {2017 International Joint Conference on Neural Networks (IJCNN)},
  pages     = {526--533},
  year      = {2017},
  doi       = {10.1109/IJCNN.2017.7965898},
}

@inproceedings{Dai19,
  author    = {Xiaoliang Dai and Hongxu Yin and Niraj~K. Jha},
  title     = {Incremental Learning Using a Grow-and-Prune Paradigm with
Efficient Neural Networks},
  booktitle = {Proceedings of the 2019 {IEEE} International Symposium on
Circuits and Systems (ISCAS)},
  pages     = {1--5},
  year      = {2019},
```

```
  doi       = {10.1109/ISCAS.2019.8702331},
  note      = {arXiv:1905.10952},
}

@inproceedings{Yoon18,
  author    = {Jaehong Yoon and Eunho Yang and Jeongtae Lee and Sung~Ju Hwang},
  title     = {Lifelong Learning with Dynamically Expandable Networks},
  booktitle = {International Conference on Learning Representations (ICLR
2018)},
  year      = {2018},
  url       = {https://arxiv.org/abs/1708.01547},
}

@article{Stanley02,
  author    = {Kenneth~O. Stanley and Risto Miikkulainen},
  title     = {Evolving Neural Networks Through Augmenting Topologies},
  journal   = {Evolutionary Computation},
  volume    = {10},
  number    = {2},
  pages     = {99--127},
  year      = {2002},
  doi       = {10.1162/106365602320169811},
}

@misc{Badirli20,
  author       = {Sarkhan Badirli and Xuanqing Liu and Zhengming Xing and
Avradeep Bhowmik and Khoa Doan and Sathiya S. Keerthi},
  title        = {{GrowNet}: Gradient Boosting Neural Networks},
  howpublished = {arXiv preprint arXiv:2002.07971},
  year         = {2020},
  note         = {Under review},
}
```

---


1 3 4 The Cascade-Correlation Learning Architecture
https://proceedings.neurips.cc/paper/1989/file/69adc1e107f7f7d035d7baf04342e1ca-Paper.pdf

2 dblp: BibTeX record conf/nips/FahlmanL89
https://dblp.org/rec/conf/nips/FahlmanL89.html?view=bibtex

5 ijsce.org
https://www.ijsce.org/wp-content/uploads/papers/v3i3/C1666073313.pdf

6 8 9 A Growing Neural Gas Network Learns Topologies
https://proceedings.neurips.cc/paper/1994/file/d56b9fc4b0f1be8871f5e1c40c0067e7-Paper.pdf

7 dblp: BibTeX record conf/nips/Fritzke94
https://dblp.dagstuhl.de/rec/conf/nips/Fritzke94.html?view=bibtex

[10] [11] [1710.09829] Dynamic Routing Between Capsules
https://arxiv.org/abs/1710.09829

[12] PII: 0893-6080(91)90056-B
https://sites.bu.edu/steveg/files/2016/06/CarGroRos1991NNFuzzyART.pdf

[13] [14] [26] [1606.04671] Progressive Neural Networks
https://arxiv.org/abs/1606.04671

[15] Dynamically Expandable Graph Convolution for Streaming ...
https://dl.acm.org/doi/10.1145/3543507.3583237

[16] [17] [1806.09055] DARTS: Differentiable Architecture Search
https://arxiv.org/abs/1806.09055

[18] Dynamic neural networks: advantages and challenges
https://academic.oup.com/nsr/article/11/8/nwae088/7624214

[19] [22] [1612.03770] Neurogenesis Deep Learning
https://arxiv.org/abs/1612.03770

[20] Dynamic Relevance-Weighting-Based Width-Adaptive Auto-Encoder
https://www.mdpi.com/2076-3417/15/12/6455

[21] Continual lifelong learning with neural networks: A review
https://www.sciencedirect.com/science/article/pii/S0893608019300231

[23] [1711.02017] NeST: A Neural Network Synthesis Tool Based on a Grow-and-Prune Paradigm
https://arxiv.org/abs/1711.02017

[24] [25] Evolving Neural Networks Through Augmenting Topologies
https://nn.cs.utexas.edu/?stanley:ec02

[27] [28] [29] READ_ORDER.md
file://file_00000000f60c622fa0ffe992ba6598a4