Neural Networks in Lean4

mkaratarakis

May 22, 2025

Chapter 1

NN

We strictly follow Chapter 4 of [5] for the definitions of neural and Hopfield networks.

Definition 1. An (artificial) neural network is a directed graph $G = (U, C)$, where neurons $u \in U$ are connected by directed edges $c \in C$ (connections). The neuron set is partitioned as $U = U_{\text{in}} \cup U_{\text{out}} \cup U_{\text{hidden}}$, with $U_{\text{in}}, U_{\text{out}} \neq \emptyset$ and $U_{\text{hidden}} \cap (U_{\text{in}} \cup U_{\text{out}}) = \emptyset$. Each connection $(v, u) \in C$ has a weight w_{uv}, and each neuron u has real-valued quantities: network input net_u, activation act_u, and output out_u. Input neurons $u \in U_{\text{in}}$ also have a fourth quantity, the external input ext_u. The predecessors and successors of a vertex u in a directed graph $G = (U, C)$ are defined as $\text{pred}(u) = \{v \in V \mid (v, u) \in C\}$ and $\text{succ}(u) = \{v \in V \mid (u, v) \in C\}$ respectively. Each neuron u is associated with the following functions:

$$f_{\text{net}}^{(u)} : \mathbb{R}^{2|\text{pred}(u)|+\kappa_1(u)} \to \mathbb{R}, \quad f_{\text{act}}^{(u)} : \mathbb{R}^{1+\kappa_2(u)} \to \mathbb{R}, \quad f_{\text{out}}^{(u)} : \mathbb{R} \to \mathbb{R}.$$

These functions compute net_u, act_u, and out_u, where $\kappa_1(u)$ and $\kappa_2(u)$ count the number of parameters of those functions, which can depend on the neurons. Specifically, the new activation act'_u of a neuron u is computed as follows:

$$\text{act}'_u = f_{\text{act}}^{(u)}(f_{\text{net}}^{(u)}(w_{uv_1}, \dots, w_{uv_{\text{pred}(u)}}, f_{\text{out}}^{(v_1)}(\text{act}_{v_1}), \dots, f_{\text{out}}^{(v_{\text{pred}(u)})}(\text{act}_{v_{\text{pred}(u)}}), \sigma^{(u)}), \theta^{(u)})$$

where $\sigma^{(u)} = (\sigma_1^{(u)}, \dots, \sigma_{\kappa_1(u)}^{(u)})$ and $\theta = (\theta_1^{(u)}, \dots, \theta_{\kappa_2(u)}^{(u)})$ are the input parameter vectors.

Definition 2. A Hopfield network is a neural network with graph $G = (U, C)$ as described in the previous section, that satisfies the following conditions: $U_{\text{hidden}} = \emptyset$, and $U_{\text{in}} = U_{\text{out}} = U$, $C = U \times U - \{(u, u) \mid u \in U\}$, i.e., no self-connections. The connection weights are symmetric, i.e., for all $u, v \in U$, we have $w_{uv} = w_{vu}$ when $u \neq v$. The activation of each neuron is either 1 or -1 depending on the input. There are no loops, meaning neurons don't receive their own output as input. Instead, each neuron u receives inputs from all other neurons, and in turn, all other neurons receive the output of neuron u.

- The network input function is given by

$$\forall u \in U : \quad f_{\text{net}}^{(u)}(w_u, in_u) = \sum_{v \in U - \{u\}} w_{uv} \cdot \text{out}_v.$$

- The activation function is a threshold function

$$\forall u \in U : \quad f_{\text{act}}^{(u)}(\text{net}_u, \theta_u) = \begin{cases} 1 & \text{if } \text{net}_u \geq \theta_u, \\ -1 & \text{otherwise.} \end{cases}$$

- The output function is the identity

$$\forall u \in U: \quad f_{\text{out}}^{(u)}(\text{act}_u) = \text{act}_u.$$

Theorem 3 (Convergence Theorem for Hopfield networks (Theorem 8.1, [5]))**.** *If the activations of the neurons of a Hopfield network are updated asynchronously, a stable state is reached in a finite number of steps.*

Theorem 4 (Corollary of convergence Theorem for Hopfield networks (Theorem 8.1, [5]))**.** *If the neurons are traversed in an arbitrary, but fixed cyclic fashion, at most $n \cdot 2^n$ steps (updates of individual neurons) are needed, where n is the number of neurons of the network.*

Definition 5. We define asymmetric Hopfield networks as general neural networks with the same graph514 structure and functions as symmetric Hopfield networks but with this matrix decomposition515 property instead of symmetry.

Definition 6. The potential function for asymmetric Hopfield networks at time step k represents the energy of the network at time step k, considering that neuron ($k \mod n$) is being updated.

Lemma 7. *The significance of this potential function lies in its relationship to Lyapunov stability566 theory. We prove it is bounded regardless of the network's configuration.*

Definition 8. The Boltzmann distribution:

$$P(s) = \frac{e^{-E(s)/T}}{Z}$$

where $E(s)$ is the energy of state s, T is the temperature parameter and $Z = \sum_s e^{-E(s)/T}$ is the partition function.

Definition 9. For neuron updates, we use Gibbs sampling, as introduced by Geman and Geman [1], where a neuron u is updated according to :

$$P(s_u = +1 | s_{-u}) = \frac{1}{1 + e^{-2h_u/T}}$$

where h_u is the local field defined as $h_u = \sum_v w_{uv} s_v - \theta_u$.

This formula can be derived directly from the Boltzmann distribution by considering the conditional probability of a single neuron's state given all others:

$$P(s_u = +1 | s_{-u}) = \frac{P(s_u = +1, s_{-u})}{P(s_u = +1, s_{-u}) + P(s_u = -1, s_{-u})}$$

The energy difference between states with $s_u = +1$ and $s_u = -1$ is $\Delta E = -2h_u$, which gives us the formula above after substitution and simplification.

Definition 10. We also implement simulated annealing, as introduced by Kirkpatrick et al. [4], which systematically decreases the temperature T over time according to a cooling schedule:

$$T(t) = T_0 \times e^{-\alpha t}$$

where T_0 is the initial temperature and α is the cooling rate.

Definition 11. Another sampling method we formalize is the Metropolis-Hastings algorithm, introduced by Metropolis et al. [7] and later generalized by Hastings [2], which accepts or rejects proposed state changes with probability:

$$P(\text{accept}) = \min(1, e^{-(E(s')-E(s))/T})$$

where s' is the proposed state after flipping a neuron. This allows the network to sometimes move to higher energy states, helping it escape local minima.

Definition 12. To measure convergence to the equilibrium Boltzmann distribution, we use the total variation distance, as described by Levin and Peres [6] :

$$d_{TV}(\mu, \nu) = \frac{1}{2} \sum_{s} |\mu(s) - \nu(s)|.$$

Definition 13. The stochastic Hopfield Markov process, which models the evolution of Hopfield network states over discrete time steps using Gibbs sampling at fixed temperature. In this simplified model, the transition kernel is time-homogeneous (same for all steps).

Chapter 2

Hopfield's 1982 paper

This module formalizes the key mathematical concepts from J.J. Hopfield's 1982 paper [3] "Neural networks and physical systems with emergent collective computational abilities", focusing on aspects that are not already covered in the main HopfieldNet formalization.

The paper introduces a model of neural networks with binary neurons and studies their collective computational properties, particularly as content-addressable memories. The key insights include:

- The phase space flow and stable states of the network
- The storage capacity and pattern retrieval capabilities
- The relationship between energy minimization and memory retrieval
- Tolerance to noise and component failures

Definition 14. A 'PhaseSpacePoint' represents a state in the phase space of the Hopfield system. In the paper, this corresponds to the instantaneous state of all neurons (p.2554): "A point in state space then represents the instantaneous condition of the system."

Definition 15. The 'localField' for neuron i in state s is the weighted sum of inputs from other neurons, minus the threshold. This corresponds to $\sum jTijVj - \theta_i$ in the paper.

Definition 16. The 'updateRule' defines the neural state update according to the paper's Equation 1: $Vi \to 1if \sum jTijVj > Ui\ Vi \to 0if \sum jTijVj < Ui$ In our formalization, we use -1 instead of 0 for the "not firing" state.

Definition 17. A 'PhaseSpaceFlow' describes how the system state evolves over time. It maps each point in phase space to its successor state after updating one neuron. From the paper (p.2554): "The equations of motion of the system describe a flow in state space."

Definition 18. A 'FixedPoint' of the phase space flow is a state that does not change under evolution. In the paper, these correspond to the locally stable states of the network (p.2554): "Various classes of flow patterns are possible, but the systems of use for memory particularly include those that flow toward locally stable points..."

Definition 19. A 'BasinOfAttraction' of a fixed point is the set of all states that converge to it. In the paper (p.2554): "Then, if the system is started sufficiently near any Xa, as at $X = Xa+\Delta$, it will proceed in time until $X \equiv Xa$."

Definition 20. The ‘EnergyLandscape‘ of a Hopfield network is the energy function defined over all possible states. In the paper, this is the function E defined in Equation 7: $E = -1/2 \sum \sum TijViVj$

Definition 21. The ‘EnergyChange‘ when updating neuron i is always non-positive, as proven in the paper with Equation 8.

This theorem formalizes a key result from the paper: the energy function always decreases (or remains constant) under asynchronous updates.

Definition 22. This theorem captures the convergence result from the paper: ”Every initial state flows to a limit point (if synchrony is not assumed).”

2.1 Memory Storage Algorithm

Definition 23. The ‘normalizedPattern‘ converts a neural state to a vector with -1/+1 values, matching the $(2V_i - 1)$ term from equation 2 in Hopfield’s paper.

Definition 24. The ‘hebbian‘ function computes the weight matrix according to Equation 2 of Hopfield’s paper: $T_{ij} = \sum_s (2V_i^s - 1)(2V_i^s - 1) with T_{ii} = 0$ Note that this is equivalent to the existing ‘Hebbian‘ definition in HopfieldNet.HN, but we make the connection to the paper explicit here.

Definition 25. The ‘pseudoOrthogonality‘ property from Hopfield’s paper (Equations 3-4) states: For random patterns, the dot product between different patterns is approximately 0, while the dot product of a pattern with itself is approximately N. This property is essential for understanding the storage capacity of Hopfield networks.

2.2 Connection to Physical Systems

Definition 26. The ‘spin_glass_analogy‘ formalizes the connection between Hopfield networks and physical spin glass systems, as discussed in the paper.

From the paper (p.2555): ”This case is isomorphic with an Ising model.”

Definition 27. The ‘energy_convergence‘ theorem formalizes the connection between energy minimization and the convergence to fixed points. This theorem establishes the connection between energy minimization and convergence to fixed points in Hopfield networks, as described in the paper From the paper (p.2555): ”State changes will continue until a least (local) E is reached.”

2.3 Content Addressable Memory

Definition 28. The ‘retrievalDistance‘ function measures how far from a pattern we can initialize the network and still have it converge to that pattern. From the paper (p.2556): ”For distance 5, the nearest state was reached more than 90% of the time. Beyond that distance, the probability fell off smoothly.

Definition 29. A Content Addressable Memory is a system that can retrieve a complete pattern from a partial or corrupted version.

This formalizes the central concept from the paper (p.2554): ”A general content-addressable memory would be capable of retrieving this entire memory item on the basis of sufficient partial information.

Definition 30. A generic function type representing how a metric (like completion probability or familiarity) decays or changes based on a distance-like value and network size.

Definition 31. A specific exponential decay model, often used to model probabilities or familiarity scores. This corresponds to 'exp(-value / (N/C))' where C is a constant (e.g., 10).

Definition 32. The 'AbstractCompletionProbability' measures the likelihood of correctly completing a pattern as a function of the Hamming distance 'd' from the stored pattern. This formalizes the empirical finding from the paper (p.2556): "For distance ≤ 5, the nearest state was reached more than 90% of the time. Beyond that distance, the probability fell off smoothly."

Definition 33. 'ErrorCorrection' quantifies the network's ability to correct errors in the input pattern. It's measured as the reduction in Hamming distance to the closest stored pattern after convergence.

Definition 34. The 'error_correction_guarantee' theorem establishes that Hopfield networks can correct a substantial fraction of errors in the input pattern.

This formalizes a key capability of content-addressable memories.

Definition 35. The 'BasinOfAttraction'' of a pattern is the set of all states that converge to it. This concept is central to understanding the storage and retrieval properties of Hopfield networks.

From the paper (p.2554): "Then, if the system is started sufficiently near any Xa, as at $X = Xa + \Delta$, it will proceed in time until $X \equiv Xa$.

Definition 36. The 'BasinVolume' is the "size" of the basin of attraction, measured as the fraction of the state space that converges to a given pattern.

This quantifies the robustness of memory retrieval.

Theorem 37. *The 'basin_volume_bound' theorem establishes that the basin volume decreases exponentially with the number of stored patterns. This formalizes how memory capacity affects retrieval robustness.*

Definition 38. 'AbstractFamiliarityMeasure' quantifies how familiar a given state 's' is to the network, based on its proximity (closest Hamming distance) to stored patterns, using a provided 'decay_func'.

The paper discusses (p.2557): "The state 00000... is always stable. For a threshold of 0, this stable state is much higher in energy than the stored memory states and very seldom occurs." A high familiarity measure (close to 1) indicates 's' is similar to a stored pattern. The 'ExponentialDecayMetric' can be used as 'decay_func'.

2.4 Memory Capacity

Definition 39. The 'StorageCapacity' of a Hopfield network is the maximum number of patterns that can be stored and reliably retrieved. The paper suggests this is around 0.15N. From the paper (p.2556): "About 0.15 N states can be simultaneously remembered before error in recall is severe."

Definition 40. The error function is defined as:

$$\text{erf}(x) = \frac{2}{\sqrt{\pi}} \int_0^x e^{-t^2}\, dt$$

This function is central in probability theory, especially for normal distributions.

Definition 41. The ‘PatternRetrievalError‘ function computes the probability of an error in pattern retrieval for a network storing m patterns. This increases as m approaches and exceeds 0.15N. This corresponds to the error probability P discussed in Equation 10 of the paper: $P = \int_{\sigma}^{\infty} \frac{1}{\sqrt{2\pi}} e^{-x^2/2}\, dx = \frac{1}{2}\left(1 - \operatorname{erf}\left(\frac{\sigma}{\sqrt{2}}\right)\right)$, where $\sigma = \frac{N}{2\sqrt{nN}}$ and n is the number of patterns.

Theorem 42. *The result from the paper that a Hopfield network can store approximately 0.15N patterns with high reliability, where N is the number of neurons. This theorem formalizes the key result about storage capacity from the paper, utilizing the Hebbian_stable theorem from the existing codebase.*

2.5 Fault Tolerance

Definition 43. The ‘DeleteNeuron‘ function simulates the failure of a neuron by removing its connections. This corresponds to setting weights to/from that neuron to zero. The paper discusses (p.2558): ”The collective properties are only weakly sensitive to details of the modeling or the failure of individual devices.”

Definition 44. The ‘FaultTolerance‘ of a Hopfield network is its ability to maintain function despite the failure of some components. The paper notes that these networks are inherently robust to component failures.

Definition 45. When m is at most a tenth of total neurons, each pattern is fixed point in the undamaged network

Lemma 46. *When deleting a single neuron from a network, the resulting weighted sum for a neuron u that's not the deleted neuron equals the original weighted sum minus the contribution from the deleted neuron.*

Definition 47. When deleting neurons from an empty list, the result is the original network

Definition 48. When deleting a list of neurons with a new neuron added at the front, the effect on the weighted sum equals the effect of deleting the first neuron and then deleting the rest of the list.

Definition 49. For a singleton list, the effect matches the single neuron deletion case

Definition 50. The effect of deleting a neuron from an already deleted network on a neuron u that is not in the deleted set.

Definition 51. Helper lemma: DeleteNeuron commutes with foldl of DeleteNeuron if the neuron is not in the list.

Definition 52. Helper lemma: Weights are preserved by foldl if indices are not in the list

Definition 53. Helper lemma: Deleting a list of neurons recursively subtracts their contributions. Requires that the list of neurons has no duplicates.

Definition 54. DeleteNeurons removes weights connected to deleted neurons

2.6 Lemmas for Bounding Field Reduction due to Neuron Deletion

Definition 55. Calculates the contribution to the deleted field sum from the target pattern ‘k‘ itself in the Hebbian weight definition.

Definition 56. Defines the cross-talk contribution to the deleted field sum. This term arises from the interaction of the target pattern ‘k‘ with other stored patterns `l ≠ k` over the set of deleted neurons.

Lemma 57. *Axiom stating the bound on the absolute value of the cross-talk term. This encapsulates the statistical argument from Hopfield's paper that for random-like patterns, the sum of interfering terms is bounded.*

Lemma 58. *Decomposes the total sum representing the field reduction into the contribution from the target pattern ‘k‘ and the cross-talk term from other patterns $l \neq k$.*

***Hopfield Assumption:** Assumes the standard Hebbian learning rule where $T_{ii} = 0$. The ‘Hebbian‘ definition in ‘HN.lean‘ implements this by subtracting ‘m • 1‘.*

Lemma 59. *Placeholder lemma for bounding the cross-talk term. Proving a tight bound likely requires assumptions beyond simple orthogonality, such as patterns being random and uncorrelated, analyzed in the limit of large N. The bound likely depends on ‘m‘ (number of patterns) and ‘N‘ (number of neurons). **Hopfield Assumption:** Implicitly assumes patterns behave statistically like random vectors.*

Lemma 60. *The field reduction from deleting neurons has a bounded effect. This version uses the decomposition and the (unproven) cross-talk bound.*

Lemma 61. *With constrained m and limited deleted neurons, the field remains strong enough*

Lemma 62. *For random orthogonal patterns, the cross-talk term has a bounded absolute value. This is a fundamental assumption from Hopfield's paper about how patterns interact.*

Lemma 63. *TO DO*

Lemma 64. *With bounded numbers of patterns and deleted neurons, the field remains strong enough to maintain the pattern stability, adjusted for N/5 bound.*

Lemma 65. *TO DO*

Lemma 66. *TO DO*

Lemma 67. *TO DO*

Definition 68. TO DO

Definition 69. When deleting neurons from a Finset, we can use Finset.toList to convert the Finset to a List. This matches the API needed by DeleteNeurons.

Definition 70. A Hopfield network can tolerate the failure of up to 10% of its neurons while maintaining all stored patterns as fixed points, provided:

1. The stored patterns are orthogonal
2. The number of patterns is at most 10% of the network size

Bibliography

[1] Stuart Geman and Donald Geman. Stochastic Relaxation, Gibbs Distributions, and the Bayesian Restoration of Images. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, PAMI-6(6):721–741, 1984.

[2] W. K. Hastings. Monte Carlo Sampling Methods Using Markov Chains and Their Applications. *Biometrika*, 57(1):97–109, 04 1970.

[3] John J Hopfield. Neural Networks and Physical Systems with Emergent Collective Computational Abilities. *Proceedings of the national academy of sciences*, 79(8):2554–2558, 1982.

[4] S. Kirkpatrick, C. D. Gelatt, and M. P. Vecchi. Optimization by Simulated Annealing. *Science*, 220(4598):671–680, 1983.

[5] Rudolf Kruse, Christian Borgelt, Christian Braune, Sanaz Mostaghim, Matthias Steinbrecher, Frank Klawonn, and Christian Moewes. *Computational Intelligence*. Springer, 2011.

[6] David A Levin and Yuval Peres. *Markov Chains and Mixing Times*, volume 107. American Mathematical Soc., 2017.

[7] Nicholas Metropolis, Arianna W. Rosenbluth, Marshall N. Rosenbluth, Augusta H. Teller, and Edward Teller. Equation of State Calculations by Fast Computing Machines. *The Journal of Chemical Physics*, 21(6):1087–1092, 06 1953.