Neural Networks in Lean4

mkaratarakis

May 14, 2025

Chapter 1

NN

We strictly follow Chapter 4 of [4] 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, [4]))**.** *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, [4]))**.** *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. [3], 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. [6] 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 [5] :

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

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).

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

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

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

[6] 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.