NOTE **Communicated by John Lazzaro**

On the Computational Power of Winner-Take-All

Wolfgang Maass
Institute for Theoretical Computer Science, Technische Universität Graz, A-8010 Graz, Austria

This article initiates a rigorous theoretical analysis of the computational power of circuits that employ modules for computing winner-take-all. Computational models that involve competitive stages have so far been neglected in computational complexity theory, although they are widely used in computational brain models, artificial neural networks, and analog VLSI. Our theoretical analysis shows that winner-take-all is a surprisingly powerful computational module in comparison with threshold gates (also referred to as McCulloch-Pitts neurons) and sigmoidal gates. We prove an optimal quadratic lower bound for computing winner-take-all in any feedforward circuit consisting of threshold gates. In addition we show that arbitrary continuous functions can be approximated by circuits employing a single soft winner-take-all gate as their only nonlinear operation.

Our theoretical analysis also provides answers to two basic questions raised by neurophysiologists in view of the well-known asymmetry between excitatory and inhibitory connections in cortical circuits: how much computational power of neural networks is lost if only positive weights are employed in weighted sums and how much adaptive capability is lost if only the positive weights are subject to plasticity.

1 Introduction

Computational models that involve competitive stages are widely used in computational brain models, artificial neural networks, and analog VLSI (see Arbib, 1995). The simplest competitive computational module is a hard winner-take-all gate that computes a function $\mathrm{WTA}_n\colon \mathbb{R}^n \to \{0,1\}^n$ whose output $\langle b_1, \dots, b_n\rangle = \mathrm{WTA}_n(x_1, \dots, x_n)$ satisfies

$$b_i = \begin{cases} 1, & \text{if} \quad x_i > x_j \quad \text{for all } \ j \neq i \\ 0, & \text{if} \quad x_j > x_i \quad \text{for some } \ j \neq i. \end{cases}$$

Thus, in the case of pairwise different inputs $x_1, \dots, x_n$ a single output bit b_i has value 1, which marks the position of the largest input x_i.[1]

[1] Different conventions are considered in the literature in case that there is no unique "winner" x_i, but we need not specify any convention in this article since our lower-bound result for WTA_n holds for all of these versions.

In this article we also investigate the computational power of two common variations of winner-take-all: k-winner-take-all, where the ith output b_i has value 1 if and only if x_i is among the k largest inputs, and soft winner-take-all, where the ith output is an analog variable r_i whose value reflects the rank of x_i among the input variables.

Winner-take-all is ubiquitous as a computational module in computational brain models, especially in models involving computational mechanisms for attention (Niebur & Koch, 1998). Biologically plausible models for computing winner-take-all in biological neural systems exist on the basis of both the assumption that the analog inputs x_i are encoded through firing rates—where the most frequently firing neuron exerts the strongest inhibition on its competitors and thereby stops them from firing after a while—and the assumption that the analog inputs x_i are encoded through the temporal delays of single spikes—where the earliest-firing neuron (that encodes the largest x_i) inhibits its competitors before they can fire (Thorpe, 1990).

We would like to point to another link between results of this article and computational neuroscience. There exists a notable difference between the computational role of weights of different signs in artificial neural network models on one hand, and anatomical and physiological data regarding the interplay of excitatory and inhibitory neural inputs in biological neural systems on the other hand (see Abeles, 1991, and Shepherd, 1998). Virtually any artificial neural network model is based on an assumed symmetry between positive and negative weights. Typically weights of either sign occur on an equal footing as coefficients in weighted sums that represent the input to an artificial neuron. In contrast, there exists a strong asymmetry regarding positive (excitatory) and negative (inhibitory) inputs to biological neurons. At most 15% of neurons in the cortex are inhibitory neurons, and these are the only neurons that can exert a negative (inhibitory) influence on the activity of other neurons. Furthermore, the location and structure of their synaptic connections to other neurons differ drastically from those formed by excitatory neurons. Inhibitory neurons are usually connected to just neurons in their immediate vicinity, and it is not clear to what extent their synapses are changed through learning. Furthermore, the location of their synapses on the target neurons (rarely on spines, often close to the soma, frequently with multiple synapses on the target neuron) suggests that their computational function is not symmetric to that of excitatory synapses. Rather, such data would support a conjecture that their impact on other neurons may be more of a local regulatory nature—that inhibitory neurons do not function as computational units per se like the (excitatory) pyramidal neurons, whose synapses are subject to fine-tuning by various learning mechanisms. These observations from anatomy and neurophysiology have given rise to the question of whether a quite different style of neural circuit design may be feasible, which achieves sufficient computational power without requiring symmetry between excitatory and inhibitory interaction among neurons. The circuits constructed in section 3 and 4 of this article provide a positive answer to this question. It is shown there that neural circuits that use inhibition exclusively for lateral inhibition in the context of winner-take-all have the same computational power as multilayer perceptrons that employ weighted sums with positive and negative weights in the usual manner. Furthermore, one can, if one wants, keep the inhibitory synapses in these circuits fixed and modify just the excitatory synapses in order to program the circuits so that they adopt a desired input-output behavior. It has long been known that winner-take-all can be implemented via inhibitory neurons in biologically realistic circuit models (Elias & Grossberg, 1975; Amari & Arbib, 1977; Coultrip, Granger, & Lynch, 1992; Yuille and Grzywacz, 1989). The only novel contribution of this article is the result that in combination with neurons that compute weighted sums (with positive weights only), such winner-take-all modules have universal computational power for both digital and analog computation.

A large number of efficient implementations of winner-take-all in analog VLSI have been proposed, starting with Lazzaro, Ryckebusch, Mahowald, and Mead (1989). The circuit of Lazzaro et al. computes an approximate version of WTA_n with just $2n$ transistors and wires of total length $O(n)$, with lateral inhibition implemented by adding currents on a single wire of length $O(n)$. Its computation timescales with the size of its largest input. Numerous other efficient implementations of winner-take-all in analog VLSI have subsequently been produced (e.g., Andreou et al., 1991; Choi & Sheu, 1993; Fang, Cohen, & Kincaid, 1996). Among them are circuits based on silicon spiking neurons (DeYong, Findley, & Fields, 1992; Meador & Hylander, 1994; Indiveri, in press) and circuits that emulate attention in artificial sensory processing (DeWeerth and Morris, 1994; Horiuchi, Morris, Koch, & DeWeerth, 1997; Brajovic & Kanade, 1998; Indiveri, in press). In spite of these numerous hardware implementations of winner-take-all and numerous practical applications, we are not aware of any theoretical results on the general computational power of these modules. It is one goal of this article to place these novel circuits in the context of other circuit models that are commonly studied in computational complexity theory and to evaluate their relative strength.

There exists an important structural difference between those that are commonly studied in computational complexity theory and those that one typically encounters in hardware or wetware. Almost all circuit models in computational complexity theory are feedforward circuits—their architecture constitutes a directed graph without cycles (Wegener, 1987; Savage, 1998). In contrast, typical physical realizations of circuits contain besides feedforward connections also lateral connections (connections among gates on the same layer), and frequently also recurrent connections (connections from a higher-layer backward to some lower layer of the circuit). This gives rise to the question whether this discrepancy is relevant—for example, whether there are practically important computational tasks that can be implemented substantially more efficiently on a circuit with lateral connections than on a strictly feedforward circuit. In this article, we address this issue in the following manner: we view modules that compute winner-take-all (whose implementation usually involves lateral connections) as "black boxes" of which we only model their input-output behavior. We refer to these modules as winner-take-all gates in the following. We study possible computational uses of such winner-take-all gates in circuits where these gates (and other gates) are wired together in a feedforward fashion.[2] We charge one unit of time for the operation of each gate. We will show that in this framework, the new modules, which may internally involve lateral connections, do in fact add substantial computational power to a feedforward circuit.

The arguably most powerful gates that have previously been studied in computational complexity theory are *threshold gates* (also referred to as McCulloch-Pitts neurons or perceptrons; see Minsky & Papert, 1969; Siu, Roychowdury, & Kailath, 1995) and sigmoidal gates (which may be viewed as soft versions of threshold gates). A threshold gate with weights $\alpha_1, \ldots, \alpha_n \in \mathbb{R}$ and threshold $\Theta \in \mathbb{R}$ computes the function $G\colon \mathbb{R}^n \to \{0, 1\}$ defined by $G(x_1, \ldots, x_n) = 1 \Leftrightarrow \sum_{i=1}^{n} \alpha_i x_i \geq \Theta$. Note that AND and OR of n bits as well as NOT are special cases of threshold gates. A threshold circuit (also referred to as multilayer perceptron) is a feedforward circuit consisting of threshold gates. The depth of a threshold circuit C is the maximal length of a directed path from an input node to an output gate. The circuit is called layered if all such paths have the same length. Note that a circuit with k hidden layers has depth $k + 1$ in this terminology. Except for theorem 1 we will discuss in this article only circuits with a single output. The other results can be extended to networks with several outputs by duplicating the network so that each output variable is formally computed by a separate network.

We will prove in section 2 that WTA_n is a rather expensive computational operation from the point of view of threshold circuits, since any such circuit needs quadratically in n many gates to compute WTA_n. We will show in section 3 that the full computational power of two layers of threshold gates can be achieved by a single k-winner-take-all gate applied to positive weighted sums of the input variables. Furthermore, we will show in section 4 that by replacing the single k-winner-take-all gate by a single soft winner-take-all gate, these extremely simple circuits become universal approximators for arbitrary continuous functions.

[2] We examine computations in such a circuit for only a single batch-input, not for streams of varying inputs. Hence, it does not matter for this analysis whether one implements a winner-take-all gate by a circuit that needs to be reinitialized before the next input, or by a circuit that immediately responds to changes in its input.

2 An Optimal Quadratic Lower Bound for Hard Winner-Take-All

We show in this section that any feedforward circuit consisting of threshold gates needs to consist of quadratically in n many gates for computing WTA_n. This result also implies a lower bound for any circuit of threshold gates involving lateral and/or recurrent connections that compute WTA_n (provided one assumes that each gate receives at time t only outputs from other gates that were computed before time t). One can simulate any circuit with s gates whose computation takes t discrete time steps by a feedforward circuit with $s \cdot t$ gates whose computation takes t discrete time steps. Hence, for example, if there exists some circuit consisting of $O(n)$ threshold gates with lateral and recurrent connections that computes WTA_n in t discrete time steps, then WTA_n can be computed by a feedforward circuit consisting of $O(t \cdot n)$ threshold gates. Therefore theorem 1 implies that WTA_n cannot be computed in sublinear time by any linear-size circuit consisting of threshold gates with arbitrary (i.e., feedforward, lateral, and recurrent) connections. In case linear-size implementations of (approximations of) WTA_n in analog VLSI can be built whose computation time grows sublinearly in n, then this negative result would provide theoretical evidence for the superior computational capabilities of analog circuits with lateral connections.

For $n = 2$ it is obvious that WTA_n can be computed by a threshold circuit of size 2 and that this size is optimal. For $n \geq 3$ the most straightforward design of a (feedforward) threshold circuit C that computes WTA_n uses $\binom{n}{2} + n$ threshold gates. For each pair $\langle i, j\rangle$ with $1 \leq i < j \leq n$, one employs a threshold gate G_{ij} that outputs 1 if and only if $x_j \geq x_i$. The ith output b_i of WTA_n for a circuit input $\underline{x}$ is computed by a threshold gate G_i with $G_i = 1 \Leftrightarrow \sum_{j<i} G_{ji}(\underline{x}) + \sum_{j>i} -G_{ij}(\underline{x}) \geq i - 1$.

This circuit design appears to be suboptimal, since most of its threshold gates—the $\binom{n}{2}$ gates G_{ij}—do not make use of their capability to evaluate weighted sums of many variables, with arbitrary weights from $\mathbb{R}$. However, the following result shows that no feedforward threshold circuit (not even with an arbitrary number of layers, and threshold gates of arbitrary fan-in with arbitrary real weights) can compute WTA_n with fewer than $\binom{n}{2} + n$ gates.

Theorem 1. *Assume that $n \geq 3$ and WTA_n is computed by some arbitrary feedforward circuit C consisting of threshold gates with arbitrary weights. Then C consists of at least $\binom{n}{2} + n$ threshold gates.*

Proof. Let C be any threshold circuit that computes WTA_n for all $\underline{x} = \langle x_1, \ldots, x_n\rangle \in \mathbb{R}^n$ with pairwise different $x_1, \ldots, x_n$. We say that a threshold gate in the circuit C contains an input variable x_k if there exists a direct "wire" (i.e., an edge in the directed graph underlying C) from the kth input node to this gate, and the kth input variable x_k occurs in the weighted sum of this threshold gate with a weight $\neq 0$. This threshold gate may also receive outputs from other threshold gates as part of its input, since we do not assume that C is a layered circuit.

Fix any $i, j \in \{1, \ldots, n\}$ with $i \neq j$. We will show that there exists a gate G_{ij} in C that contains the input variables x_i and x_j, but no other input variables. The proof proceeds in four steps:

Step 1. Choose $h \in (0.6, 0.9)$ and $\rho \in (0, 0.1)$ such that no gate G in C that contains the input variable x_j, but no other input variable, changes its output value when x_j varies over $(h-\rho, h+\rho)$, no matter which fixed binary values $\underline{a}$ have been assigned to the other inputs of gate G. For any fixed $\underline{a}$ there exists at most a single value $t_{\underline{a}}$ for x_j, so that G changes its output for $x_j = t_{\underline{a}}$ when x_j varies from $-\infty$ to $+\infty$. It suffices to choose h and ρ so that none of these finitely many values of $t_{\underline{a}}$ (for arbitrary binary $\underline{a}$ and arbitrary gates G) falls into the interval $(h-\rho, h+\rho)$.

Step 2. Choose a closed ball $B \subseteq (0, 0.5)^{n-2}$ with a center $\underline{c} \in \mathbb{R}^{n-2}$ and a radius $\delta > 0$ so that no gate G in C changes its output when we set $x_i = x_j = h$ and let the vector of the other $n-2$ input variables vary over B (this is required to hold for any fixed binary values of the inputs that G may receive from other threshold gates). We exploit here that for fixed $x_i = x_j = h$, the gates in C (with inputs from other gates replaced by all possible binary values) partition $(0, 0.5)^{n-2}$ into finitely many sets S, each of which can be written as an intersection of half-spaces, so that no gate in C changes its output when we fix $x_i = x_j = h$ and let the other $n-2$ input variables of the circuit vary over S (while keeping inputs to C from other gates artificially fixed).

Step 3. Choose $\gamma \in (0, \rho)$ so that in every gate G in C that contains besides x_j some other input variable x_k with $k \notin \{i, j\}$, a change of x_j by an amount γ causes a smaller change of the weighted sum at this threshold gate G than a change by an amount δ of any of the input variables x_k with $k \notin \{i, j\}$ that it contains. This property can be satisfied because we can choose for γ an arbitrarily small positive number.

Step 4. Set $x_i := h$, $\langle x_k \rangle_{k \notin \{i,j\}} := \underline{c}$, and let x_j vary over $[h - \frac{\gamma}{2}, h + \frac{\gamma}{2}]$. By assumption, the output of C changes since the output of $\text{WTA}_n(x_1, \ldots, x_n)$ changes (x_i is the winner for $x_j < h$, x_j is the winner for $x_j > h$). Let G_{ij} be some gate in C that changes its output, whereas no gate that lies on a path from an input variable to G_{ij} changes its output. Hence, G_{ij} contains the input variable x_j. The choice of h and $\rho > \gamma$ in step 1 implies that G_{ij} contains besides x_j some other input variable. Assume for a contradiction that G_{ij} contains an input variable x_k with $k \notin \{i, j\}$. By the choice of γ in step 3 this implies that the output of G_{ij} changes when we set $x_i = x_j = h$ and move x_k by an amount up to δ from its value in $\underline{c}$, while keeping all other input variables and inputs that G_{ij} receives from other threshold gates fixed (even if some preceding threshold gates in C would change their output in response to this change in the input variable x_k). This yields a contradiction to the definition of the ball B in step 2. Thus we have shown that $k \in \{i, j\}$ for any input variable x_k that G_{ij} contains. Therefore G_{ij} contains exactly the input variables x_i and x_j.

So far we have shown that for any $i, j \in \{1, \ldots, n\}$ with $i \neq j$, there exists a gate G_{ij} in C that contains the two input variables x_i, x_j, and no other input variables.

It remains to be shown that apart from these $\binom{n}{2}$ gates G_{ij} the circuit C contains n other gates $G_1, \ldots, G_n$ that compute the n output bits $b_1, \ldots, b_n$ of WTA$_n$. It is impossible that $G_k = G_l$ for some $k \neq l$, since $b_k \neq b_l$ for some arguments of WTA$_n$. Assume for a contradiction that $G_k = G_{ij}$ for some $k, i, j \in \{1, \ldots, n\}$ with $i \neq j$. We have $k \neq i$ or $k \neq j$. Assume without loss of generality that $k \neq i$. Let l be any number in $\{1, \ldots, n\} - \{k, i\}$. Assign to $x_1, \ldots, x_n$ some pairwise different values $a_1, \ldots, a_n$ so that $a_l > a_{l'}$ for all $l' \neq l$. Since $l \neq k$ and $i \neq k$, we have that for any $x_i \in \mathbb{R}$, the kth output variable b_k of WTA$_n(a_1, \ldots, a_{i-1}, x_i, a_{i+1}, \ldots a_n)$ has value 0 (a_k cannot be the maximal argument for any value of x_i since $a_l > a_k$). On the other hand, since G_{ij} contains the variable x_i, there exist values for x_i (move $x_i \to \infty$ or $x_i \to -\infty$) so that G_{ij} outputs 1, no matter which binary values are assigned to the inputs that G_{ij} receives in C from other threshold gates (while we keep x_j fixed at value a_j). This implies that G_{ij} does not output b_k in circuit C, that is, $G_k \neq G_{ij}$. Therefore, the circuit C contains in addition to the gates G_{ij} n other gates that provide the circuit outputs $b_1, \ldots, b_n$.

3 Simulating Two Layers of Threshold Gates with a Single *k*-Winner-Take-All Gate as the Only Nonlinearity

A popular variation of winner-take-all (which has also been implemented in analog VLSI; Urahama & Nagao, 1995) is k-winner-take-all. The output of k-winner-take-all indicates for each input variable x_i whether x_i is among the k largest inputs. Formally we define for any $k \in \{1, \ldots, n\}$ the function k-WTA$_n$: $\mathbb{R}^n \to \{0, 1\}^n$ where k-WTA$_n$ $(x_1, \ldots, x_n) = \langle b_1, \ldots, b_n \rangle$ has the property that

$$b_i = 1 \Leftrightarrow (x_j > x_i \text{ holds for at most } k - 1 \text{ indices } j).$$

We will show in this section that a single gate that computes k-WTA$_n$ can absorb all nonlinear computational operations of any two-layer threshold circuit with one output gate and any number of hidden gates.

Theorem 2. *Any two-layer feedforward circuit C (with m analog or binary input variables and one binary output variable) consisting of threshold gates can be simulated by a circuit consisting of a single k-winner-take-all gate k-WTA$_n$ applied to n weighted sums of the input variables with positive weights, except for some set $S \subseteq \mathbb{R}^m$ of inputs that has measure 0.*

In particular, any boolean function f: $\{0, 1\}^m \rightarrow \{0, 1\}$ can be computed by a single k-winner-take-all gate applied to weighted sums of the input bits.

If C has polynomial size and integer weights, whose size is bounded by a polynomial in m, then n can be bounded by a polynomial in m, and all weights in the simulating circuit are natural numbers whose size is bounded by a polynomial in m.

Remark 1. The exception set of measure 0 in this result is a union of up to n hyperplanes in $\mathbb{R}^m$. This exception set is apparently of no practical significance, since for any given finite set $\tilde{D}$, not just for $\tilde{D} \subseteq \{0, 1\}^m$ but, for example, for $\tilde{D} := \{\langle z_1, \ldots, z_m\rangle \in \mathbb{R}^m$: each z_i is a rational number with bit-length $\leq 1000\}$, one can move these hyperplanes (by adjusting the constant terms in their definitions) so that no point in $\tilde{D}$ belongs to the exception set. Hence the k-WTA circuit can simulate the given threshold circuit C on any finite set $\tilde{D}$, with an architecture that is independent of $\tilde{D}$.

On the other hand, the proof of the lower bound result from theorem 1 requires that the circuit C computes WTA everywhere, not just on a finite set.

Remark 2. One can easily show that the exception set S of measure 0 in theorem 2 is necessary: The set of inputs $\underline{z} \in \mathbb{R}^m$ for which a k-WTA$_n$ gate applied to weighted sums outputs 1 is always a closed set, whereas the set of inputs for which a circuit C of depth 2 consisting of threshold gates outputs 1 can be an open set. Hence in general, a circuit C of the latter type cannot be simulated for all inputs $\underline{z} \in \mathbb{R}^m$ by a k-WTA$_n$ gate applied to weighted sums.

Corollary 1. *Any layered threshold circuit C of arbitrary even depth d can be simulated for all inputs except for a set of measure 0 by a feedforward circuit consisting of ℓ k-WTA gates on $\frac{d}{2}$ layers (each applied to positive weighted sums of circuit inputs on the first layer and of outputs from preceding k-WTA gates on the subsequent layers), where ℓ is the number of gates on even-numbered layers in C. Thus, one can view the simulating circuit as a d-layer circuit with alternating layers of linear and k-WTA gates.*

Alternatively one can simulate the layers 3 to d of circuit C by a single k-WTA gate applied to a (possibly very large) number of linear gates. This yields a simulation of C (except for some input set S of measure 0) by a four-layer circuit with alternating layers of linear gates and k-WTA gates. The number of k-WTA gates in this circuit can be bounded by $\ell_2 + 1$, where ℓ_2 is the number of threshold gates on layer 2 of C.

Proof of Theorem 2. Since the outputs of the gates on the hidden layer of C are from $\{0, 1\}$, we can assume without loss of generality that the weights $\alpha_1, \ldots, \alpha_n$ of the output gate G of C are from $\{-1, 1\}$. (See, e.g., Siu et al., 1995, for details. One first observes that it suffices to use integer weights for threshold gates with binary inputs; one can then normalize these weights to values in $\{-1, 1\}$ by duplicating gates on the hidden layer of C.) Thus, for any $\underline{z} \in \mathbb{R}^m$, we have $C(\underline{z}) = 1 \Leftrightarrow \sum_{j=1}^{n} \alpha_j \hat{G}_j(\underline{z}) \geq \Theta$, where $\hat{G}_1, \ldots, \hat{G}_n$ are the threshold gates on the hidden layer of C, $\alpha_1, \ldots, \alpha_n$ are from $\{-1, 1\}$, and Θ is the threshold of the output gate G. In order to eliminate the negative weights in G, we replace each gate $\hat{G}_j$ for which $\alpha_j = -1$ by another threshold gate G_j so that $G_j(\underline{z}) = 1 - \hat{G}_j(\underline{z})$ for all $\underline{z} \in \mathbb{R}^m$ except on some hyperplane. We utilize here that $\neg \sum_{i=1}^{m} w_i z_i \geq \Theta \Leftrightarrow \sum_{i=1}^{m} (-w_i) z_i > -\Theta$ for arbitrary $w_i, z_i, \Theta \in \mathbb{R}$. We set $G_j := \hat{G}_j$ for all $j \in \{1, \ldots, n\}$ with $\alpha_j = 1$. Then we have for all $\underline{z} \in \mathbb{R}^m$, except for $\underline{z}$ from some exception set S consisting of up to n hyperplanes,

$$\sum_{j=1}^{n} \alpha_j \hat{G}_j(\underline{z}) = \sum_{j=1}^{n} G_j(\underline{z}) - |\{j \in \{1, \ldots, n\}\colon \alpha_j = -1\}|.$$

Hence $C(\underline{z}) = 1 \Leftrightarrow \sum_{j=1}^{n} G_j(\underline{z}) \geq k$ for all $\underline{z} \in \mathbb{R}^m - S$, for some suitable $k \in \mathbb{N}$.

Let $w_1^j, \ldots, w_m^j \in \mathbb{R}$ be the weights and $\Theta^j \in \mathbb{R}$ be the threshold of gate G_j, $j = 1, \ldots, n$. Thus, $G_j(\underline{z}) = 1 \Leftrightarrow \sum_{i\colon w_i^j > 0} |w_i^j| z_i - \sum_{i\colon w_i^j < 0} |w_i^j| z_i \geq \Theta^j$. Hence with

$$S_j\colon = \sum_{i\colon w_i^j < 0} |w_i^j| z_i + \Theta^j + \sum_{\ell \neq j} \sum_{i\colon w_i^\ell > 0} |w_i^\ell| z_i \qquad \text{for } j = 1, \ldots, n$$

and

$$S_{n+1}\colon = \sum_{j=1}^{n} \sum_{i\colon w_i^j > 0} |w_i^j| z_i,$$

we have for every $j \in \{1, \ldots, n\}$ and every $\underline{z} \in \mathbb{R}^m$:

$$S_{n+1} \geq S_j \Leftrightarrow \sum_{i\colon w_i^j > 0} |w_i^j| z_i - \sum_{i\colon w_i^j < 0} |w_i^j| z_i \;\; \geq \;\; \Theta^j \Leftrightarrow G_j(\underline{z}) = 1.$$

This implies that the $(n+1)$st output b_{n+1} of the gate $(n - k + 1)-\mathrm{WTA}_{n+1}$ applied to $S_1, \ldots, S_{n+1}$ satisfies

$$\begin{aligned} b_{n+1} = 1 &\Leftrightarrow |\{j \in \{1, \ldots, n+1\}\colon S_j > S_{n+1}\}| \leq n - k \\ &\Leftrightarrow |\{j \in \{1, \ldots, n+1\}\colon S_{n+1} \geq S_j\}| \geq k + 1 \\ &\Leftrightarrow |\{j \in \{1, \ldots, n\}\colon S_{n+1} \geq S_j\}| \geq k \\ &\Leftrightarrow \sum_{j=1}^{n} G_j(\underline{z}) \geq k \\ &\Leftrightarrow C(\underline{z}) = 1. \end{aligned}$$

Note that all the coefficients in the sums $S_1, \ldots, S_{n+1}$ are positive.

Proof of Corollary 1. Apply the preceding construction separately to the first two layers of C, then to the second two layers of C, and so on. Note that the later layers of C receive just boolean inputs from the preceding layer. Hence one can simulate the computation of layer 3 to d also by a two-layer threshold circuit, which requires just a single $k-$WTA gate for its simulation in our approach from theorem 2.

4 Soft Winner-Take-All Applied to Positive Weighted Sums Is an Universal Approximator

We consider in this section a soft version soft-WTA of a winner-take-all gate. Its output $\langle r_1, \ldots, r_n\rangle$ for real valued inputs $\langle x_1, \ldots, x_n\rangle$ consists of n analog numbers r_i, whose value reflects the relative position of x_i within the ordering of $x_1, \ldots, x_n$ according to their size. Soft versions of winner-take-all are also quite plausible as computational function of cortical circuits with lateral inhibition. Efficient implementations in analog VLSI are still elusive, but in Indiveri (in press), an analog VLSI circuit for a version of soft winner-take-all has been presented where the time when the ith output unit is activated reflects the rank of x_i among the inputs.

We show in this section that single gates from a fairly large class of soft winner-take-all gates can serve as the only nonlinearity in universal approximators for arbitrary continuous functions. The only other computational operations needed are weighted sums with positive weights. We start with a basic version of a soft winner-take-all gate soft-WTA$_{n,k}$ for natural numbers k, n with $k \leq \frac{n}{2}$, which computes the following function $\langle x_1, \ldots, x_n\rangle \mapsto \langle r_1, \ldots, r_n\rangle$ from $\mathbb{R}^n$ into $[0, 1]^n$:

$$r_i := \pi\left(\frac{|\{j \in \{1, \ldots, n\}\colon\ x_i \geq x_j\}| - \frac{n}{2}}{k}\right),$$

where $\pi\colon \mathbb{R} \to [0, 1]$ is the piecewise linear function defined by

$$\pi(x) = \begin{cases} 1, & \text{if} \quad x > 1 \\ x, & \text{if} \quad 0 \leq x \leq 1 \\ 0, & \text{if} \quad x < 0. \end{cases}$$

If one implements a soft winner-take-all gate via lateral inhibition, one can expect that its ith output r_i is lowered (down from 1) by every competitor x_j that wins the competition with x_i (i.e., $x_j > x_i$). Furthermore, one can expect that the ith output r_i is completely annihilated (i.e., is set equal to 0) once the number of competitors x_j that win the competition with x_i reach a certain critical value c. This intuition has served as a guiding principle in the previously described formalization. However, for the sake of mathematical simplicity, we have defined the outputs r_i of soft-WTA$_{n,k}$ not in terms of the number of competitors x_j that win over x_i (i.e., $x_j > x_i$), but rather in terms of the number of competitors x_j that do not win over x_i (i.e., $x_i \geq x_j$). Obviously one has $|\{j \in \{1,\ldots,n\}\colon\ x_i \geq x_j\}| = n - |\{j \in \{1,\ldots,n\} : x_j > x_i\}|$. Hence, the value of r_i goes down if more competitors x_j win over x_i—as desired. The critical number c of winning competitors that suffice to "annihilate" r_i is formalized through the "threshold" $\frac{n}{2}$. Our subsequent result shows that in principle, it suffices to set the annihilation-threshold always equal to $\frac{n}{2}$. But in applications, one may, of course, choose other values for the annihilation threshold.

It is likely that an analog implementation of a soft-winner-take-all gate will not be able to produce an output that is really linearly related to $|\{j \in \{1,\ldots,n\}\colon\ x_i \geq x_j\}| - \frac{n}{2}$ over a sufficiently large range of values. Instead, an analog implementation is more likely to output a value of the form

$$g\left(\frac{|\{j \in \{1,\ldots,n\}\colon\ x_i \geq x_j\}| - \frac{n}{2}}{k}\right),$$

where the piecewise linear function π is replaced by some possibly rather complicated nonlinear warping of the target output values. The following result shows that such gates with any given nonlinear warping g can serve just as well as the competitive stage (and as the only nonlinearity) in universal approximators. More precisely, let g be any continuous function $g\colon \mathbb{R} \to [0,1]$ that has value 1 for $x > 1$, value 0 for $x < 0$, and which is strictly increasing over the interval $[0,1]$. We denote a soft winner-take-all gate that employs g instead of π by soft-WTA$^g_{n,k}$.

Theorem 3. *Assume that $h\colon D \to [0,1]$ is an arbitrary continuous function with a bounded and closed domain $D \subseteq \mathbb{R}^m$ (for example: $D = [0,1]^m$). Then for any $\varepsilon > 0$ and for any function g (satisfying above conditions) there exist natural numbers k, n, biases $\alpha^j_0 \in \mathbb{R}$, and nonnegative coefficients α^j_i for $i = 1,\ldots,m$ and $j = 1,\ldots,n$, so that the circuit consisting of the soft winner-take-all gate soft-WTA$^g_{n,k}$ applied to the n sums $\sum_{i=1}^m \alpha^j_i z_i + \alpha^j_0$ for $j = 1,\ldots,n$ computes a function[3] $f\colon D \to [0,1]$ so that $|f(\underline{z}) - h(\underline{z})| < \varepsilon$ for all $\underline{z} \in D$.*

Thus, circuits consisting of a single soft WTA-gate applied to positive weighted sums of the input variables are universal approximators for continuous functions.

Remark 3. The proof shows that the number n of sums that arise in the circuit constructed in theorem 3 can essentially be bounded in terms of the number of hidden gates in a one-hidden-layer circuit C of sigmoidal gates that approximates the given continuous function h (more precisely,

[3] More precisely, we set $f(\underline{z}) := r_n$ for the nth output variable r_n of soft-WTA$^g_{n,k}$.

the function $g^{-1} \circ h$), the maximal size of weights (expressed as multiple of the smallest nonzero weight) on the second layer of C, and $1/\varepsilon$ (where ε is the desired approximation precision). Numerous practical applications of backprop suggest that at least the first one of these three numbers can be kept fairly small for most functions h that are of practical interest.

Proof of Theorem 3. The proof has the following structure. We first apply the regular universal approximation theorem for sigmoidal neural nets in order to approximate the continuous function $g^{-1}(h(\underline{z}))$ by a weighted sum $\sum_{j=1}^{\tilde{n}} \tilde{\beta}_j G_j^1(\underline{z})$ of π-gates, that is, of sigmoidal gates G_j^1 that employ the piecewise linear activation function π that has already been defined. As the next step, we simplify the weights and approximate the weighted sum $\sum_{j=1}^{\tilde{n}} \tilde{\beta}_j G_j^1(\underline{z})$ by another weighted sum $\sum_{j=1}^{\hat{n}} \frac{\alpha_j}{\hat{k}} G_j^2(\underline{z})$ of π gates G_j^2 with $\alpha_j \in \{-1, 1\}$ for $j = 1, \ldots, \hat{n}$ and some suitable $\hat{k} \in \mathbb{N}$. We then eliminate all negative weights by making use of the fact that one can rewrite $-G_j^2(\underline{z})$ as $G_j^3(\underline{z}) - 1$ with another π gate G_j^3, for all j with $\alpha_j = -1$. By setting $G_j^3 := G_j^2$ for all j with $\alpha_j = 1$, we then have $\sum_{j=1}^{\hat{n}} \frac{\alpha_j}{\hat{k}} G_j^2(\underline{z}) = \frac{\sum_{j=1}^{\hat{n}} G_j^3(\underline{z}) - \hat{T}}{\hat{k}}$ for all $\underline{z} \in D$, with some $\hat{T} \in \{0, \ldots, \hat{n}\}$. As the next step, we approximate each π gate G^3 by a sum $\frac{1}{\ell} \sum_{i=1}^{\ell} G_{(i)}$ of ℓ threshold gates $G_{(i)}$. Thus, altogether we have constructed an approximation of the continuous function $g^{-1}(h(\underline{z}))$ by a weighted sum $\sum_{j=1}^{n'} \frac{G_j(\underline{z}) - T}{k}$ of threshold gates with a uniform value $\frac{1}{k}$ for all weights.[4] By expanding our technique from the proof of theorem 2, we can replace this simple sum of threshold gates by a single soft-WTA$_{n,k}^{g}$ gate applied to weighted sums in such a way that the resulting WTA circuit approximates the given function h.

We now describe the proof in detail. According to Leshno, Lin, Pinkus, and Schocken (1993) there exist $\tilde{n} \in \mathbb{N}$, $\tilde{\beta}_j \in \mathbb{R}$, and π gates G_j^1 for $j = 1, \ldots, \tilde{n}$ so that

$$\left| \sum_{j=1}^{\tilde{n}} \tilde{\beta}_j G_j^1(\underline{z}) - g^{-1}(h(\underline{z})) \right| < \frac{\tilde{\varepsilon}}{3} \qquad \text{for all } \underline{z} \in D,$$

where g^{-1} is the inverse of the restriction of the given function g to $[0, 1]$, and $\tilde{\varepsilon} > 0$ is chosen so that $|g(x) - g(y)| < \varepsilon$ for any $x, y \in \mathbb{R}$ with $|x - y| < \tilde{\varepsilon}$. We are relying here on the fact that $g^{-1} \circ h$ is continuous.

[4] One could, of course, also apply the universal approximation theorem directly to threshold gates (instead of π gates) in order to get an approximation of $g^{-1} \circ h$ by a weighted sum of threshold gates. But the elimination of negative weights would then give rise to an exception set S, as in theorem 3.

As the next step, we simplify the weights. It is obvious that there exist $\hat{k} \in \mathbb{N}$ and $\beta_1, \ldots, \beta_{\tilde{n}} \in \mathbb{Z}$ so that $\sum_{j=1}^{\tilde{n}} |\tilde{\beta}_j - \frac{\beta_j}{\hat{k}}| < \frac{\tilde{\varepsilon}}{3}$. We then have

$$\left| \sum_{j=1}^{\tilde{n}} \frac{\beta_j}{\hat{k}} G_j^1(\underline{z}) - g^{-1}(h(\underline{z})) \right| < \frac{2}{3}\tilde{\varepsilon} \qquad \text{for all } \underline{z} \in D.$$

We set $\hat{n} := \sum_{j=1}^{\tilde{n}} |\beta_j|$, and for all $j \in \{1, \ldots, \tilde{n}\}$ we create $|\beta_j|$ copies of the π-gate G_j^1. Let $G_1^2, \ldots, G_{\hat{n}}^2$ be the resulting sequence of π gates G_j^2. Then there exist $\alpha_j \in \{-1, 1\}$ for $j = 1, \ldots, \hat{n}$ so that

$$\left| \sum_{j=1}^{\hat{n}} \frac{\alpha_j}{\hat{k}} G_j^2(\underline{z}) - g^{-1}(h(\underline{z})) \right| < \frac{2}{3}\tilde{\varepsilon} \qquad \text{for all } \underline{z} \in D.$$

We now eliminate all negative weights from this weighted sum. More precisely, we show that there are π gates $G_1^3, \ldots, G_{\hat{n}}^3$ so that

$$\sum_{j=1}^{\hat{n}} \frac{\alpha_j}{\hat{k}} G_j^2(\underline{z}) = \frac{\sum_{j=1}^{\hat{n}} G_j^3(\underline{z}) - \hat{T}}{\hat{k}} \qquad \text{for all } \underline{z} \in D, \tag{4.1}$$

where $\hat{T}$ is the number of $j \in \{1, \ldots, \hat{n}\}$ with $\alpha_j = -1$. Consider some arbitrary $j \in \{1, \ldots, \hat{n}\}$ with $\alpha_j = -1$. Assume that G_j^2 is defined by $G_j^2(\underline{z}) = \pi(\underline{w} \cdot \underline{z} + w_0 + \frac{1}{2})$ for all $\underline{z} \in \mathbb{R}^m$, with parameters $\underline{w} \in \mathbb{R}^m$ and $w_0 \in \mathbb{R}$. Because of the properties of the function π, we have

$$-\pi\left(\underline{w} \cdot \underline{z} + w_0 + \frac{1}{2}\right) = -1 + \pi\left(-\underline{w} \cdot \underline{z} - w_0 + \frac{1}{2}\right)$$

for all $\underline{z} \in \mathbb{R}^m$. Indeed, if $|\underline{w} \cdot \underline{z} + w_0| \le \frac{1}{2}$ we have $-\pi(\underline{w} \cdot \underline{z} + w_0 + \frac{1}{2}) = -\underline{w} \cdot \underline{z} - w_0 - \frac{1}{2} = -1 + \pi(-\underline{w} \cdot \underline{z} - w_0 + \frac{1}{2})$. If $\underline{w} \cdot \underline{z} + w_0 < -\frac{1}{2}$ then $-\pi(\underline{w} \cdot \underline{z} + w_0 + \frac{1}{2}) = 0 = -1 + \pi(-\underline{w} \cdot \underline{z} - w_0 + \frac{1}{2})$, and if $\underline{w} \cdot \underline{z} + w_0 > \frac{1}{2}$ then $-\pi(\underline{w} \cdot \underline{z} + w_0 + \frac{1}{2}) = -1 = -1 + \pi(-\underline{w} \cdot \underline{z} - w_0 + \frac{1}{2})$. Thus, if we define the π gate G_j^3 by

$$G_j^3(\underline{z}) = \pi\left(-\underline{w} \cdot \underline{z} - w_0 + \frac{1}{2}\right) \qquad \text{for all } \underline{z} \in \mathbb{R}^m,$$

we have $-G_j^2(\underline{z}) = -1 + G_j^3(\underline{z})$ for all $\underline{z} \in \mathbb{R}^m$. Besides transforming all π gates G_j^2 with $\alpha_j = -1$ in this fashion, we set $G_j^3 = G_j^2$ for all $j \in \{1, \ldots, \hat{n}\}$ with $\alpha_j = 1$. Obviously equation 4.1 is satisfied with these definitions.

Since the activation function π can be approximated arbitrarily close by step functions, there exists for each π gate G^3 a sequence $G_{(1)}, \ldots, G_{(\ell)}$ of threshold gates so that

$$\left| G^3(\underline{z}) - \frac{\sum_{i=1}^{\ell} G_{(i)}(\underline{z})}{\ell} \right| < \frac{\tilde{\varepsilon} \cdot \hat{k}}{3 \cdot \hat{n}} \qquad \text{for all } \underline{z} \in \mathbb{R}^m.$$

By applying this transformation to all π-gates $G_1^3, \ldots, G_{\hat{n}}^3$, we arrive at a sequence $G_1, \ldots, G_{n'}$ of threshold gates with $n' := \hat{n} \cdot \ell$ so that one has for $k := \hat{k} \cdot \ell$ and $T := \hat{T} \cdot \ell$ that

$$\left| \frac{\sum_{j=1}^{n'} G_j(\underline{z}) - T}{k} - g^{-1}(h(\underline{z})) \right| < \tilde{\varepsilon} \qquad \text{for all } \underline{z} \in D.$$

According to the choice of $\tilde{\varepsilon}$, this implies that

$$\left| g\left(\frac{\sum_{j=1}^{n'} G_j(\underline{z}) - T}{k} \right) - h(\underline{z}) \right| < \varepsilon \qquad \text{for all } \underline{z} \in D.$$

By adding dummy threshold gates G_j that give constant output for all $\underline{z} \in D$, one can adjust n' and T so that $n' \geq 2k$ and $T = \frac{n'-1}{2}$, without changing the value of $\sum_{j=1}^{n'} G_j(\underline{z}) - T$ for any $\underline{z} \in D$.

It remains to show that

$$g\left(\frac{\sum_{j=1}^{n'} G_j(\underline{z}) - \frac{n'-1}{2}}{k} \right)$$

can be computed for all $\underline{z} \in D$, by some soft winner-take-all gate soft-WTA$_{n,k}^{g}$ applied to n weighted sums with positive coefficients. We set $n := n' + 1$. For $j = 1, \ldots, n'$ and $i = 1, \ldots, m$ let $w_i^j, \Theta^j \in \mathbb{R}$ be parameters so that

$$G_j(\underline{z}) = 1 \Leftrightarrow \sum_{i:\ w_i^j > 0} |w_i^j| z_i - \sum_{i:\ w_i^j < 0} |w_i^j| z_i \geq \Theta^j.$$

Then $\sum_{j=1}^{n'} G_j(\underline{z}) = |\{j \in \{1, \ldots, n'\}\colon S_{n'+1} \geq S_j\}|$ for $S_{n'+1} := \sum_{j=1}^{n'} \sum_{i:\ w_i^j > 0} |w_i^j| z_i$ and $S_j := \sum_{i:\ w_i^j < 0} |w_i^j| z_i + \Theta^j + \sum_{\ell \neq j} \sum_{i:\ w_i^\ell > 0} |w_i^\ell| z_i$. This holds because we have for every $j \in \{1, \ldots, n'\}$

$$S_{n'+1} \geq S_j \Leftrightarrow \sum_{i:\ w_i^j > 0} |w_i^j| z_i - \sum_{i:\ w_i^j < 0} |w_i^j| z_i \geq \Theta^j.$$

Since $n = n' + 1$ and therefore $\frac{n'-1}{2} + 1 = \frac{n}{2}$, the preceding implies that the nth output variable r_n of soft-WTA$_{n,k}^{g}$ applied to $S_1, \ldots, S_n$ outputs

$$g\left(\frac{|\{j \in \{1, \ldots, n\}\colon S_n \geq S_j\}| - \frac{n}{2}}{k} \right) = g\left(\frac{\sum_{j=1}^{n'} G_j(\underline{z}) - \frac{n'-1}{2}}{k} \right)$$

for all $\underline{z} \in D$. Note that all weights in the weighted sums $S_1, \ldots, S_n$ are positive.

5 Conclusions

We have established the first rigorous analytical results regarding the computational power of winner-take-all.

The lower-bound result of section 2 shows that the computational power of hard winner-take-all is already quite large, even if compared with the arguably most powerful gate commonly studied in circuit complexity theory: the threshold gate (also referred to a McCulloch-Pitts neuron or perceptron). Theorem 1 yields an optimal quadratic lower bound for computing hard winner-take-all on any feedforward circuit consisting of threshold gates. This implies that no circuit consisting of linearly many threshold gates with arbitrary (i.e., feedforward, lateral, and recurrent) connections can compute hard winner-take-all in sublinear time. Since approximate versions of winner-take-all can be computed very fast in linear-size analog VLSI chips (Lazzaro et al., 1989), this lower-bound result may be viewed as evidence for a possible gain in computational power that can be achieved by lateral connections in analog VLSI (and apparently also in cortical circuits).

It is well known (Minsky & Papert, 1969) that a single threshold gate is unable to compute certain important functions, whereas circuits of moderate (i.e., polynomial)-size consisting of two layers of threshold gates with polynomial size weights have remarkable computational power (Siu et al., 1995). We showed in section 3 that any such two-layer (i.e., one hidden layer) circuit can be simulated by a single competitive stage, applied to polynomially many weighted sums with positive integer weights of polynomial size.

In section 4, we analyzed the computational power of soft winner-take-all gates in the context of analog computation. It was shown that a single soft winner-take-all gate may serve as the only nonlinearity in a class of circuits that have universal computational power in the sense that they can approximate any continuous functions. In addition, we showed that this result is robust with regard to any continuous nonlinear warping of the output of such a soft winner-take-all gate, which is likely to occur in an analog implementation of soft winner-take-all in hardware or wetware. Furthermore, our novel universal approximators require only positive linear operations besides soft winner-take-all, thereby showing that in principle, no computational power is lost if inhibition is used exclusively for unspecific lateral inhibition in neural circuits, and no flexibility is lost if synaptic plasticity (i.e., "learning") is restricted to excitatory synapses.

This result appears to be of interest for understanding the function of biological neural systems, since typically only 15% of synapses in the cortex are inhibitory, and plasticity for those synapses is somewhat dubious.

Our somewhat surprising results regarding the computational power and universality of winner-take-all point to further opportunities for low-power analog VLSI chips, since winner-take-all can be implemented very efficiently in this technology. In particular, our theoretical results of section 4 predict that efficient implementations of soft winner-take-all will be useful in many contexts. Previous analog VLSI implementations of winner-take-all have primarily been used for special-purpose computational tasks. In contrast, our results show that a VLSI implementation of a single soft winner-take-all in combination with circuitry for computing weighted sums of the input variables yields devices with universal computational power for analog computation. A more qualitative account of the results of this article can be found in Maass, Wolfgang (1999).

The theoretical results of this article support the viability of an alternative style of neural circuit design, where complex multilayer perceptrons—feedforward circuits consisting of threshold gates or sigmoidal gates with positive and negative weights—are replaced by a single competitive stage applied to positive weighted sums. One may argue that this circuit design is more compatible with anatomical and physiological data from biological neural circuits.

Acknowledgments

I thank Moshe Abeles, Peter Auer, Rodney Douglas, Timmer Horiuchi, Giacomo Indiveri, and Shih-chii Liu for helpful discussions and the anonymous referees for helpful comments.

Research for this article was partially supported by the ESPRIT Working Group NeuroCOLT, No. 8556, and the Fonds zur Förderung der wissenschaftlichen Forschung (FWF), Austria, project P12153.

References

Abeles, M. (1991). *Corticonics: Neural circuits of the cerebral cortex*. Cambridge: Cambridge University Press.

Amari, S., & Arbib, M. (1977). Competition and cooperation in neural nets. In J. Metzler (Ed.), *Systems neuroscience* (pp. 119–165). San Diego: Academic Press.

Andreou, A. G., Boahen, K. A., Pouliquen, P. O, Pavasovic, A., Jenkins, R. E., & Strohbehn, K. (1991). Current-mode subthreshold MOS circuits for analog VLSI neural systems. *IEEE Trans. on Neural Networks*, *2*, 205–213.

Arbib, M. A. (1995). *The handbook of brain theory and neural networks*. Cambridge, MA: MIT Press .

Brajovic, V., & Kanade, T. (1998). Computational sensor for visual tracking with attention. *IEEE Journal of Solid State Circuits*, *33*, 8, 1199–1207.

Choi, J., & Sheu B. J. (1993). A high precision VLSI winner-take-all circuit for self-organizing neural networks. *IEEE J. Solid-State Circuits*, *28*, 576–584.

Coultrip, R., Granger R., & Lynch, G. (1992). A cortical model of winner-take-all competition via lateral inhibition. *Neural Networks*, *5*, 47–54.

DeWeerth, S. P., & Morris, T. G. (1994). Analog VLSI circuits for primitive sensory attention. *Proc. IEEE Int. Symp. Circuits and Systems*, *6*, 507–510.

DeYong, M., Findley, R., & Fields, C. (1992). The design, fabrication, and test of a new VLSI hybrid analog-digital neural processing element. *IEEE Trans. on*

Neural Networks, *3*, 363–374.

Elias, S. A., & Grossberg, S. (1975). Pattern formation, contrast control, and oscillations in the short term memory of shunting on-center off-surround networks. *Biol. Cybern.*, *20*, 69–98.

Fang, Y., Cohen, M., & Kincaid, M. (1996). Dynamics of a winner-take-all neural network. *Neural Networks*, *9*, 1141–1154.

Horiuchi, T. K., Morris, T. G., Koch, C., & DeWeerth, S. P. (1997). Analog VLSI circuits for attention-based visual tracking. In M. Mozer, M. Jordan, & T. Petsche (Eds.), *Advances in neural information processing systems* (pp. 706–712). Cambridge, MA: MIT Press.

Indiveri, G. (in press). Modeling selective attention using a neuromorphic analog-VLSI device. *Neural Computation*.

Lazzaro, J., Ryckebusch, S., Mahowald, M. A., & Mead, C. A. (1989). Winner-take-all networks of $O(n)$ complexity. In D. Touretzky (Ed.), *Advances in neural information processing systems, 1* (pp. 703–711). San Mateo, CA: Morgan Kaufmann.

Leshno, M., Lin, V., Pinkus, A., & Schocken, S. (1993). Multilayer feedforward networks with a nonpolynomial activation function can approximate any function. *Neural Networks*, *6*, 861–867.

Maass, Wolfgang. (1999). Neural computation with winner-take-all as the only nonlinear operation. *Advances in Neural Information Processing Systems 1999*, **12**. Cambridge: MIT Press.

Meador, J. L., & Hylander, P. D. (1994). Pulse coded winner-take-all networks. In M. E. Zaghloul, J. Meador, & R. W. Newcomb (Eds.), *Silicon implementation of pulse coded neural networks* (pp. 79–99). Boston: Kluwer Academic Publishers.

Minsky, M. C., & Papert, S. A. (1969). *Perceptrons*. Cambridge, MA: MIT Press.

Niebur, E., & Koch, C. (1998). Computational architectures for attention. In R. Parasuraman (Ed.), *The Attentive Brain* (pp. 163–186). Cambridge, MA: MIT Press.

Savage, J. E. (1998). *Models of computation: Exploring the power of computing*. Reading, MA: Addison-Wesley.

Shepherd, G. M. (Ed.). (1998). *The synaptic organization of the brain*. Oxford: Oxford University Press.

Siu, K.-Y., Roychowdhury, V., & Kailath, T. (1995). *Discrete neural computation: A theoretical foundation*. Englewood Cliffs, NJ: Prentice Hall.

Thorpe, S. J. (1990). Spike arrival times: A highly efficient coding scheme for neural networks. In R. Eckmiller, G. Hartmann, and G. Hauske (Eds.), *Parallel processing in neural systems and computers* (pp. 91–94). Amsterdam: Elsevier.

Urahama, K., & Nagao, T. (1995). *k*-winner-take-all circuit with $O(N)$ complexity. *IEEE Trans. on Neural Networks*, *6*, 776–778.

Yuille, A. L., & Grzywacz, N. M. (1989). A winner-take-all mechanism based on presynaptic inhibition. *Neural Computation*, *1*, 334–347.

Wegener, I. (1987). *The complexity of boolean functions*. New York: Wiley.

Received May 21, 1999; accepted November 15, 1999.

This article has been cited by:

1. Jürgen Schmidhuber. 2015. Deep learning in neural networks: An overview. *Neural Networks* **61**, 85-117. [CrossRef]
2. Jonathan Tapson, Greg Cohen, André van Schaik. 2014. ELM solutions for event-based systems. *Neurocomputing* . [CrossRef]
3. Ryotaro Kamimura. 2014. Improving visualisation and prediction performance of supervised self-organising map by modified contradiction resolution. *Connection Science* 1-28. [CrossRef]
4. Ryotaro Kamimura. 2014. Input information maximization for improving self-organizing maps. *Applied Intelligence* **41**, 421-438. [CrossRef]
5. Bastiaan J. Boom, Jiyin He, Simone Palazzo, Phoenix X. Huang, Cigdem Beyan, Hsiu-Mei Chou, Fang-Pang Lin, Concetto Spampinato, Robert B. Fisher. 2014. A research tool for long-term and continuous analysis of fish assemblage in coral-reefs using underwater camera footage. *Ecological Informatics* **23**, 83-97. [CrossRef]
6. Jonathan Binas, Ueli Rutishauser, Giacomo Indiveri, Michael Pfeiffer. 2014. Learning and stabilization of winner-take-all dynamics through interacting excitatory and inhibitory plasticity. *Frontiers in Computational Neuroscience* **8**. . [CrossRef]
7. F. Dietlein, W. Eschner. 2014. Inferring primary tumor sites from mutation spectra: a meta-analysis of histology-specific aberrations in cancer-derived cell lines. *Human Molecular Genetics* **23**, 1527-1537. [CrossRef]
8. A. J. Genot, T. Fujii, Y. Rondelez. 2013. Scaling down DNA circuits with competitive neural networks. *Journal of The Royal Society Interface* **10**, 20130212-20130212. [CrossRef]
9. Timothy GawneCortical Computations Using Relative Spike Timing 375-390. [CrossRef]
10. Elena Montañés, Jose Barranquero, Jorge Díez, Juan José del Coz. 2013. Enhancing directed binary trees for multi-class classification. *Information Sciences* **223**, 42-55. [CrossRef]
11. Anthony Genot, Teruo Fujii, Yannick Rondelez. 2012. Computing with Competition in Biochemical Networks. *Physical Review Letters* **109**. . [CrossRef]
12. Colin G. Johnson. 2012. Connotation in Computational Creativity. *Cognitive Computation* **4**, 280-291. [CrossRef]
13. Shuai Li, Jiguo Yu, Mingming Pan, Sanfeng Chen. 2012. Winner-take-all based on discrete-time dynamic feedback. *Applied Mathematics and Computation* . [CrossRef]
14. Ueli Rutishauser, Jean-Jacques Slotine, Rodney J. Douglas. 2012. Competition Through Selective Inhibitory Synchrony. *Neural Computation* **24**:8, 2033-2052. [Abstract] [Full Text] [PDF] [PDF Plus]
15. András Lőrincz, Zsolt Palotai, Gábor Szirtes. 2011. Sparse and silent coding in neural circuits. *Neurocomputing* . [CrossRef]
16. Ueli Rutishauser, Rodney J. Douglas, Jean-Jacques Slotine. 2011. Collective Stability of Networks of Winner-Take-All Circuits. *Neural Computation* **23**:3, 735-773. [Abstract] [Full Text] [PDF] [PDF Plus]
17. Qingshan Liu, Jinde Cao, Guanrong Chen. 2010. A Novel Recurrent Neural Network with Finite-Time Convergence for Linear Programming. *Neural Computation* **22**:11, 2962-2978. [Abstract] [Full Text] [PDF] [PDF Plus]
18. Kristian Ambrosch, Wilfried Kubinger. 2010. Accurate hardware-based stereo vision. *Computer Vision and Image Understanding* **114**, 1303-1316. [CrossRef]
19. Liisa Välikangas, Guje Sevón. 2010. Of managers, ideas and jesters, and the role of information technology. *The Journal of Strategic Information Systems* **19**, 145-153. [CrossRef]
20. Jun Wang. 2010. Analysis and Design of a k -Winners-Take-All Model With a Single State Variable and the Heaviside Step Activation Function. *IEEE Transactions on Neural Networks* **21**, 1496-1506. [CrossRef]
21. Qingshan Liu, Chuangyin Dang, Jinde Cao. 2010. A Novel Recurrent Neural Network With One Neuron and Finite-Time Convergence for k-Winners-Take-All Operation. *IEEE Transactions on Neural Networks* **21**, 1140-1148. [CrossRef]
22. Michael Pfeiffer, Bernhard Nessler, Rodney J. Douglas, Wolfgang Maass. 2010. Reward-Modulated Hebbian Learning of Decision Making. *Neural Computation* **22**:6, 1399-1444. [Abstract] [Full Text] [PDF] [PDF Plus]
23. Dietmar Plenz, Jeffery R. WickensThe Striatal Skeleton 99-112. [CrossRef]
24. Youshen Xia, Changyin Sun. 2009. A novel neural dynamical approach to convex quadratic program and its efficient applications. *Neural Networks* **22**, 1463-1470. [CrossRef]
25. Matthias Oster, Rodney Douglas, Shih-Chii Liu. 2009. Computation with Spikes in a Winner-Take-All Network. *Neural Computation* **21**:9, 2437-2465. [Abstract] [Full Text] [PDF] [PDF Plus]
26. Timothée Masquelier, Rudy Guyonneau, Simon J. Thorpe. 2009. Competitive STDP-Based Spike Pattern Learning. *Neural Computation* **21**:5, 1259-1276. [Abstract] [Full Text] [PDF] [PDF Plus]

27. Claudius Gros. 2009. Cognitive Computation with Autonomously Active Neural Networks: An Emerging Field. *Cognitive Computation* **1**, 77-90. [CrossRef]

28. Ueli Rutishauser, Rodney J. Douglas. 2009. State-Dependent Computation Using Coupled Recurrent Networks. *Neural Computation* **21**:2, 478-509. [Abstract] [Full Text] [PDF] [PDF Plus]

29. Xiaolin Hu, Jun Wang. 2008. An Improved Dual Neural Network for Solving a Class of Quadratic Programming Problems and Its k-Winners-Take-All Application. *IEEE Transactions on Neural Networks* **19**, 2022-2031. [CrossRef]

30. P AUER, H BURGSTEINER, W MAASS. 2008. A learning rule for very simple universal approximators consisting of a single layer of perceptrons#. *Neural Networks* **21**, 786-795. [CrossRef]

31. Timothy J. Gawne. 2008. Stimulus selection via differential response latencies in visual cortical area V4. *Neuroscience Letters* **435**, 198-203. [CrossRef]

32. Q LIU, J WANG. 2008. Two k-winners-take-all networks with discontinuous activation functions#. *Neural Networks* **21**, 406-413. [CrossRef]

33. Kristian Ambrosch, Wilfried Kubinger, Martin Humenberger, Andreas Steininger. 2008. Flexible Hardware-Based Stereo Matching. *EURASIP Journal on Embedded Systems* **2008**, 1-13. [CrossRef]

34. Stephen B. Furber, Gavin Brown, Joy Bose, John Michael Cumpstey, Peter Marshall, Jonathan L. Shapiro. 2007. Sparse Distributed Memory Using Rank-Order Neural Codes. *IEEE Transactions on Neural Networks* **18**, 648-659. [CrossRef]

35. A LAZAR, G PIPA, J TRIESCH. 2007. Fading memory and time series prediction in recurrent networks with different forms of plasticity. *Neural Networks* **20**, 312-322. [CrossRef]

36. G OU, Y MURPHEY. 2007. Multi-class pattern classification using neural networks. *Pattern Recognition* **40**, 4-18. [CrossRef]

37. Jeffery R. Wickens, Gordon W. Arbuthnott, Tomomi ShindouSimulation of GABA function in the basal ganglia: computational models of GABAergic mechanisms in basal ganglia function 313-329. [CrossRef]

38. W RICHARDS, H SEBASTIANSEUNG, G PICKARD. 2006. Neural voting machines. *Neural Networks* **19**, 1161-1167. [CrossRef]

39. Cengiz Günay, Anthony S. Maida. 2006. A stochastic population approach to the problem of stable recruitment hierarchies in spiking neural networks. *Biological Cybernetics* **94**, 33-45. [CrossRef]

40. Shubao Liu, Jun Wang. 2006. A Simplified Dual Neural Network for Quadratic Programming With Its KWTA Application. *IEEE Transactions on Neural Networks* **17**, 1500-1510. [CrossRef]

41. GODFRIED TOUSSAINT. 2005. GEOMETRIC PROXIMITY GRAPHS FOR IMPROVING NEAREST NEIGHBOR METHODS IN INSTANCE-BASED LEARNING AND DATA MINING. *International Journal of Computational Geometry & Applications* **15**, 101-150. [CrossRef]

42. S FURBER, W JOHNBAINBRIDGE, J MIKECUMPSTEY, S TEMPLE. 2004. Sparse distributed memory using -of- codes. *Neural Networks* **17**, 1437-1451. [CrossRef]

43. Rodney J. Douglas, Kevan A.C. Martin. 2004. NEURONAL CIRCUITS OF THE NEOCORTEX. *Annual Review of Neuroscience* **27**:10.1146/neuro.2004.27.issue-1, 419-451. [CrossRef]

44. B JAIN, F WYSOTZKI. 2004. Discrimination networks for maximum selection. *Neural Networks* **17**, 143-154. [CrossRef]

45. Jiří Šíma, Pekka Orponen. 2003. General-Purpose Computation with Neural Networks: A Survey of Complexity Theoretic Results. *Neural Computation* **15**:12, 2727-2778. [Abstract] [PDF] [PDF Plus]

46. Wolfgang Maass, Thomas Natschläger, Henry Markram. 2002. Real-Time Computing Without Stable States: A New Framework for Neural Computation Based on Perturbations. *Neural Computation* **14**:11, 2531-2560. [Abstract] [PDF] [PDF Plus]

47. Xiaohui Xie, Richard H. R. Hahnloser, H. Sebastian Seung. 2002. Selectively Grouping Neurons in Recurrent Networks of Lateral Inhibition. *Neural Computation* **14**:11, 2627-2646. [Abstract] [PDF] [PDF Plus]

48. Dezhe Jin, H. Seung. 2002. Fast computation with spikes in a recurrent neural network. *Physical Review E* **65**. . [CrossRef]