# Statistical Mechanics for Neural Networks and Artificial Intelligence

Chapter 7:
Introduction to Energy-Based Neural Networks:
The Hopfield Network and the (Restricted) Boltzmann Machine
DRAFT / EXCERPT

Alianna J. Maren
Northwestern University School of Professional Studies
Master of Science in Data Studies

Draft: 2019-05-09

## 7.1 Introduction to Energy-Based Neural Networks

One of the most important aspects of advanced machine learning / deep learning studies is the shift into a physics-based approach; specifically into *statistical mechanics*. Statistical mechanics, combined with Bayesian probability theory and also with neural network methods, contribute to the central themes of machine learning, as illustrated in the following Fig. 7.1.

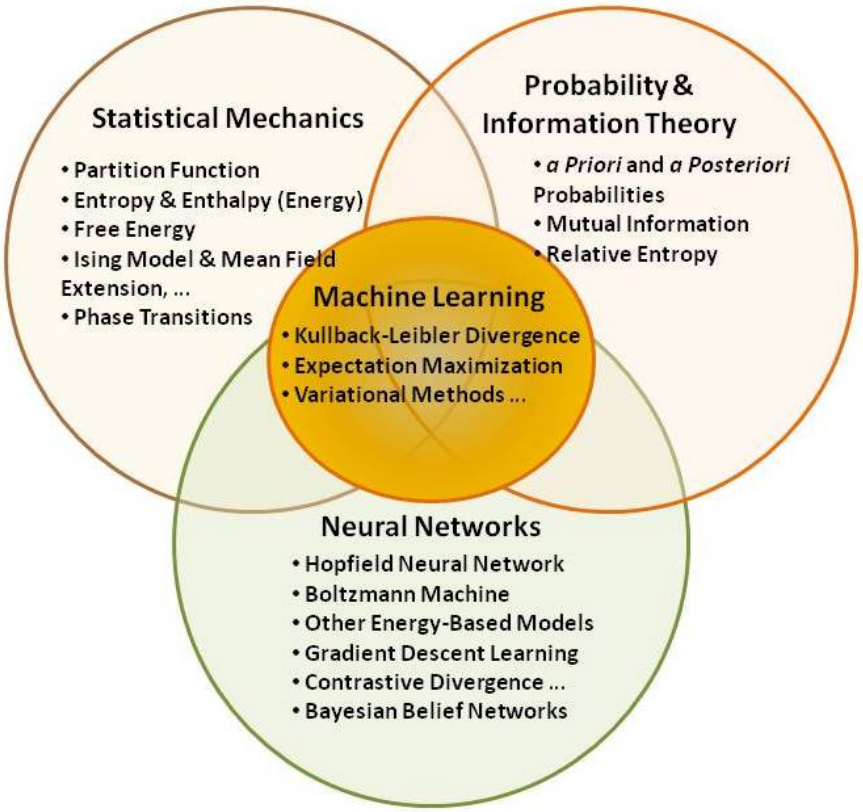


Figure 7.1: Machine learning algorithms are at the confluence of statistical mechanics, probability and information theory, and neural networks.

One of the most interesting, distinctive, and even arcane aspects about advanced neural networks and machine learning algorithms is that they use two very different forms of probability-thinking. These two different methods,

coming from statstical mechanics and Bayesian probabilities (respectively) are hugely different ways of thinking about the likelihood of whether or not something will happen.

Statistical mechanics, a realm of theoretical physics, is used in neural networks largely as an allegory; as a model created in one field that has been (very usefully) applied to another. It's almost like using physics as story-telling. The notion that these methods could be successfully used is so extreme that its almost shocking that these methods could find a new home in neural networks and deep learning.

The notions of statistical mechanics are central to the learning methods for restricted Boltzmann machines (RBMs). A restricted Boltzmann machine learns using a very different underlying approach than that used by stochastic gradient descent implementations (e.g., backpropagation). This means that RBMs can have multi-layered architectures and learn to distinguish between more complex patterns, overcoming the limitations of simple Multilayer Perceptrons (MLPs), as we previously discussed.

Statistical mechanics deals with the probabilities of occurrence of small units that can be distinguished from each other only by their energy states. In contrast, Bayesian probabilities provide a remarkably different way of thinking about the probabilities with which things can happen. Together, these two probability-oriented methods provide the foundations for advanced machine learning methods.

Now that we've identified the importance of both statistical mechanics and Bayesian methods, we will restrict our attention (for this chapter and the immediately-following ones) to statistical mechanics and its foundational relationship with neural networks. We'll pick up on the full confluence of statistical mechanics and Bayesian methods later, when we address more advanced topics.

The first time that the role of statistical mechanics became well-known in neural networks was when John Hopfield presented his work in 1982 [1]. His work drew on some similar lines of thinking developed by Little and colleagues in 1974 [2].

This chapter presents some of the key concepts in statistical mechanics; sufficient to understand the subject of some classic papers: Hopfield's original work (introducing what became known as the Hopfield network), and a few key works on the Boltzmann machine, developed by Geoffrey Hinton and colleagues.

## 7.2 Introduction to the Hopfield Neural Network and the Boltzmann Machine

The Hopfield neural network and its immediates successor, the Boltzmann machine (in both original and restricted forms) are instances of energy-based neural networks. They each achieve their desired connection weight values by minimizing an *energy equation*.

At first glance, the two networks do not seem to be structurally the same. However, they have a great deal in common, as we'll see shortly.

The following Fig. 7.2 illustrates both the Hopfield and the Boltzmann machine neural networks, so that we can easily compare the structures. The Hopfield neural network is shown on the left-hand-side, and the Boltzmann machine (in two different configurations, but still the same network) is shown in the center and right-hand-side graphs.

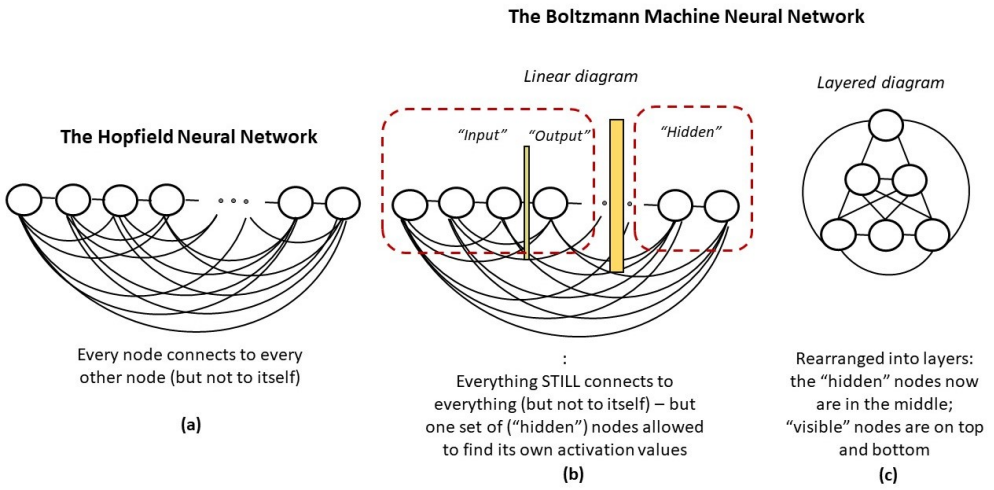


Figure 7.2: Illustration of the Hopfield and Boltzmann machine neural network archtiectures; the Boltzmann machine is essentially a Hopfileld neural network with certain connections removed.

To understand the restricted Boltzmann machine (RBM), which is central to current deep learning theory, it helps to first understand the simple (non-restricted) Boltzmann machine. And to understand the simple Boltzmann machine, it helps us to first understand the Hopfield neural network. Thus, we'll briefly examine the Hopfield neural network.

The Hopfield neural network, as it came to be known, was immediately interesting to the newly-forming neural networks community. Hopfield networks can be used for different tasks; one of the most interesting was as an optimization method. However, its first and most fundamental application was as an *autoencoder*. An autoencoder is a device that can remember previously-stored patterns, if triggered to their recall by presentation of a partial or noisy version of an original pattern.

Despite the Hopfield network's interesting ability to reconstruct stored patterns, it had a severe memory limitation. It could only learn a number of patterns that was about 15% of the total number of nodes in the system. So if, for example, a Hopfield network was created with 20 nodes (or neurons, or units), then it could learn and retrieve only three distinct patterns. This memory restriction caused many people to lose interest in this network.

Geoffrey Hinton, who (like John Hopfield) was also a physicist, came up with a novel insight into how the structure of the Hopfield neural network could be rearranged. This led to creation of the Boltzmann machine, and then the restricted Boltzmann machine (RBM), which has been the cornerstone of of deep learning.

So, in order to understand the restricted Boltzmann machine, we're going to start at the beginning - with the equations and structure of the Hopfield neural network. Once we understand that, it's a straightforward, natural, and intuitive step to understand Boltzmann machines - both in their original and restricted forms. This then paves the way for us to understand and use the wide range of methods involving energy-based systems in neural networks and machine learning.

## 7.3 The Hopfield Neural Network - Energy Equation and Structure

The Hopfield neural network, most often simply called the *Hopfield network*, is a beautiful instance of how ***form and function*** perfectly reflect each other. The *function* of this network is expressed in its energy equation. This energy equation is perfectly mirrored in its *form*, or the structure of this network.

We'll begin by looking at the energy equation, as presented by John Hopfield in his classic 1982 paper, shown in the following Fig. 7.3.

Notice that there is just one energy equation here, given as *Eqn. [7]* in the Hopfield 1982 paper. The second equation, *Eqn. [8]*, is an energy update equation; it describes how the energy is changed as the weight update rule is applied. Thus, our focus is on that first *Eqn. [7]*, as illustrated in Fig. 7.3.

**Energy Equation Used by Hopfield (1982)**

**Studies of the collective behaviors of the model**

The model has stable limit points. Consider the special case $T_{ij} = T_{ji}$, and define

$$E = -\frac{1}{2}\sum_{i \neq j}\sum T_{ij}V_iV_j \ . \qquad [7]$$

$\Delta E$ due to $\Delta V_i$ is given by

$$\Delta E = -\Delta V_i \sum_{j \neq i'} T_{ij}V_j \ . \qquad [8]$$

Thus, the algorithm for altering $V_i$ causes $E$ to be a monotonically decreasing function. State changes will continue until a least (local) $E$ is reached. This case is isomorphic with an Ising model. $T_{ij}$ provides the role of the exchange coupling, and there is also an external local field at each site. When $T_{ij}$ is symmetric but has a random character (the spin glass) there are known to be many (locally) stable states (29).

Figure 7.3: Extract from J. Hopfield (1982, April). Neural networks and physical systems with emergent collective computational abilities, *Proc. Natl. Acad. Sci. U.S.A.*, *79*, pp. 2554-2558.

The first of the two equations shown in Fig. 7.3 defines the overall energy of the system, and the second shows what happens to the overall energy when we flip any given node from *1* to *0*, or vice versa. We will focus on the first equation here, and defer the second equation to a later chapter.

Before we interpret the first equation (*Eqn. [7]*) in Hopfield's paper, we're going to briefly note where he says "This case is isomorphic with an Ising model ..." This is a reference to statistical mechanics, which we'll address starting with the next chapter. An *Ising model* is a classic model in statistical mechanics, and is very relevant to our work in energy-based neural networks. We could say that the entirety of energy-based neural networks is built on a foundation that uses the Ising model as a starting point. We'll follow this

line of thought in the next few chapters. For now, we focus our attention on the first equation, *Eqn. [7]* in Hopfield's 1982 paper, as shown in Fig. 7.3.

For readability, the equation in Fig. 7.3 is reproduced here as Eqn. 7.1.

$$E = -\frac{1}{2}\sum_{i \neq j}\sum T_{i,j}V_iV_j. \tag{7.1}$$

Our first step in understanding this equation is to interpret the terms $V_i$, $V_j$, and $T_{ij}$.We refer to Hopfield's original 1982 paper, where he states:

"The processing devices will be called neurons. Each neuron $i$ has two states like those of McCulloch and Pitts (*Author's note:* the reference citation is updated here for the reader's benefit, see [3]): $V_i = 0$ ("not firing") and $V_i = 1$ ("firing at maximum rate"). When neuron $i$ has a connection made to it from $j$, the strength of connection is defined as $T_{ij}$. (Nonconnected neurons have $T_{ij} \equiv 0$.)"

Let's examine both the energy equation for the Hopfield neural network, and the illustration of its structure. For ease in visualizing the Hopfield neural network, the depiction of it from from Fig. 7.2 is presented here at larger scale, as Fig. 7.4.

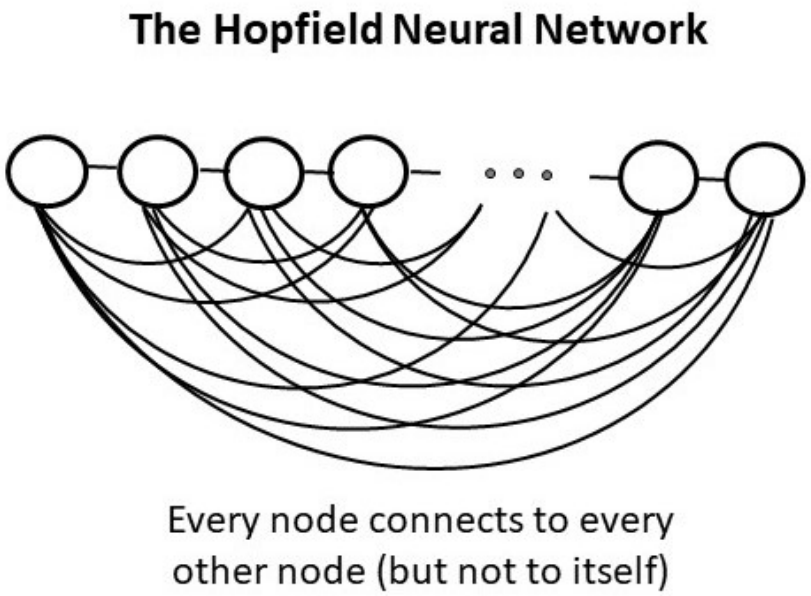


Figure 7.4: The Hopfield neural network archtiecture; expanded from Fig. 7.2.

From Fig. 7.4, we see that every node is connected to each other node, but not to itself.

In the Hopfield neural network, we have connections between each of the different nodes. The notion of having some nodes be "visible" and other nodes "hidden" is not present in the Hopfield neural network; the notion of having "hidden" nodes was a innovation introduced a few years later by Geoffrey Hinton, and was essential to the notion of the Boltzmann machine. Essentially, all nodes in a Hopfield network are "visible."

The energy equation for the Hopfield neural network addresses all possible combinations of nodes in the network; this is why we see the double sumamtion sign in Eqn. 7.1. This equation has the values for two different nodes in it; we see both $V_i$ and $V_j$. We know that there are no connections from any given node back to itself, because that would be a connection between $V_i$ and $V_i$; for example node *1* connecting back to *1*. The equation explicitly states that we don't have these connections, because we see $i \neq j$ as the subscript under the two summation signs. As we refer back to Fig. 7.4, we see this is the case; each node connects to each other node, but there are no "loops" connecting a node back to itself.

There's just two more things that we can glean about the structure and operation of the Hopfield neural network from its energy equation. First, we see that there is a multiplying factor of $-1/2$ in front of the summations. Also, from Hopfield's original work, from which we saw an excerpt in Fig. 7.3, we read "Consider the special case $T_{i,j} = T_{j,i}$ ..." This means that the connection strength between any two nodes is the same, whether we read it from the first node to the second or vice versa. For example, the connection strength going from node *2* to node *3* is the same as the connection strength going from node *3* back to node *2*.

The way that the equation is set up, we count each direction of connections. For example, we separately count the interactions between node *2* to node *3* ($T_{2,3}V_2V_3$) and between node *3* to node *2* ($T_{3,2}V_3V_2$). Because we've essentially counted the same thing twice, we need to divide by two; this is why we have the normalizing factor of $1/2$ in front of the double summation.

The negative sign is introduced so that we can have positive values for the connection parameter $T_{i,j}$. Remember, our algorithms are going to seek a minimum in the energy equation. This is similar to how, when we did stochastic gradient descent using the backpropagation algorithm, we were seeking a minimum value. Our values for $T_{i,j}$ are not constrained to be positive, but putting the negative sign in front of the double summation

energy term allows them to be positive more often than not, and the system will still (likely) come to a minimum state as we apply our energy-minimization algorithm.

Summing up what we've learned so far, we note that the Hopfield neural network, which is the predecessor neural network for the Boltzmann machine, has the following properties:

1. The network is constructed from a set of nodes, all of which are "visible,"
2. Each node connects to each other node, but not back to itself,
3. The nodes in this system are *binary*; meaning that they can each be in one of two states; "on" or "off;" for both the Hopfield and the Boltzmann machine networks, these values are $(1, 0)$,
4. The connections between nodes in this system are *symmetric*, so that $T_{i,j} = T_{j,i}$, and
5. The stable state of this network is governed by an *energy equation*, and the training algorithm adapts the connection parameter $T_{i,j}$ between each pair of nodes ($V_i$ and $V_j$) in order to achieve a total minimum value.

We're not going to address the training algorithm right now (the energy-minimizing algorithm), as the purpose of this section was just to make a connection between the formalism of the energy equation and the structure of the Hopfield neural network. We've seen that, for the Hopfield neural network, ***form equals function***, in that the set of fully-connected nodes (without self-connection) is illustrated in both Fig. 7.4 and Eqn. 7.1.

The important thing that we've done here has been to lay a foundation, because our next step is to similarly understand the correspondence between the energy equation for the Boltzmann machine and the structure and nature of the Boltzmann machine neural network. That is the goal of the next section.

# Bibliography

[1] J. Hopfield, "Neural networks and physical systems with emergent collective computational abilities," *Proc. Natl. Acad. Sci. U.S.A.*, vol. 79, p. 2554  2558, April 1982.

[2] W. Little, "The existence of persistent states in the brain," *Math Biosci.*, vol. 19, pp. 101–120, February 1974.

[3] W. McCulloch and W. Pitts, "A logical calculus of the ideas immanent in nervous activity," *Bull. Math. Biophys.*, vol. 5, pp. 115–133, 1943.