EasyChair Preprint

№ 15830

On the Dynamics of Real Valued Hopfield Type Neural Networks Based on Ceiling Neurons

Rama Murthy Garimella, Fidelis Zanetti de Castro, Aman Singh, Jyothi Prasanna, Maha Lakshmi Bairaju, Manasa Jagannadhan and Vidya Sree Vankam

EasyChair preprints are intended for rapid dissemination of research results and are integrated with the rest of EasyChair.

February 14, 2025

On the Dynamics of Real-Valued Hopfield-Type Neural Networks based on Ceiling Neurons

Rama Murthy Garimella[1] Fidelis Zanetti de Castro[2] Aman Singh[3]
Jyothi Prasanna[4] Mahalakshmi Bairaju[5] Manasa Jagannadan[6]
Vidya Sree Vankam[7]
[1] Ecole Centrale School of Engineering, Mahindra University
Hyderabad, Telangana, India.
[2]Federal Institute of Education, Science and Technology of Espírito Santo at Serra
Serra, Espírito Santo, Brazil.
[3] International Institute of Information Technology
Hyderabad, Telangana, India.
[4] Rajiv Gandhi University of Knowledge Technologies, RK Valley
Idupulapaya, Kadapa, Andhra Pradesh, India.
[5,6,7] Rajiv Gandhi University of Knowledge Technologies, RK Valley
Vempalli (M), Kadapa, Andhra Pradesh, India.

Abstract

In this chapter, we introduce Hopfield-type neural networks based on the ceiling neuron model. Inspired by the binary Hopfield neural network and by the definition of ceiling neuron, we propose an extension of the Hopfield neural network from a binary set of possible states of a real-valued neuron to a multistate set. We investigate the dynamics of the proposed model in asynchronous and synchronous update modes. Computational experiments illustrate that the Hopfield neural network based on the ceiling neuron in an asynchronous update mode always settles down to a stationary state under the usual conditions on synaptic weights, that is, the synaptic weight matrix is symmetric with non-negative diagonal elements. In turn, the Hopfield neural network with ceiling neurons in the synchronous update mode can generate sequences with limit cycles, similar to the classic Hopfield neural network.

1 Introduction

The Hopfield neural network (HNN) is one of the most important recurrent neural networks introduced in the literature. It was conceived by the American physicist W. A. Little in 1974 [22], and was popularized by the American physicist, biologist, and neurologist John Joseph Hopfield, in 1982 [13]. Hopfield investigated content-addressable memories aimed to store and recall binary vectors using the Hebbian rule [11, 13]. Besides implementing content-addressable memories, HNNs have been applied in control [8, 29], classification [26, 35], computer vision and image processing [21, 34], and optimization [12, 20, 28].

Due to the limited representational capability of McCulloch-Pitts neurons, extensions of HNNs have been proposed aimed to process information of different natures, especially information involving multidimensional data. In this sense, researchers have proposed, for example, complex-valued Hopfield neural networks, which allow to process 2-dimensional data as single entities using complex-valued neurons [1, 2, 6, 9, 17, 24, 25, 30]. Furthermore, to process other kinds of multidimensional data as single entities, researchers have extended the HNNs from the field of real numbers to hypercomplex fields. Examples include quaternion-valued HNNs [15, 16, 31, 32, 33], octonion-valued HNNs [5, 18, 19], and other hypercomplex-valued HNNs based on Cayley-Dickson or Clifford algebras, for instance [7].

Nevertheless, in this chapter we focus on another alternative, internally to the field of real numbers, to introduce a kind of neuron allowing to capture information of a multidimensional nature. The so-called ceiling neuron, introduced in [10], acts to increase the cardinality of the set of possible states of a real-valued neuron through the introduction of a finite set of thresholds to each neuron of an HNN. As we will see, this approach allows associating a multistate set of cardinality equal to $K + 1$ to each real-valued neuron of an HNN, where K is a positive integer called resolution factor.

This chapter is organized as follows: Section 2 introduces the ceiling neuron model and highlights its main operational differences compared to the the classic McCulloch-Pitts neuron model. Sections 3 and 4 discutes aspects inherent to

HNNs based on ceiling neurons and McCulloch-Pitts neurons, with emphasis on their dynamics. In order to illustrate the dynamics of HNNs, Section 5 presents some computational experiments. We finish the chapter with some concluding remarks at Section 6.

2 The McCulloch-Pitts and the Ceiling Neuron Models

The first computational model of a neuron was proposed by the neuroscientist Warren McCulloch and the logician Walter Pitts in 1943 [23]. Let us review this model. Consider a neuron called neuron i. Let real numbers $x_1, x_2, \ldots, x_N$ be the inputs of the neuron i, which, from a biological point of view, represent excitatory or inhibitory postsynaptic potentials at neural dendrites $1, 2, \ldots, N$. Associated with each of them we consider the real-valued synaptic weights $w_{i1}, w_{i2}, \ldots, w_{iN}$, respectively, which represent the intensities of the postsynaptic potentials at dendrites. The weighted sum $x_1 w_{i1} + x_2 w_{i2} + \ldots + x_N w_{iN}$ is called *action potential of the neuron* i, and is denoted in this chapter by v_i. Formally, we write

$$v_i = \sum_{j=1}^{N} w_{ij} x_j. \tag{1}$$

The McCulloch–Pitts neuron allows binary activations, i.e., it either "fires" with an activation equal to 1 or "does not fire" with an activation equal to 0. Specifically, if the action potential v_i is greater than a fixed threshold real value θ_i, the neuron i "fires". Otherwise, the neuron i "does not fire". This thresholding process can be mathematically represented by using the Heaviside-type step function $\mathtt{h} : \mathbb{R} \to \{0, 1\}$ defined by

$$\mathtt{h}(x) = \begin{cases} 1, & x > 0 \\ 0, & \text{otherwise} \end{cases}. \tag{2}$$

We obtain the output $y_i \in \{0, 1\}$ of the neuron i by means of the equation

$$y_i = \mathtt{h}(v_i - \theta_i). \tag{3}$$

Figure 1 illustrates the the operating mode of a McCulloch-Pitts neuron.

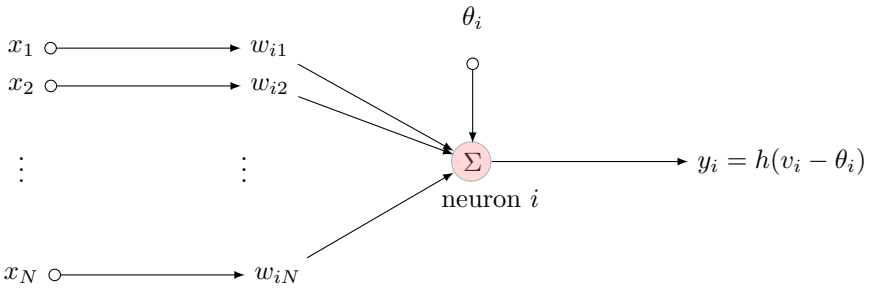

Figure 1: The McCulloch-Pitts neuron model.

Let us present the ceiling neuron [10]. Unlike McCulloch-Pitts neuron model, in the ceiling neuron model each neuron i has a set $\{\theta_{1i}, \theta_{2i}, \ldots, \theta_{Ki}\}$ of K distinct real-valued thresholds associated with it. Consequently, the set of possible values of a neuron is expanded from the binary set $\{0, 1\}$ to the multistate set $S = \{0, 1, \ldots, K\}$, where $K \geq 1$ is some positive integer. In the context of this chapter, the number K will be called *resolution factor*, due to its structural conformity with the resolution factors commonly used in complex-valued HNNs [1, 6, 9, 17, 24, 30].

Similar to the McCulloch-Pitts neuron model, consider a neuron i with real-valued inputs $x_1, x_2, \ldots, x_N$, and the real-valued weights $w_{i1}, w_{i2}, \ldots, w_{iN}$, respectively. Besides, let us define the set of the real-valued thresholds associated with the neuron i by $\{\theta_{1i}, \theta_{2i}, \ldots, \theta_{Ki}\}$, with $\theta_{\mu i} \neq \theta_{\eta i}$ for all $\mu \neq \eta$.

The output y_i of the ceiling neuron i is obtained as follows. For each threshold θ_{ji}, with $j = 1, \ldots, K$, we apply the Heaviside-type step function given by (2) on the difference $v_i - \theta_{ji}$, and add all the results obtained. Formally, the output of the ceiling neuron i is given by equation

$$y_i = \sum_{j=1}^{K} \mathtt{h}(v_i - \theta_{ji}), \tag{4}$$

where v_i is the action potential of the neuron i given by (6). It is easy to see that $y_i \in S = \{0, 1, \ldots, K\}$.

Figure 2 illustrates the operating mode of a ceiling neuron.

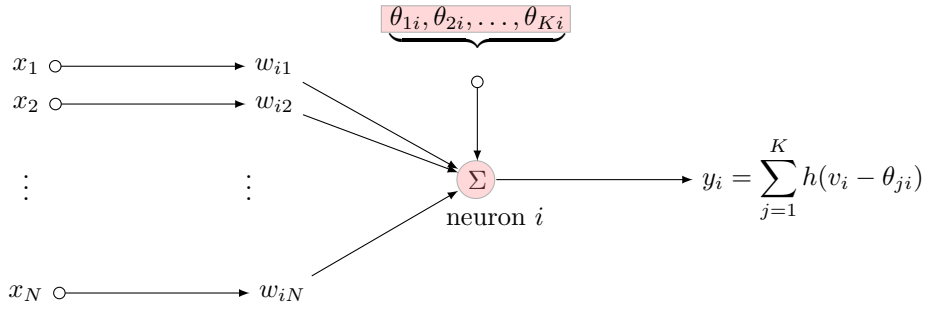

Figure 2: The ceiling neuron model.

From a practical point of view, the output y_i of a ceiling neuron can be interpreted as the number of thresholds lower than the action potential v_i. In this sense, it is important to note that a single ceiling neuron divides the space of the values of a neuron into $K + 1$ subspaces of decision. In contrast, to mimic this same action, K neurons of McCulloch-Pitts would be needed.

HNNs based on ceiling neurons can be applied in reconstruction of grayscale or color images, image filtering, multiclass classification, semantic segmentation, or optimization, for instance, in a different way compared to a classic HNN approach, or a hypercomplex-valued HNN approach.

3 Binary Hopfield Neural Networks

The binary HNN is one of the most important recurrent neural networks from the literature. It is composed by a totally connected single-layer with neurons of McCulloch-Pitts [13]. Consider a HNN with N neurons, let w_{ij} be the jth synaptic weight of the ith neuron, and let θ_i be the threshold of the ith neuron. The state of the ith neuron at time t is denoted by $x_i(t) \in \{0, 1\}$, for $i = 1, \ldots, N$. The output $x_i(t + 1)$ of the vector $\mathbf{x}(t + 1)$ can be obtained according to the equation

$$x_i(t+1) = \begin{cases} \mathtt{h}(v_i(t) - \theta_i), & v_i(t) - \theta_i \neq 0 \\ x_i(t), & \text{otherwise,} \end{cases} \tag{5}$$

where

$$v_i(t) = \sum_{j=1}^{N} w_{ij} x_j(t) \tag{6}$$

is the *action potential of ith neuron at time t*, and $\mathtt{h} : \mathbb{R} \to \{0, 1\}$ is the Heaviside-type step function given by (2).

Note that Heaviside-type step function $\mathtt{h} : \mathbb{R} \to \{0, 1\}$ given by (2) is intrinsically related to the classic real sign function $\mathtt{sgn} : \mathbb{R} \to \{-1, 1\}$ by means of the invertible equation

$$\mathtt{sgn}(x) = 2\mathtt{h}(x) - 1. \tag{7}$$

Equation (7) provides an easy way to convert an HNN with neuron values in $\{0, 1\}$ (binary) to other HNN with values in $\{-1, 1\}$ (bipolar), and vice versa. It is important to note that the dynamics of an HNN is invariant for this conversion.

The neurons of an HNN can be updated asynchronously or synchronously. If the neurons are updated asynchronously, that is, a single neuron is updated at time t, HNNs always settles at an equilibrium state if the synaptic weight satisfies the usual conditions $w_{ij} = w_{ji}$ and $w_{ii} \geq 0$, that is, the synaptic weight matrix is symmetric with non-negative diagonal elements, for all $i, j = 1, \ldots, N$, [4, 14]. This means that the sequences $\{\mathbf{x}(t)\}_{t \geq 0}$ generated by the HNN are always convergent, given any initial state $\mathbf{x}(0) \in \{0, 1\}^N$. In an asynchronous update mode, the neurons can be updated randomly, or a pre-defined order can be imposed on the updates. In the computational experiments, we chose the following update order: first update the first neuron, then the second neuron, and so on.

In turn, in the synchronous update mode all neurons of the HNN are updated at the same time t. Some researchers see this mode of updating as less plausible from a biological point of view [27]. They base their assertions on the absence of scientific evidence about the existence of a global clock that influences or determines natural neural networks. Regardless of the biological plausibility, this mode of operation of an HNN occurs mathematically as follows.

Consider an HNN with N neurons, let $\mathbf{W}$ be a matrix such that w_{ij} be denotes the jth synaptic weight of the ith neuron, and θ a N-dimensional column vector with ith component θ_i equal to the threshold of the ith neuron. The vector $\mathbf{x}(t+1)$

is obtained from $\mathbf{W}$, θ, and $\mathbf{x}(t)$ according to the equation

$$x_i(t+1) = \begin{cases} \mathtt{h}\left((\mathbf{W}\mathbf{x}(t))_i - \theta_i\right), & (\mathbf{W}\mathbf{x}(t))_i - \theta_i \neq 0 \\ x_i(t), & \text{otherwise,} \end{cases} \tag{8}$$

where $\mathtt{h}$ is the Heaviside function given by (2).

By using the synchronous update mode, and imposing the same conditions on W, HNNs can produce limit cycles of length 2, that is, periodic sequences of period 2 can be generated by the equation (8). Further details on the dynamics of an HNN using synchronous update can be found in [3].

Remark 1 *It is important to highlight the difference between the equations* (5) *and* (8)*. In equation* (5)*, only the action potential of the ith neuron of $\mathbf{x}(t)$ is calculated at each time t by using the inner product between the ith row of $\mathbf{W}$ and the vector $\mathbf{x}(t)$. Then, the activation function $\mathtt{h}$ together with the threshold θ_i are used to update the vector $\mathbf{x}(t)$. The values of i and t are incremented, and the the process is repeated until some stopping criteria is satisfied. In turn, in the equation* (8) *all action potentials of the neurons $1, 2, \ldots, N$ are calculated at the same time t using the usual matrix product between $\mathbf{W}$ and $\mathbf{x}(t)$. Then, the activation function h together with the components of the vector θ are used to update all neurons. After, the value of t is incremented, and the process is repeated until some stopping criteria is satisfied. Also note that, in general, $(\mathbf{W}\mathbf{x}(t))_i \neq v_i(t)$.*

4 Real-Valued Hopfield Neural Networks based on Ceiling Neurons

In this section, we present an extension of the HNN model replacing the McCulloch-Pitts neuron with the ceiling neuron. Consider a HNN with N neurons, and let $\mathbf{W}$ be the $N \times N$ matrix where w_{ij} denotes the jth synaptic weight of the ith neuron. Let us define the matrix with all thresholds of the HNN by $\mathbf{\Theta} \in \mathbb{R}^{K \times N}$. Specifically, the ith column of $\mathbf{\Theta}$ contains the K distinct thresholds $\theta_{1i}, \theta_{2i}, \ldots, \theta_{Ki}$ of the ith neuron, where $i \in \{1, 2, \ldots, N\}$. The state of the ith neuron at time t is denoted by $x_i(t) \in S = \{0, 1, \ldots, K\}$.

In an asynchronous update mode, each ceiling neuron i of the vector $\mathbf{x}(t+1)$ is obtained according to the equation

$$x_i(t+1) = \begin{cases} \displaystyle\sum_{j=1}^{K} \mathtt{h}(v_i - \theta_{ji}), & v_i(t) - \theta_{ji} \neq 0 \ , \forall \ j \in \{1, 2, \ldots, K\} \\ x_i(t), & \text{otherwise,} \end{cases} \tag{9}$$

where $v_i(t)$ is given by (6), and $\mathtt{h}$ is the Heaviside-type step function given by (2).

Now, consider the HNN with N ceiling neurons in the synchronous update mode. Let $\mathbf{W} \in \mathbb{R}^{N \times N}$ be a matrix of synaptic weights, and $\mathbf{\Theta} \in \mathbb{R}^{K \times N}$ the matrix of thresholds of the HNN. In this case, the vector $\mathbf{x}(t+1)$ is obtained from $\mathbf{W}$, $\mathbf{\Theta}$, and $\mathbf{x}(t)$ according to the equation

$$x_i(t+1) = \begin{cases} \displaystyle\sum_{j=1}^{K} \mathtt{h}((\mathbf{W}\mathbf{x}(t))_i - \theta_{ji}), & (\mathbf{W}\mathbf{x}(t))_i - \theta_{ji} \neq 0 \ , \forall \ j \in \{1, \ldots, K\} \\ x_i(t), & \text{otherwise,} \end{cases} \tag{10}$$

where $\mathtt{h}$ is the Heaviside-type step function given by (2).

Remark 2 *Broadly speaking, the practical difference between the HNN with ceiling neurons given by the equation* (9)*, and the HNN given by* (10) *resides in the way in which the action potentials are obtained, accordingly to the remark 1, similarly to the classic HNNs.*

Theorem 1 presents sufficient conditions for an HNN with ceiling neurons always settles down at a stationary state.

Theorem 1 *Consider an HNN with N ceiling neurons and a resolution factor $K \geq 1$. Let $\mathbf{\Theta} \in \mathbb{R}^{K \times N}$ be the matrix of thresholds of the HNN, and $S = \{0, 1, \ldots, K\}$ the set of all possible values for a ceiling neuron. If the synaptic weights satisfy the conditions $w_{ji} = w_{ij}$, and $w_{ii} \geq 0$ for all $i, j \in \{1, \ldots, N\}$, then the sequences $\{\mathbf{x}(t)\}_{t \geq 0}$ generated by the evolution equation* (9) *are always convergent given any initial state $\mathbf{x}(0) \in S^N$.*

Remark 3 *Theorem 1 is true for $K = 1$ [12]. In this case, an HNN based on N ceiling neurons is equivalent, apart from an isomorphism, to an HNN based on the N McCulloch-Pitts neurons. The proof of the theorem follows, for example, from the introduction of the Lyapunov functional $E : \{0,1\}^N \to \mathbb{R}$ given by $E(\mathbf{x}(t)) = -\frac{1}{2}\mathbf{x}(t)^T\mathbf{W}\mathbf{x}(t) + \mathbf{\Theta}\mathbf{x}(t)$ for the HNN [12].*

To prove Theorem 1 there is no need to build explicitly a Lyapunov functional for the HNN. Just note that the each vector $\mathbf{x}(t) \in S = \{0, 1, \ldots, K\}^N$ can be decomposed as a sum of vectors in $\{0,1\}^N$. Precisely, let $M = \max\{\mathbf{x}(t)\} \leq K$ be the maximum value between all components of the vector $\mathbf{x}(t)$. Note that there exists a sequence of vectors $\mathbf{z}_\mu \in \{0,1\}^N$ such that $\sum_{\mu=1}^{M} \mathbf{z}_\mu = \mathbf{x}(t)$. Then, since HNNs with resolution factor $K = 1$ satisfy Theorem 1, HNNs with arbitrary K also satisfy. In other words, any vector obtained as an output from an HNN with $K > 1$ thresholds can be decomposed as the sum of a certain amount of vectors obtained as outputs from an HNN with $K = 1$ threshold.

Remark 4 *If the synaptic weights satisfy the conditions $w_{ji} = w_{ij}$ and $w_{ii} \geq 0$ for all $i, j \in \{1, \ldots, N\}$, then the sequences $\{\mathbf{x}(t)\}_{t\geq 0}$ generated by the evolution equation* (10) ***are not always convergent*** *given any initial state $\mathbf{x}(0) \in S^N$.*

The experiments from Section 5 illustrate what has been addressed so far.

5 Computational Experiments

Experiment 1 *The objective of this experiment is to compare, under Theorem 1 conditions, the dynamics of the asynchronous and synchronous HNN models with celing neurons.*

For this, consider the symmetric synaptic weights matrix $\mathbf{W} = \begin{bmatrix} 0 & 3 & -2 \\ 3 & 0 & -4 \\ -2 & -4 & 0 \end{bmatrix}$, *the thresholds matrix* $\mathbf{\Theta} = \begin{bmatrix} 0.5 & 0.1 & 0.1 \\ 0.3 & 0.2 & 0.9 \end{bmatrix}$, *and the initial state* $\mathbf{x}(0) = \begin{bmatrix} 0 \\ 1 \\ 1 \end{bmatrix}$ *for the HNN.*

1a) *In an asynchronous update, that is, by using the evolution equation* (8)*, we obtain the following dynamics for the HNN.*

First iteration:

$$v_1(0) = 0 \cdot 0 + 3 \cdot 1 + (-2) \cdot 1 = 1.$$

Then, $x_1(1) = \mathtt{h}(1 - 0.5) + \mathtt{h}(1 - 0.3) = 1 + 1 = 2$*. Thus,*

$$\mathbf{x}(1) = \begin{bmatrix} 2 \\ 1 \\ 1 \end{bmatrix}.$$

Second iteration:

$$v_2(1) = 3 \cdot 2 + 0 \cdot 1 - 4 \cdot 1 = 2.$$

Then, $x_2(2) = \mathtt{h}(2 - 0.1) + \mathtt{h}(2 - 0.2) = 1 + 1 = 2$*. Consequently,*

$$\mathbf{x}(2) = \begin{bmatrix} 2 \\ 2 \\ 1 \end{bmatrix}.$$

Third iteration:

$$v_3(2) = -2 \cdot 2 - 4 \cdot 2 + 0 \cdot 1 = -12.$$

Then, $x_3(3) = \mathtt{h}(-12 - 0.1) + \mathtt{h}(-12 - 0.9) = 0 + 0 = 0$*. Thus,*

$$\mathbf{x}(3) = \begin{bmatrix} 2 \\ 2 \\ 0 \end{bmatrix}.$$

Similarly, we obtain $\mathbf{x}(3) = \mathbf{x}(4) = \ldots = \begin{bmatrix} 2 \\ 2 \\ 0 \end{bmatrix}$.

Therefore, the asynchronous version of the HNN settles down into the stationary state $\mathbf{x}(3)$ *in* 3 *iterations.*

1b) *In the synchronous update, that is, by using the evolution equation* (10)*, the HNN produces a limit cycle of length* 2*. Indeed,*

At time $t = 1$:

$$\mathbf{W} \cdot \mathbf{x}(0) = \begin{bmatrix} 0 & 3 & -2 \\ 3 & 0 & -4 \\ -2 & -4 & 0 \end{bmatrix} \cdot \begin{bmatrix} 0 \\ 1 \\ 1 \end{bmatrix} = \begin{bmatrix} 1 \\ -4 \\ -4 \end{bmatrix}.$$

Then,

$$\mathbf{x}(1) = \begin{bmatrix} \mathtt{h}(+1 - 0.5) + \mathtt{h}(+1 - 0.3) \\ \mathtt{h}(-4 - 0.1) + \mathtt{h}(-4 - 0.2) \\ \mathtt{h}(-4 - 0.1) + \mathtt{h}(-4 - 0.2) \end{bmatrix} = \begin{bmatrix} 2 \\ 0 \\ 0 \end{bmatrix}.$$

For $t = 2$:

$$\mathbf{W} \cdot \mathbf{x}(1) = \begin{bmatrix} 0 & 3 & -2 \\ 3 & 0 & -4 \\ -2 & -4 & 0 \end{bmatrix} \cdot \begin{bmatrix} 2 \\ 0 \\ 0 \end{bmatrix} = \begin{bmatrix} 0 \\ 6 \\ -4 \end{bmatrix}.$$

Consequently,

$$\mathbf{x}(2) = \begin{bmatrix} \mathtt{h}(0 - 0.5) + \mathtt{h}(0 - 0.3) \\ \mathtt{h}(+6 - 0.1) + \mathtt{h}(+6 - 0.2) \\ \mathtt{h}(-4 - 0.1) + \mathtt{h}(-4 - 0.2) \end{bmatrix} = \begin{bmatrix} 0 \\ 2 \\ 0 \end{bmatrix}.$$

Finally, at time $t = 3$:

$$\mathbf{W} \cdot \mathbf{x}(2) = \begin{bmatrix} 0 & 3 & -2 \\ 3 & 0 & -4 \\ -2 & -4 & 0 \end{bmatrix} \cdot \begin{bmatrix} 0 \\ 2 \\ 0 \end{bmatrix} = \begin{bmatrix} 6 \\ 0 \\ -8 \end{bmatrix}.$$

Then,

$$\mathbf{x}(3) = \begin{bmatrix} \mathtt{h}(+6 - 0.5) + \mathtt{h}(+6 - 0.3) \\ \mathtt{h}(0 - 0.1) + \mathtt{h}(0 - 0.2) \\ \mathtt{h}(-8 - 0.1) + \mathtt{h}(-8 - 0.2) \end{bmatrix} = \begin{bmatrix} 2 \\ 0 \\ 0 \end{bmatrix} = \mathbf{x}(1).$$

In this case, we obtain the periodic sequence

$$\mathbf{x}(t)_{t \geq 0} = \{\mathbf{x}(0), \mathbf{x}(1), \mathbf{x}(2), \mathbf{x}(1), \mathbf{x}(2), \ldots\}.$$

Unlike the asynchronous HNN, the synchronous HNN with the same input parameters produced a limit cycle of length 2, similarly to the binary case.

Experiment 2 *This experiment aims to illustrate some differences between asynchronous and synchronous dynamics of an HNN with celing neurons. Consider the synaptic weights matrix* $\mathbf{W} = \begin{bmatrix} 0 & -3 \\ -3 & 0 \end{bmatrix}$*, and the thresholds matrix* $\mathbf{\Theta} = \begin{bmatrix} -3.5 & -2.5 \\ 1.5 & -0.5 \end{bmatrix}$*. In this case,* $S = \{0, 1, 2\}$ *because the resolution factor of the HNN is* $K = 2$*. The HNN has the following set of state vectors:*

$$V = \left\{ \begin{bmatrix} 0 \\ 0 \end{bmatrix}, \begin{bmatrix} 0 \\ 1 \end{bmatrix}, \begin{bmatrix} 0 \\ 2 \end{bmatrix}, \begin{bmatrix} 1 \\ 0 \end{bmatrix}, \begin{bmatrix} 1 \\ 1 \end{bmatrix}, \begin{bmatrix} 1 \\ 2 \end{bmatrix}, \begin{bmatrix} 2 \\ 0 \end{bmatrix}, \begin{bmatrix} 2 \\ 1 \end{bmatrix}, \begin{bmatrix} 2 \\ 2 \end{bmatrix} \right\}.$$

Representing the elements of V *in the order in which they are arranged by the blue edges, marked with the digits* $0, 1 \ldots, 8$ *respectively, we can quickly visualize the dynamics generated by the Equation* (9) *(asynchronous update mode) through the directed graph illustrated by Figure 3. Note that the vector represented by the edge* 3*, corresponding to* $\begin{bmatrix} 1 \\ 0 \end{bmatrix}$*, is the state vector that most attracts other state vectors. The exceptions are the state vectors represented by the edges* 2*,* 5*, and* 8*.*

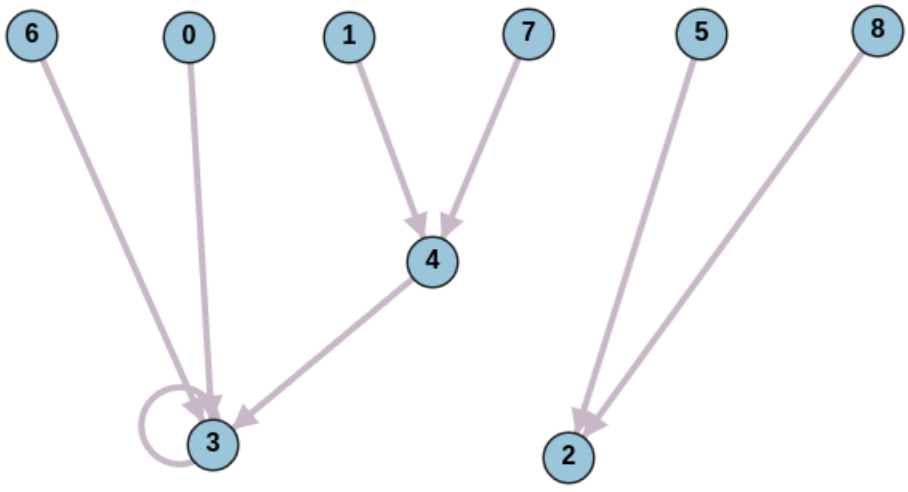

Figure 3: Directed graph illustrating the dynamics of the asynchronous HNN given by equation (9).

Similarly, consider the same matrices $\mathbf{W}$ *and* $\mathbf{\Theta}$*, but the synchronous HNN, according to the equation* (10)*. The dynamics of the HNN is represented by the*

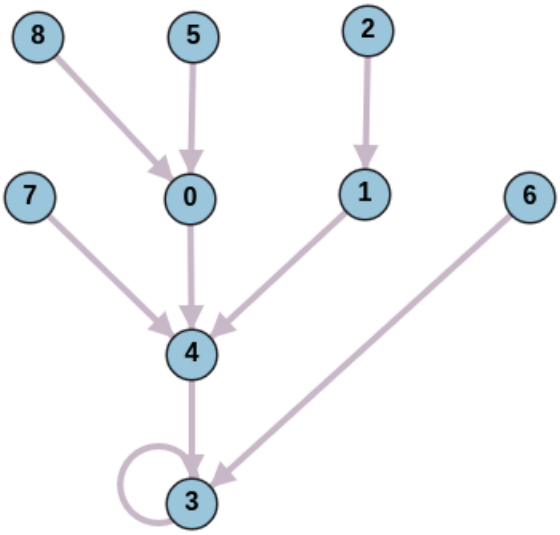

Figure 4: Directed graph illustrating the dynamics of the synchronous HNN given by equation (10).

directed graph of Figure 4. Note that the dynamics of relaxation of the HNN in the synchronous update is different from that of the asynchronous update, as we might expect. In the synchronous update, the state vector $\begin{bmatrix}1\\0\end{bmatrix}$ *is the only attractor of all the other vectors. In this particular experiment, by using asynchronous update mode, the number of stationary states in the HNN is double than the number of stationary states in synchronous updating. From the point of view of the use of HNNs for the implementation of content-addressable memories, this example suggests that the HNN with asynchronous updating could provide greater storage capacity than the synchronous HNN. Despite this, the HNN with the synchronous update mode does not always generate convergent sequences, which would hamper its use in the implementation of content-addressable memories. See the remark* 4*, and the Experiments* 1 *and* 3*.*

Experiment 3 *In order to evaluate the convergence of the sequences produced by the HNN models with ceiling neurons, we proceeded as follows: We first generated* 100×100 *real-valued matrix* $\mathbf{R}$ *with entries* $r_{ij} = \texttt{randn}$*, where* `randn`

yields a random scalar drawn from the standard normal distribution. Then, we computed $\mathbf{U} = \frac{1}{2}(\mathbf{R} + \mathbf{R}^T)$*, where* $\mathbf{R}^T$ *denotes the transpose of* $\mathbf{R}$*, and defined the synaptic weight matrix by* $\mathbf{W} = \mathbf{U} - \texttt{diag}(u_{11}, u_{22}, \ldots, u_{100100})$*,where* $\texttt{diag}(u_{11}, u_{22}, \ldots, u_{100100})$ *is the diagonal matrix composed of the diagonal elements of* $\mathbf{U}$*. Note that* $\mathbf{W}$ *satisfies* $w_{ij} = w_{ji}$ *and* $w_{ii} = 0$ *for all any indexes* i *and* j*. Figure 5 shows the probability of a randomly generated HNN model with ceiling neurons settles down into an equilibrium state in at most* 1000 *iterations, that is, we allowed the neural networks to evolve while* $t \leq 1000$*. Success probabilities have been computed by repeating the procedure* 1000 *times for each resolution factor* $K \in \{1, 2, \ldots, 10\}$*, and using the frequentist definition of probability to obtain a relative measure of the number of times that each HNN settled down to a stationary state. We would like to point out that the synaptic weight matrix* $\mathbf{W}$*, and the initial states* $\mathbf{x}(0)$ *were the same for all HNN models and were obtained by drawing each component of the set* $\{0, 1, \ldots, K\}^{100}$ *using a uniform distribution. The elements of thresholds matrix* $\boldsymbol{\Theta}$ *were obtained by drawing each component from of the real interval* $[-10, 10]$ *ensuring that each column of* $\boldsymbol{\Theta}$ *does not have repeated elements. Note that the probability of the synchronous model reaching a stationary state is always less than the probability of the corresponding asynchronous model. In addition, this probability decreases dramatically with the increase of the resolution factor* K*. In contrast, the probability of its asynchronous version to settle down into a stationary state is always equal to* 1*, which corroborates with Theorem 1.*

6 Concluding and Remarks

In this chapter, we introduce extensions of the Hopfield neural network based on ceiling neurons. Initially, we present a theoretical review of the classic Hopfield neural network model. Inspired by the mathematical definition of a ceiling neuron, we built a Hopfield neural network whose set of possible states for a neuron was expanded from the binary set $\{0, 1\}$ to the multistate set $S = \{0, 1, ..., K\}$, where $K \geq 1$ is an integer called the resolution factor of the neural network. Then, we discuss the dynamics of the proposed models in both asynchronous and synchronous

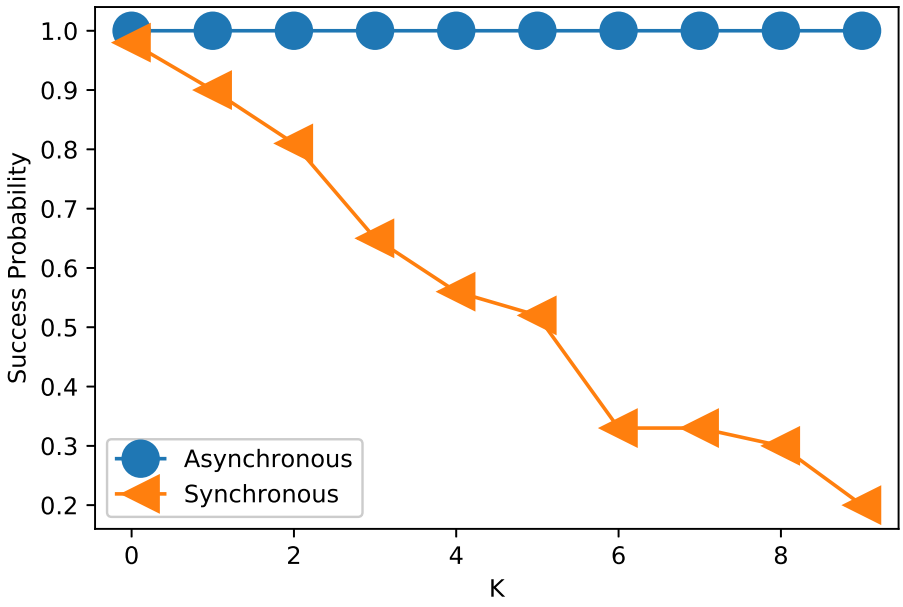

Figure 5: A comparison between the dynamics of HNN models with K ceiling neurons.

update modes. The asynchronous version of the model always settles down into a stationary state if the synaptic matrix $\mathbf{W}$ is symmetric with non-negative diagonal elements. In turn, the synchronous model does not always generate convergent sequences under the same conditions on the $\mathbf{W}$ matrix. Finally, we performed computational experiments to illustrate the dynamics explained throughout the text. As future work, we intend to implement content-addressable memories using the models based on ceiling neurons with different storage approaches aiming at storing and recalling grayscale and color images, as an alternative to complex-valued and quaternion-valued HNN models. In particular, we intend to further investigate the role of increasing the amount of thresholds for each neuron in view of the possibility of using the proposed models as specific image filters.

References

[1] AIZENBERG, I. *Complex-Valued Neural Networks with Multi-Valued Neurons*. Springer Publishing Company, Incorporated, 2016.

[2] AIZENBERG, N. N., AND AIZENBERG, I. N. CNN based on multivalued neuron as a model of associative memory for gray-scale images. In *Proceedings of the 2nd International Workshop on Cellular Neural Networks and Their Applications* (1992), pp. 36–42.

[3] BRUCK, J. On the convergence properties of the hopfield model. *Proceedings of the IEEE 78*, 10 (1990), 1579–1585.

[4] COHEN, M. A., AND GROSSBERG, S. Absolute stability of global pattern formation and parallel memory storage by competitive neural networks. *IEEE Transactions on Systems, Man, and Cybernetics SMC-13*, 5 (1983), 815–826.

[5] DE CASTRO, F. Z., AND VALLE, M. E. Continuous-valued octonionic Hopfield neural network. In *Proceedings Series of the Brazilian Society of Computational and Applied Mathematics. Sociedade Brasileira de Matemática Aplicada e Computacional.* (2018), vol. 06, pp. 2112–2118.

[6] DE CASTRO, F. Z., AND VALLE, M. E. Some remarks on the stability of discrete-time complex-valued multistate Hopfield neural networks. In *Proceeding Series of the Brazilian Society of Computational and Applied Mathematics* (2018), vol. 6.

[7] DE CASTRO, F. Z., AND VALLE, M. E. A broad class of discrete-time hypercomplex-valued Hopfield neural networks. *Neural Networks 122* (2020), 54 – 67.

[8] GAN, J. Discrete Hopfield neural network approach for crane safety evaluation. In *2017 International Conference on Mechanical, System and Control Engineering (ICMSC)* (May 2017), pp. 40–43.

[9] GARIMELLA, R. M. Some novel real/complex-valued neural network models. In *Computational Intelligence, Theory and Applications* (Berlin, Heidelberg, 2006), B. Reusch, Ed., Springer Berlin Heidelberg, pp. 473–483.

[10] GARIMELLA, R. M., MUNUGOTI, S. D., AND RAYALA, A. Novel ceiling neuron model and its applications. In *2019 International Joint Conference on*

Neural Networks (IJCNN) (2019), pp. 1–8.

[11] HEBB, D. *The Organization of Behavior*. John Wiley & Sons, New York, 1949.

[12] HOPFIELD, J., AND TANK, D. Neural computation of decisions in optimization problems. *Biological Cybernetics 52* (1985), 141–152.

[13] HOPFIELD, J. J. Neural Networks and Physical Systems with Emergent Collective Computational Abilities. *Proceedings of the National Academy of Sciences 79* (Apr. 1982), 2554–2558.

[14] HOPFIELD, J. J. Neurons with graded response have collective computational properties like those of two-state neurons. *Proceedings of the National Academy of Sciences 81* (May 1984), 3088–3092.

[15] ISOKAWA, T., HISHIMURA, H., KAMIURA, N., AND MATSUI, N. Associative Memory in Quaternionic Hopfield Neural Network. *International Journal of Neural Systems 18*, 02 (2008), 135–145.

[16] ISOKAWA, T., NISHIMURA, H., AND MATSUI, N. Quaternionic Neural Networks for Associative Memories. In *Complex-Valued Neural Networks*, A. Hirose, Ed. Wiley-IEEE Press, 2013, pp. 103–131.

[17] JANKOWSKI, S., LOZOWSKI, A., AND ZURADA, J. Complex-Valued Multistate Neural Associative Memory. *IEEE Transactions on Neural Networks 7* (1996), 1491–1496.

[18] KUROE, Y., AND IIMA, H. A model of Hopfield-type octonion neural networks and existing conditions of energy functions. In *2016 International Joint Conference on Neural Networks (IJCNN)* (2016), pp. 4426–4430.

[19] KUROE, Y., IIMA, H., AND MAEDA, Y. Four models of Hopfield-type octonion neural networks and their existing conditions of energy functions. In *2020 International Joint Conference on Neural Networks (IJCNN)* (2020), pp. 1–7.

[20] LI, C., YU, X., HUANG, T., CHEN, G., AND HE, X. A generalized Hopfield network for nonsmooth constrained convex optimization: Lie derivative approach. *IEEE Transactions on Neural Networks and Learning Systems 27* (11 2015), 1–14.

[21] LI, J., LI, X., HUANG, B., AND ZHAO, L. Hopfield neural network ap-

proach for supervised nonlinear spectral unmixing. *IEEE Geoscience and Remote Sensing Letters 13*, 7 (July 2016), 1002–1006.

[22] LITTLE, W. A. The existence of persistent states in the brain. *Mathematical biosciences 19*, 1-2 (1974), 101–120.

[23] MCCULLOCH, W., AND PITTS, W. A Logical Calculus of the Ideas Immanent in Nervous Activity. *Bulletin of Mathematical Biophysics 5* (1943), 115–133.

[24] MÜEZZINOĞLU, M., GÜZELIŞ, C., AND ZURADA, J. A New Design Method for the Complex-Valued Multistate Hopfield Associative Memory. *IEEE Transactions on Neural Networks 14*, 4 (July 2003), 891–899.

[25] NOEST, A. J. Discrete-state phasor neural networks. *Physical Review A 38* (Aug 1988), 2196–2199.

[26] PAJARES, G., GUIJARRO, M., AND RIBEIRO, A. A Hopfield neural network for combining classifiers applied to textured images. *Neural Networks 23*, 1 (Jan. 2010), 144–153.

[27] PARISI, G. I., KEMKER, R., PART, J. L., KANAN, C., AND WERMTER, S. Continual lifelong learning with neural networks: A review. *Neural Networks 113* (2019), 54–71.

[28] SERPEN, G. Hopfield Network as Static Optimizer: Learning the Weights and Eliminating the Guesswork. *Neural Processing Letters 27*, 1 (2008), 1–15.

[29] SONG, Y., XING, B., GUO, L., AND XU, X. System parameter identification experiment based on Hopfield neural network for self balancing vehicle. In *2017 36th Chinese Control Conference (CCC)* (July 2017), pp. 6887–6890.

[30] TANAKA, G., AND AIHARA, K. Complex-Valued Multistate Associative Memory With Nonlinear Multilevel Functions for Gray-Level Image Reconstruction. *IEEE Transactions on Neural Networks 20*, 9 (Sept. 2009), 1463–1473.

[31] VALLE, M. E. An Introduction to Complex-Valued Recurrent Correlation Neural Networks. In *Proceedings of the IEEE World Conference on Computational Intelligence 2014 (WCCI 2014)* (Beijing, China, July 2014).

[32] VALLE, M. E., AND DE CASTRO, F. Z. Theoretical and computational as-

pects of quaternionic multivalued Hopfield neural networks. In *2016 International Joint Conference on Neural Networks (IJCNN)* (July 2016), pp. 4418–4425.

[33] VALLE, M. E., AND DE CASTRO, F. Z. On the dynamics of Hopfield neural networks on unit quaternions. *IEEE transactions on neural networks and learning systems 29*, 6 (2017), 2464–2471.

[34] WANG, Q., SHI, W., ATKINSON, P. M., AND LI, Z. Land cover change detection at subpixel resolution with a Hopfield neural network. *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing 8*, 3 (March 2015), 1339–1352.

[35] ZHANG, H., HOU, Y., ZHAO, J., WANG, L., XI, T., AND LI, Y. Automatic welding quality classification for the spot welding based on the Hopfield associative memory neural network and chernoff face description of the electrode displacement signal features. *Mechanical Systems and Signal Processing 85* (2017), 1035 – 1043.