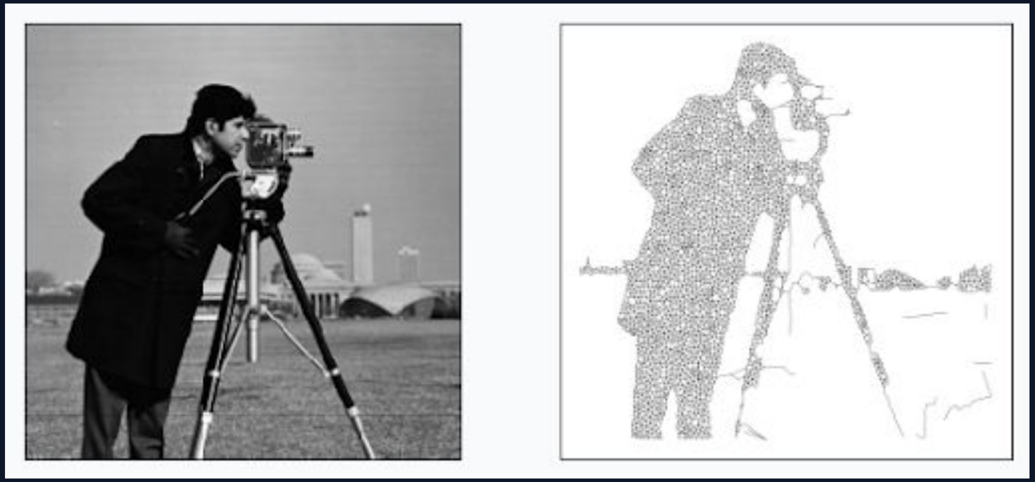
Self-Growing Neural Networks

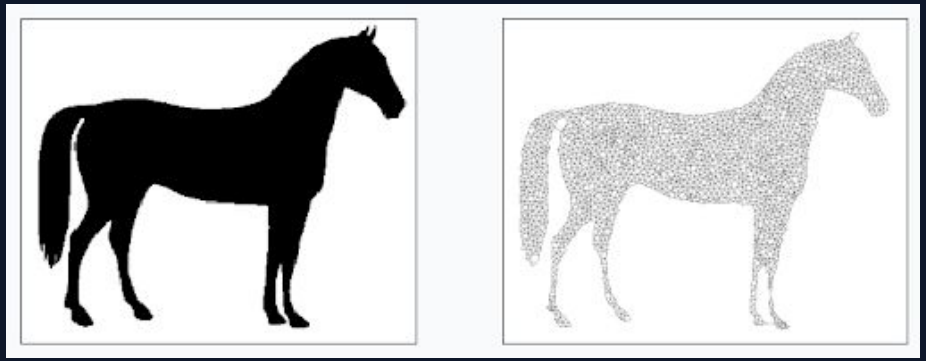
From Fixed Grids to Adaptive Graphs

Presented by Tytus Felbor & Justin Yau

Presentation Outline

- **Evolution:** SOM → Neural Gas → Growing Neural Gas
- **Comparison:** GNG vs. Growing SOM (GSOM)
- **Algorithm:** Step-by-step mechanics & Math
- **Case Study:** The "2-Moons" Benchmark Problem
- **Visualizations:** 5-stage learning progression
- **Results:** Metrics & Complexity Analysis





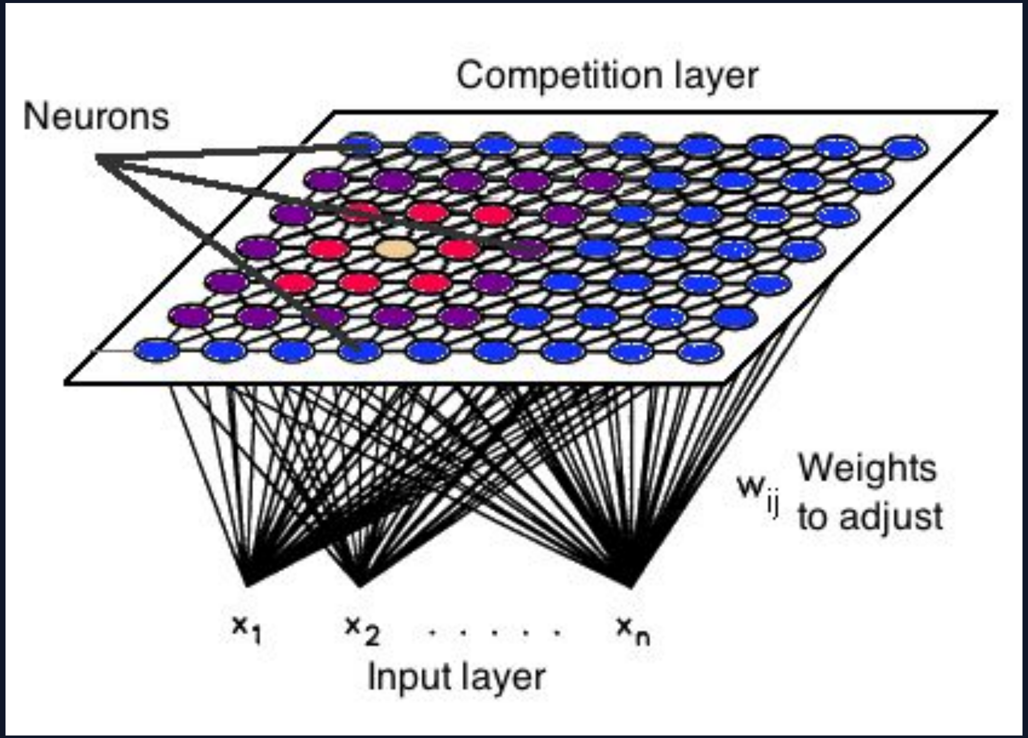
The Foundation: Self-Organizing Maps

Concept (Kohonen, 1982)

Projects high-dimensional input onto a lower-dimensional (2D) discrete grid.

Neurons are locked in a fixed topology.

```
w_new = w_old + learning_rate × neighborhood_function × (input - w_old)
```



Limitations

- Network size must be specified in advance
- If the chosen size is too small, data is poorly represented; if too large, training becomes inefficient
- Without prior knowledge of data structure, choosing optimal grid size can be difficult
- high computational complexity

Moving Beyond Grids: Neural Gas

Martinetz and Schulten (1991, 1994)

Unlike SOM, which does "ordered" vector quantization using a predefined grid, Neural Gas performs "unordered" vector quantization. Neurons are defined only by their weight vectors in input space, not by pre-assigned grid positions.

Edge Aging Mechanism

To keep the network adaptive, edges are aged. If an edge is not reactivated (i.e., the two connected neurons are not the two closest to some input), it gradually expires and is removed.

Advantages Over SOM

- No predefined grid structure; topology emerges from data
- Uses Competitive Hebbian Learning to create optimal topology (Delaunay triangulation)
- More flexible network structure

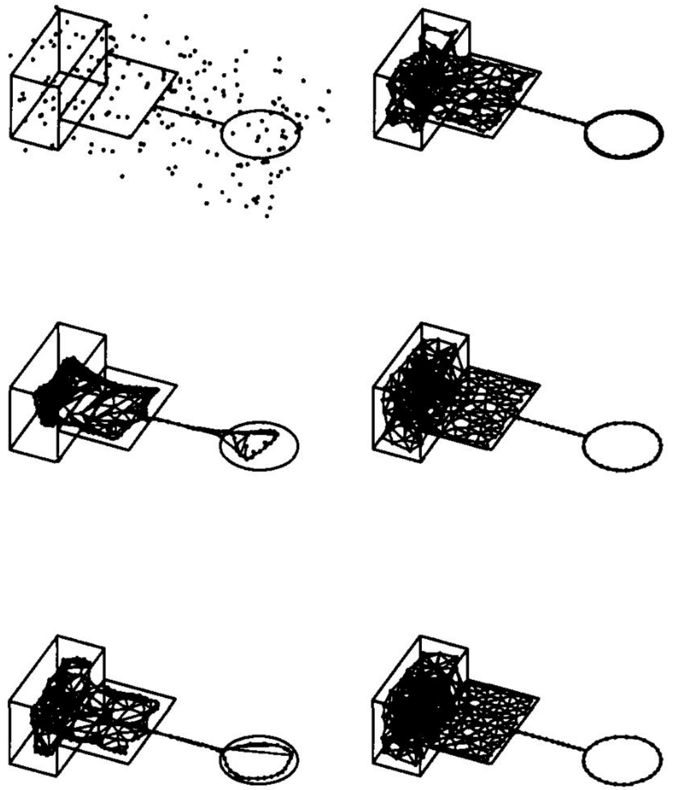


FIGURE 6. The competitive Hebb rule together with the neural gas algorithm forming a topology preserving map of a topologically heterogeneously structured manifold. The given manifold *M* consists of a three-dimensional (right parallelepiped), a two-dimensional (rectangle), and a one-dimensional (circle and connecting line) subset. The neural gas algorithm as an efficient input driven vector quantization procedure distributes the pointers w_i over the manifold *M*. With each presented pattern $v \in M$ the competitive Hebb rule establishes or refreshes an edge of the induced Delaunay triangulation. Depicted are the initial state, the network after 5000, 10,000, 15,000, 25,000, and at the final state after 40,000 adaptation steps (from top left to bottom right). At the end of the adaptation procedure the network (graph) forms a perfectly topology preserving map that reflects the topological structure and the dimensionality of the manifold *M*.

The Innovation: Growing Neural Gas (1995)

Constant Parameters

Learning rates do not decay. The system never "freezes" and can adapt indefinitely.

Error-Driven Growth

New neurons are inserted strictly based on local accumulated error, not a fixed schedule.

Embedded Ready

Linear complexity and sparse graph structure make it memory efficient.

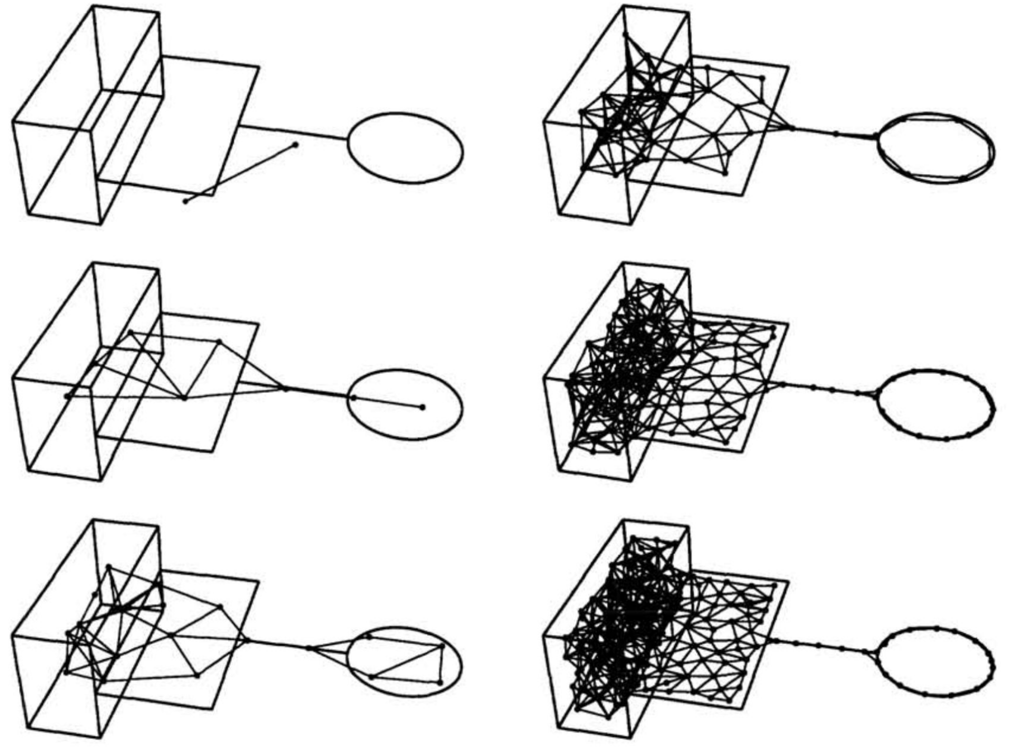


Figure 2: The "growing neural gas" network adapts to a signal distribution which has different dimensionalities in different areas of the input space. Shown are the initial network consisting of two randomly placed units and the networks after 600, 1800, 5000, 15000 and 20000 input signals have been applied. The last network shown is not the necessarily the "final" one since the growth process could in principle be continued indefinitely. The parameters for this simulation were: $\lambda = 100$, $\epsilon_b = 0.2$, $\epsilon_n = 0.006$, $\alpha = 0.5$, $a_{max} = 50$, $d = 0.995$.

GNG Algorithm

Initialization:

Start with exactly 2 randomly placed neurons and no edges.

For Each Input Sample:

Step 1 - Identify Nearest Neighbors

Step 2 - Adapt Neurons (Hebb-like Learning)

Step 3 - Accumulate Error

Step 4 - Topology Preservation (Competitive Hebbian Learning)

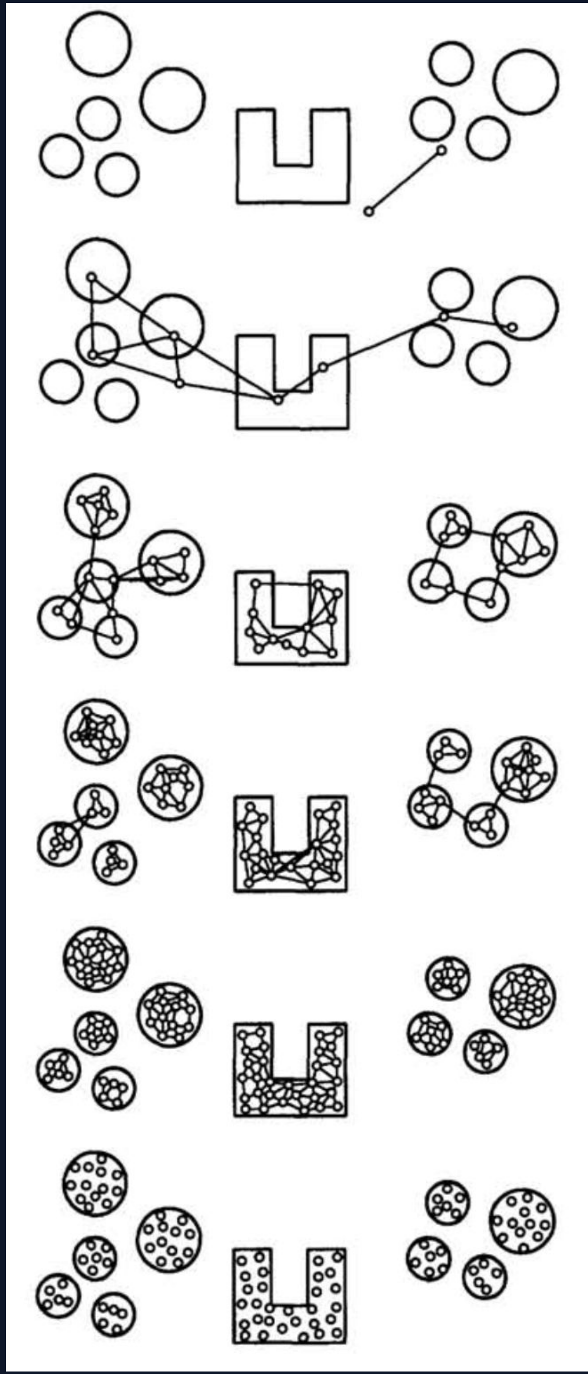
Step 5 - Edge Removal (Local Aging)

Step 6 - Error Decay

Step 7 - Periodic Node Insertion

Stopping Criterion:

Continue this process until the network reaches a desired size or quantization error plateaus.



GNG vs. Growing SOM (GSOM) : Algorithmic Differences

	GNG	GSOM
Network Topology	*Graph structure (any topology)*	*Fixed 2D grid structure (rectangular)*
Node Growth	*Error-driven, continuous*	*Boundary-based, controlled by Spread Factor*
Growth Pattern	*Insert one node at a time (between two neighbors)*	*Grow multiple nodes simultaneously on grid boundary*
Node Placement	*Halfway between high-error nodes*	*On boundary edges (rectangular grid positions)*
Parameters	*Constant throughout*	*Learning rate and neighborhood shrink over time (though exponentially)*
Adaptation Neighborhood	*Topological neighbors (determined by edges)*	*Grid-defined neighbors (rectangular distance)*
Training Termination	*User-defined criterion*	*When quantization error plateaus or max size reached*
Computational Complexity	*$O(N)$ where N = number of nodes*	*$O(N^{1.5})$ due to grid operations*
Grid/Graph	*Sparse graph (typically N edges)*	*Complete 2D grid structure*

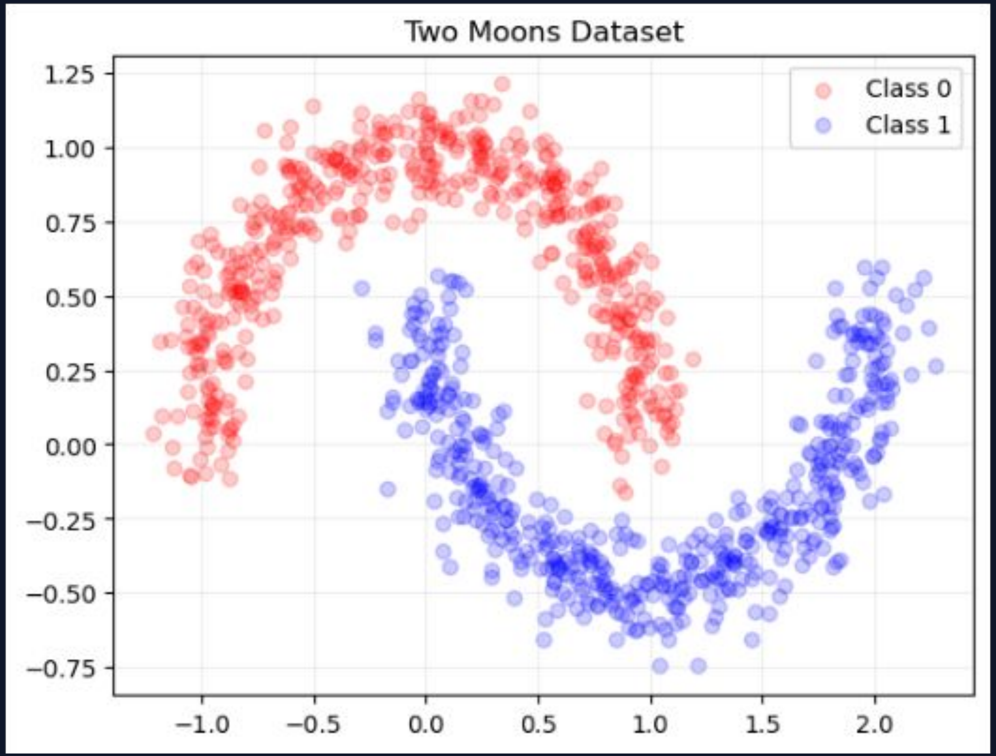
Benchmark: The "2-Moons" Problem

Dataset Characteristics

300 data points arranged in two interleaving "crescent" shapes. A classic non-convex clustering challenge.

The Challenge

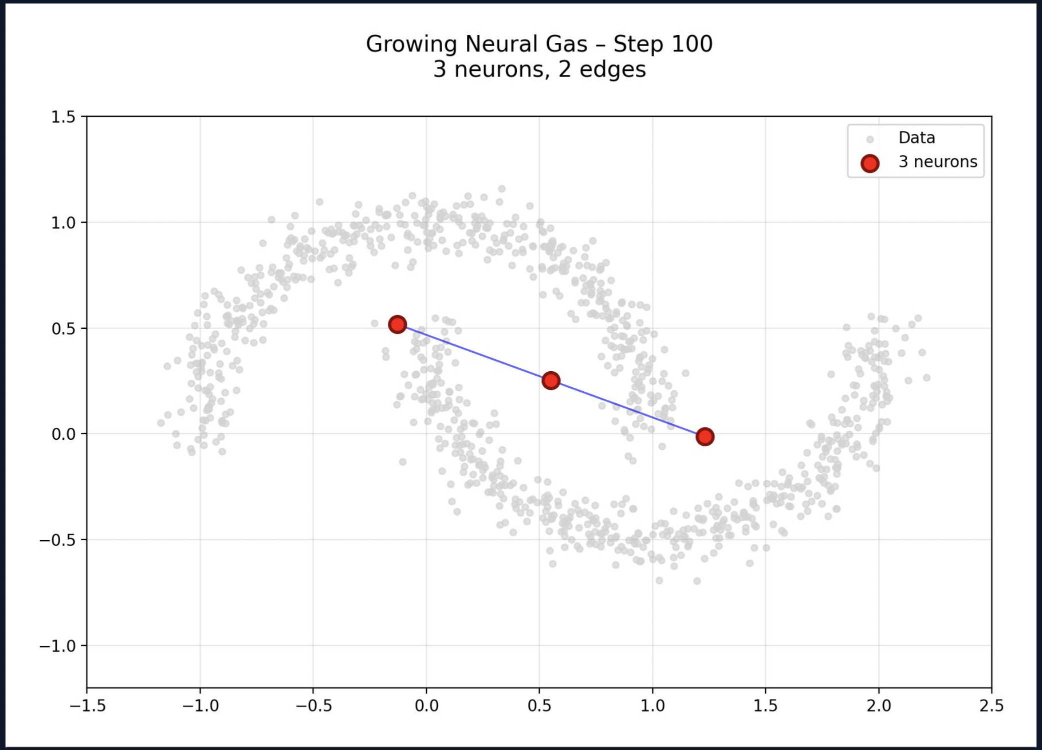
Points in different moons can be Euclidean-close but are topologically distinct. The network must discover two separate components without supervision.



Visualization 1: Initialization

Step 100: The Blank Slate

- **State:** 2 random neurons (initialized in the center).
- **Status:** High quantization error. No topology established yet.
- The algorithm begins with minimal resources and grows only as needed.

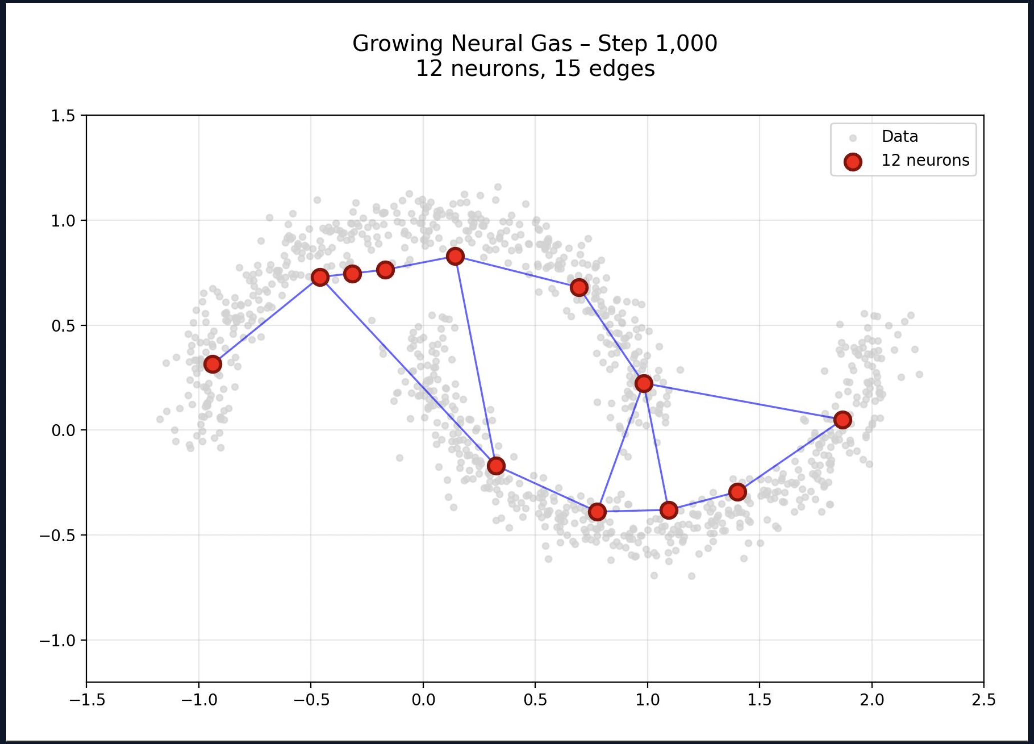


3 neurons visible, because after step 99 a new neuron is added

Visualization 2: Early Training

Step 1,000: "Waking Up"

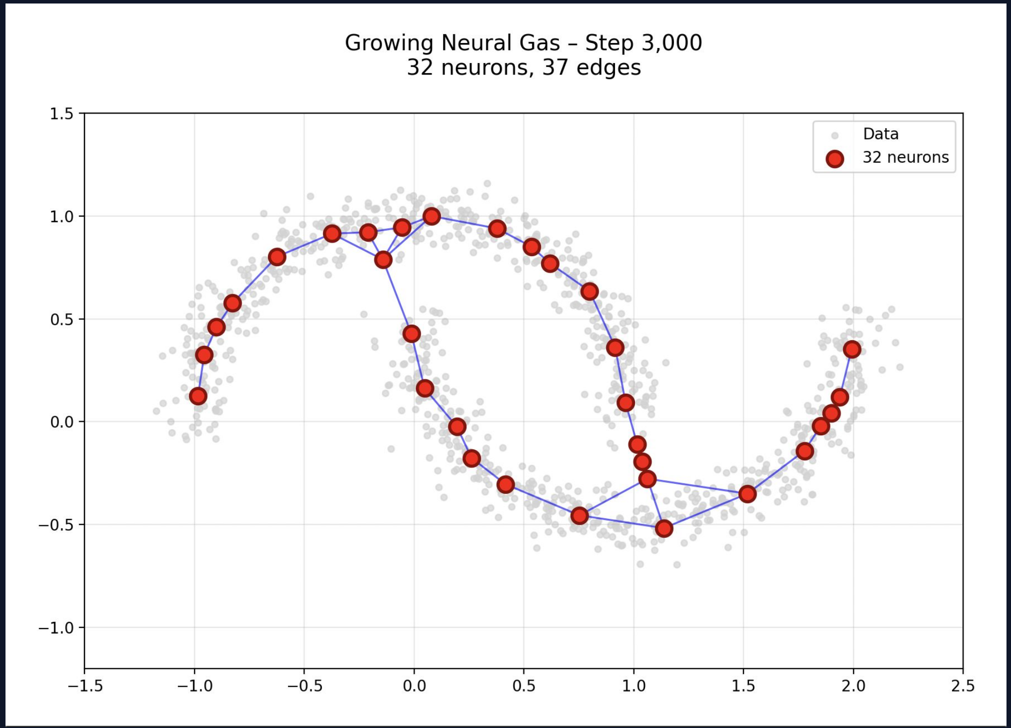
- **Migration:** Neurons move rapidly toward centers of the data.
- **Topology:** First edges form between closest units.
- **Growth:** New neurons are inserted in regions with the highest error (the largest gaps).



Visualization 3: Structure Emergence

Step 3,000: Defining Shapes

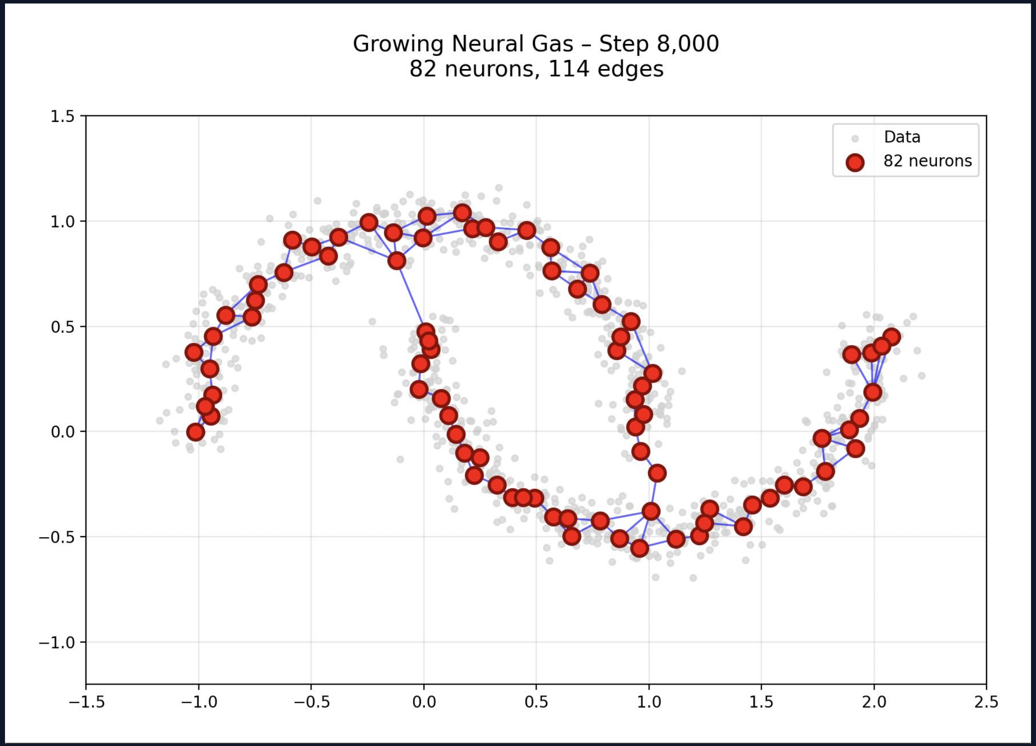
- **Separation:** Edges crossing the gap between moons age and die because they are rarely activated.
- **Formation:** Chains of neurons begin to trace the crescent shapes.



Visualization 4: Maturation

Step 8,000: Convergence

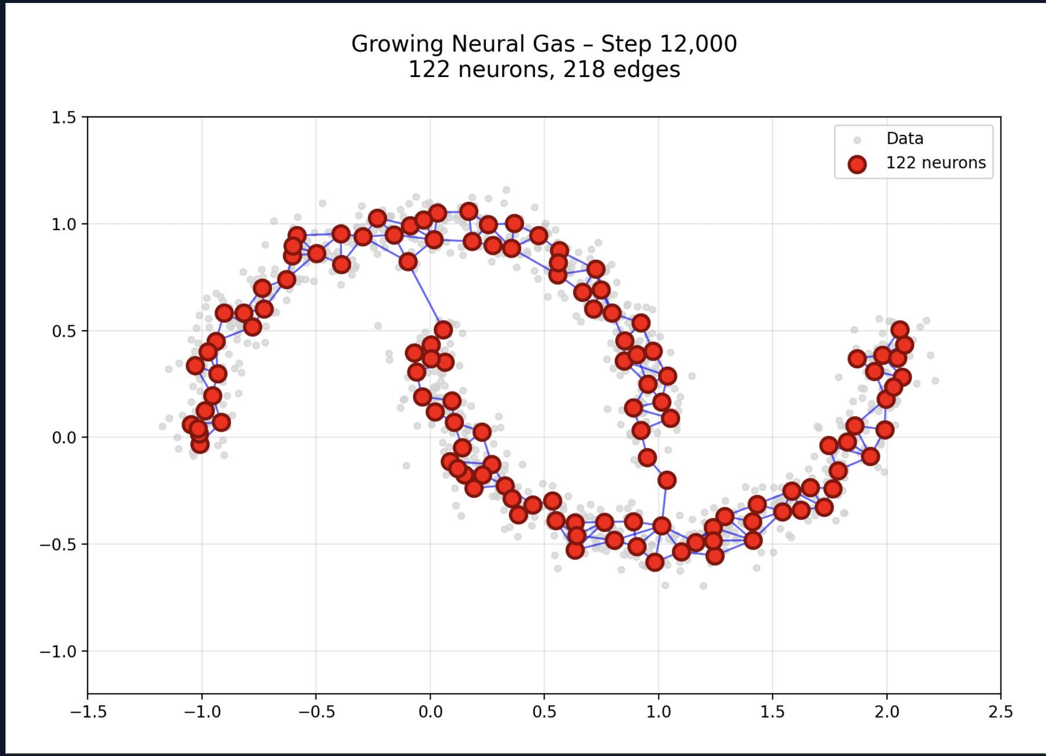
- **Coverage:** Dense population (approx 20-30 neurons) covers both moons evenly.
- **Stability:** Error has plateaued; new insertions are rare.
- **Efficiency:** Edges exist only where data density supports them.



Visualization 5: Final Topology

Step 12,000: Induced Delaunay Triangulation

- **Result:** Two completely separate connected components. No edges bridge the gap.
- **Sparsity:** Average degree is low (2-4 edges), making traversal efficient.
- **Accuracy:** 100% topological correctness for this dataset.



Quantitative Results & Performance

Metrics (2-Moons)

Quantization Error: Reduced by ~85% (0.4 → 0.05)

Sparsity Ratio: Only 2-4% of potential edges exist.

Convergence: 2-5 seconds (CPU) for 5000 iterations.

Embedded Complexity

Time: O(N) per sample (Linear vs Quadratic SOM).

Space: O(N) storage.

Scaling: Adding 10x data 10x time.

Key Takeaways

Self-Organizing

Learns structure without knowing the number of clusters (k) or network size.

Efficient

Linear complexity and constant parameters make it ideal for resource-constrained edge devices.

Topological

Successfully captures nonlinear manifolds (like 2-moons) where simple K-means would fail.

Thank you
& Questions?