# Energy-Time Tradeoff in Recurrent Neural Nets

Jiří Šíma

**Abstract** In this chapter, we deal with the energy complexity of perceptron networks which has been inspired by the fact that the activity of neurons in the brain is quite sparse (with only about 1% of neurons firing). This complexity measure has recently been introduced for feedforward architectures (i.e., threshold circuits). We shortly survey the tradeoff results which relate the energy to other complexity measures such as the size and depth of threshold circuits. We generalize the energy complexity for recurrent architectures which counts the number of simultaneously active neurons at any time instant of a computation. We present our energy-time tradeoff result for the recurrent neural nets which are known to be computationally as powerful as the finite automata. In particular, we show the main ideas of simulating any deterministic finite automaton by a low-energy optimal-size neural network. In addition, we present a lower bound on the energy of such a simulation (within a certain range of time overhead) which implies that the energy demands in a fixed-size network increase exponentially with the frequency of presenting the input bits.

## 1 Energy Complexity—Motivations and Survey

According to biological studies on energy consumption by cortical computation which are based on electrophysiological recordings [6], the energy cost of a single spike is substantially higher than that of the no-spike rest state of a neuron. On the other hand, the energy supply to the brain is known to be limited, which suffices possibly to fewer than 1%, the number of neurons that can be substantially active concurrently, and hence the activity of neurons in the cortex is quite sparse. This is not in contradiction with the fMRI observations [2] that the oxygen consumption by a functional part of the brain (e.g. visual cortex) when engaged in appropriate tasks

Jiří Šíma
Institute of Computer Science, Academy of Sciences of the Czech Republic, P. O. Box 5, 18207 Prague 8, Czech Republic, e-mail: sima@cs.cas.cz

is not substantially higher as compared to the state when this region is not employed for its purpose. The reason is that the brain exhibit permanent, although sparse, activity, and thus the difference in energy consumption by a used vs. not used area of the brain is relatively small. In fact, this confirms the brain is quite effective in performing its tasks from the energy point of view.

In contrast to their biological counterparts, artificial neural circuits are frequently designed such that they do not take energy constraints into account. In fact, only the size (i.e., the number of gates) or the depth (i.e., the number of layers) of such networks are usually somehow optimized while computations have the property that, on average, approximately a half of units in the circuit fire (i.e. output a "1") during any computation. This fact has recently motivated the definition of a new complexity measure for *feedforward* perceptron networks (threshold circuits), the so-called *energy complexity* [17] which is the maximum number of units in the network which output 1, taken over all the possible inputs to the circuit.

Although the perceptron networks represent only a very rough abstract model of biological neural networks as compared to e.g. networks of spiking neurons, in a preliminary study of energy phenomenon, the stereotypical spikes can be simulated by outputs 1 from threshold gates, which makes the analysis technically more tractable. In addition, the first results have shown that the feedforward perceptron networks which are widely used in engineering applications have a surprisingly large computational power even if their energy complexity is restricted which means these circuits can exhibit only sparse activity. Minimizing the energy complexity requires a different approach to the circuit design which is potentially closer to the brain structures. In particular, different pathways in these circuits are activated for different clusters of inputs.

Furthermore, energy complexity of feedforward neural networks has been shown to be closely related by tradeoff results to other complexity measures such as the network size, the circuit depth, and the fan-in. In particular, a formula binding the energy $e$ and the *size* $s$ of any threshold circuit that computes a symmetric Boolean function $f$ with $n$ variables (i.e., the function value of $f$ depends only on the number of 1's in the input) was derived [21], which gives e.g. the tradeoff $e \log s \geq \log n$ for the parity function $f$. Or even an exponential lower bound $\exp(\Omega(n^{1-o(n)}))$ on the number of perceptrons in any feedforward network with constant number of layers (i.e., constant depth) computing the Boolean inner product of $2n$ variables (see Eq. 6) was proven when the energy supply is restricted by $e = n^{o(1)}$ [19].

There is also a close relationship between the energy complexity $e$ and the *depth* $d$ (i.e., parallel computational time) of feedforward perceptron network: Any threshold circuit of energy complexity $e$ and size $s$, computing a Boolean function $f$ can be transformed to another threshold circuit of depth $2e+1$ and size $2es+1$, computing $f$ [18]. This means that, within polynomial size, the energy complexity provides an upper bound on the depth of feedforward networks. Moreover, the tradeoff $e = O(n/\ell)$ between the energy $e$ and the *fan-in* $\ell$ (i.e., the maximum number of inputs to a single unit) of any threshold circuit computing the modulus function $\mathrm{MOD}_m$ of $n$ variables (i.e., $\mathrm{MOD}_m$ gives 0 iff the number of 1's in the input is divisible by $m$) was proven including an almost tight lower bound $e = \Omega((n-m)/\ell))$ [16].

Last but not least, energy complexity has found its important use in circuit complexity as a tool for proving the lower bounds [20].

## 2 Energy Complexity of Recurrent Networks—Chapter Outline

In this chapter, we study, for the first time, the energy complexity of *recurrent* neural (perceptron) networks which we define to be the maximum number of neurons outputting 1 at any time instant, taken over all possible computations. Clearly, this generalizes the energy complexity of threshold circuits (Sect. 1) to recurrent architectures as the energy of feedforward networks remains the same according to this definition.

It has been known for a long time that the computational power of binary-state recurrent networks corresponds to that of finite automata since the network of size $s$ units can reach only a finite number (at most $2^s$) different states [14]. A simple way of simulating a given deterministic finite automaton $A$ with $m$ states by a neural network $N$ of size $O(m)$ is to implement each of the $2m$ transitions of $A$ (having 0 and 1 transitions for each state) by a single unit in $N$ which checks whether the input bit agrees with the respective type of transition [8]. Clearly, this simple linear-size implementation of finite automata requires only a constant energy since the determinism of automaton ensures that only a single acceptance path is traversed through the state transition graph, that is, only one neuron fires on this path at any time instant.

Much effort had been given to reducing the size of neural automata [1, 3, 4, 15] and, indeed, neural networks of size $\Theta(\sqrt{m})$ implementing a given deterministic finite automaton with $m$ states were proposed and proven to be size-optimal [3, 4]. A natural question arises: What is the energy consumption when simulating finite automata by optimal-size neural networks? We answer this question in this chapter by showing the tradeoff between the energy and the time overhead of the simulation. In particular, we prove that an optimal-size neural network of $s = \Theta(\sqrt{m})$ units can be constructed to simulate a deterministic finite automaton with $m$ states using the energy $O(e)$ for any function $e$ such that $e = \Omega(\log s)$ and $e = O(s)$, while the time overhead for processing one input bit is $\tau = O(s/e)$. This means that the frequency of presenting the input bits can increase when more energy is supplied to the simulating network. For this purpose, we adapt the asymptotically optimal method of threshold circuit synthesis [7].

In addition, we also derive lower bounds on the energy consumption $e$ of a neural network of size $s$ simulating a finite automaton within the time overhead $\tau$ per one input bit, by using the technique due to Uchizawa and Takimoto [19] which is based on a communication complexity argument [5]. In particular, for less than sublogarithmic time overhead $\tau$ satisfying $\tau \log \tau = o(\log s)$, we obtain the lower bound $\log e = \Omega_\infty\left(\frac{1}{\tau}\log s\right)$ which implies $e \geq s^{c/\tau}$ for some constant $c > 0$ and for infinitely many $s$. Thus, the energy complexity in a fixed-size neural network increases exponentially with the frequency of presenting the input bits. For example, this means

that for constant time overhead $\tau = O(1)$, the energy of any simulation meets $e \geq s^{\delta}$ for some constant $\delta$ such that $0 < \delta < 1$, and for infinitely many $s$, which can be compared to the energy $e = O(s)$ consumed by our simulation. For $\tau = O(\log^{\alpha} s)$ where $0 < \alpha < 1$, any such simulation requires energy $e = \Omega_{\infty}\left(s^{\log\log s/\log^{\delta} s}\right)$ for any $\delta > \alpha$, while $e = O\left(s/\log^{\alpha} s\right)$ is sufficient for our implementation.

This chapter is organized as follows. After a brief review of the basic definitions regarding neural networks as finite automata in Sect. 3, the main result concerning a low-energy simulation of finite automata by neural nets is presented in Sect. 4 including the basic ideas of the proof. The lower bounds on the energy consumption of such neural automata are formulated and compared to the respective upper bounds in Sect. 5. A concluding summary is given in Sect. 6. A preliminary version of this chapter has appeared as an extended abstract [12] which was further expanded to journal paper [13] including complete proofs. This chapter is focused on motivations and a survey, providing a brief exposition of the main ideas of our results on the energy complexity of recurrent neural networks while complicated technical details are only sketched or even omitted.

## 3 Neural Finite Automata

In order to precisely present our results we first recall the formal definition of an (artificial) *neural network N* and then we will introduce its I/O protocol for implementing a finite automaton. The network consists of *s units (neurons, threshold gates)*, indexed as $V = \{1, \ldots, s\}$, where $s$ is called the network *size*. The units are connected into a directed graph representing the *architecture* of $N$, in which each edge $(i, j)$ leading from unit $i$ to $j$ is labeled with an integer *weight* $w(i, j)$. The absence of a connection within the architecture corresponds to a zero weight between the respective neurons, and vice versa.

In contrast to general *recurrent* networks, which have cyclic architectures, the architecture of a *feedforward* network (or a so-called *threshold circuit*) is an acyclic graph. Hence, units in a feedforward network can be grouped in a unique minimal way into a sequence of $d+1$ pairwise disjoint *layers* $\alpha_0, \ldots, \alpha_d \subseteq V$ so that neurons in any layer $\alpha_t$ are connected only to neurons in subsequent layers $\alpha_u$, $u > t$. Usually the zeroth, or *input layer* $\alpha_0$ consists of external inputs and is not counted in the number of layers and in the network size. The last, or *output layer* $\alpha_d$ is composed of output neurons. The number of layers $d$ excluding the input one is called the *depth* of threshold circuit.

The *computational dynamics* of (not necessarily feedforward) network $N$ determines for each unit $j \in V$ its binary *state (output)* $y_j^{(t)} \in \{0, 1\}$ at discrete time instants $t = 0, 1, 2, \ldots$. We say that neuron $j$ is *active (fires)* at time $t$ if $y_j^{(t)} = 1$, while $j$ is *passive* for $y_j^{(t)} = 0$. This establishes the *network state* $\mathbf{y}^{(t)} = (y_1^{(t)}, \ldots, y_s^{(t)}) \in \{0, 1\}^s$ at each discrete time instant $t \geq 0$. At the beginning of a computation, the neural network $N$ is placed in an *initial state* $\mathbf{y}^{(0)}$ which may also include an external

input. At discrete time instant $t \geq 0$, an *excitation* of any neuron $j \in V$ is defined as

$$\xi_j^{(t)} = \sum_{i=1}^{s} w(i,j) y_i^{(t)} - h(j) \tag{1}$$

including an integer *threshold* $h(j)$ local to unit $j$. At the next instant $t+1$, the neurons $j \in \alpha_{t+1}$ from a selected subset $\alpha_{t+1} \subseteq V$ update their states $y_j^{(t+1)} = H(\xi_j^{(t)})$ in parallel by applying the *Heaviside function* $H : \Re \longrightarrow \{0,1\}$ which is defined as

$$H(\xi) = \begin{cases} 1 & \text{for } \xi \geq 0 \\ 0 & \text{for } \xi < 0\,. \end{cases} \tag{2}$$

The remaining units $j \in V \setminus \alpha_{t+1}$ do not change their outputs, that is $y_j^{(t+1)} = y_j^{(t)}$ for $j \notin \alpha_{t+1}$. In this way, the new network state $\mathbf{y}^{(t+1)}$ at time $t+1$ is determined.

Without loss of efficiency [9], we implicitly assume *synchronous* computations. Thus, the sets $\alpha_t$ which define the computational dynamics of $N$ are predestined deterministically for each time instant $t$ (e.g. $\alpha_t = V$ for any $t \geq 1$ means fully parallel synchronous updates). Note that computations in feedforward networks proceed layer by layer from the input layer up to the output one (i.e. sets $\alpha_t$ naturally coincide with layers), which implement Boolean functions. We define the *energy complexity* of $N$ to be the maximum number of active units

$$\sum_{j=1}^{s} y_j^{(t)}$$

at any time instant $t \geq 0$, taken over all the computations of $N$.

The computational power of recurrent neural networks has been studied analogously to the traditional models of computations so that the networks are exploited as acceptors of formal languages $L \subseteq \{0,1\}^*$ over the binary alphabet. For the finite networks that are to recognize regular languages, the following I/O protocol has been used [1, 3, 4, 11, 15, 14]. A binary input word (string) $\mathbf{x} = x_1 \ldots x_n \in \{0,1\}^n$ of arbitrary length $n \geq 0$ is sequentially presented to the network bit by bit via an *input neuron* $\text{in} \in V$. The state of this unit is externally set (and clamped) to the respective input bits at prescribed time instants, regardless of any influence from the remaining neurons in the network, that is,

$$y_{\text{in}}^{(\tau(i-1))} = x_i \tag{3}$$

for $i = 1, \ldots, n$ where an integer parameter $\tau \geq 1$ is the *period* or *time overhead* for processing a single input bit. Then, an *output neuron* $\text{out} \in V$ signals at time $\tau n$ whether the input word belongs to underlying language $L$, that is,

$$y_{\text{out}}^{(\tau n)} = \begin{cases} 1 & \text{for } \mathbf{x} \in L \\ 0 & \text{for } \mathbf{x} \notin L\,. \end{cases} \tag{4}$$

As usual, we will describe the limiting behavior (rate of growth) of functions when the argument tends towards infinity in terms of simpler functions by using Landau or big O notation. Recall that for functions $f \geq 1$ and $g \geq 1$ defined for all natural numbers, notations $g = O(f)$ and $g = \Omega(f)$ mean that for some real constant $c > 0$ and for all but finitely many natural numbers $n$, $g(n) \leq c \cdot f(n)$ and $g(n) \geq c \cdot f(n)$, respectively. In addition, $g = \Theta(f)$ if $g = O(f)$ and $g = \Omega(f)$ simultaneously. Similarly, $g = o(f)$ denotes that for *every* real constant $c > 0$ and for all but finitely many natural numbers $n$, $g(n) \leq c \cdot f(n)$, while $g = \Omega_\infty(f)$ means that for some real constant $c > 0$ and for infinitely many natural numbers $n$, $g(n) \geq c \cdot f(n)$. Clearly, $g = o(f)$ iff $\lim_{n\to\infty} g(n)/f(n) = 0$ iff $g \neq \Omega_\infty(f)$.

## 4 A Low-Energy Implementation of Finite Automata by Optimal-Size Neural Nets

Our main result concerning a low-energy implementation of finite automata by optimal-size neural nets is formulated in the following theorem. We will below provide a short informal proof sketch which explains the main ideas of the low-energy construction while the complicated details are deferred to technical paper [12].

**Theorem 1.** *A given deterministic finite automaton $A$ with $m$ states can be simulated by a neural network $N$ of optimal size $s = \Theta(\sqrt{m})$ neurons with time overhead $\tau = O(s/e)$ per one input bit, using the energy $O(e)$, where $e$ is any function satisfying $e = \Omega(\log s)$ and $e = O(s)$.*

*Proof.* (Sketch) For the construction of an optimal-size neural network $N$ implementing a given deterministic finite automaton $A$ we employ the approach due to Horne and Hush [3] which is based on Lupanov's result [7]. Unlike Horne and Hush who used the statement of Lupanov's theorem as a "black box", we need to modify Lupanov's construction in order to optimize the energy consumption.

Thus, a set $Q$ of $m$ states of a given deterministic finite automaton $A$ is enumerated by integers $0, 1, \ldots, m-1$ so that each $q \in Q$ is encoded in binary using $p = \lceil \log_2 m \rceil + 1$ bits including one additional (e.g. the $p$th) bit which indicates the final states (i.e. its value is 1 just for the final states of $A$). Then, the respective transition function $\delta : Q \times \{0,1\} \longrightarrow Q$ of automaton $A$, producing its new state $q_{\text{new}} = \delta(q_{\text{old}}, x) \in Q$ from the old state $q_{\text{old}} \in Q$ and current input bit $x \in \{0,1\}$, can be viewed as a vector Boolean function $\mathbf{f}_\delta : \{0,1\}^{p+1} \longrightarrow \{0,1\}^p$ in terms of binary encoding of automaton's states.

Furthermore, "transition" function $\mathbf{f}_\delta$ is implemented by a four-layer neural network $C$ of asymptotically optimal size $\Theta(2^{p/2}) = \Theta(\sqrt{m})$ using the method of threshold circuit synthesis due to Lupanov [7]. Feedforward network $C$ computing the transition function $\delta$ of $A$ can then simply be transformed to a recurrent neural network $N$ of the same size $s = \Theta(\sqrt{m})$ simulating $A$ by adding the recurrent connections from the fourth layer to the first one, as it is schematically depicted in Fig. 1. In fact, the fourth output layer of $C$ having $p$ threshold gates is identified

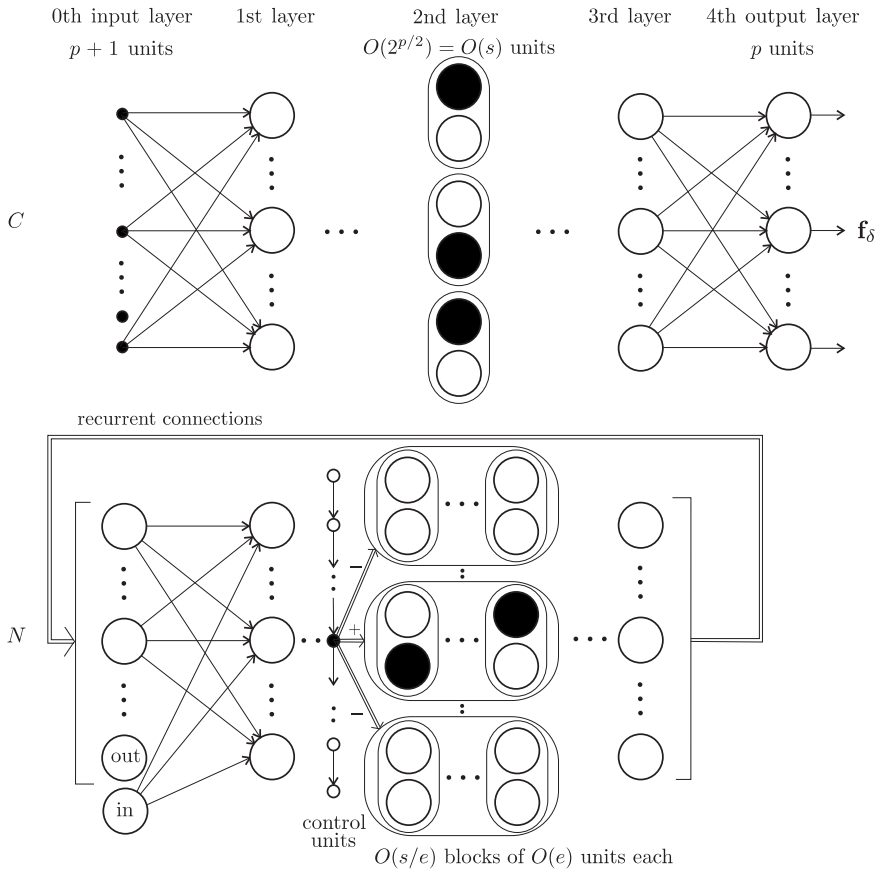


**Fig. 1** A schema of the transformation of threshold circuit $C$ (on top) implementing the "transition" function $\mathbf{f}_\delta$ of finite automaton $A$ into a low-energy neural network $N$ (at the bottom) simulating $A$.

with the $p$ units of the zeroth layer in $N$ which store the current state of $A$ while the remaining $(p+1)$th neuron in $\in V$ serves as an input to $A$ (cf. Eq. 3). The recurrent connections ensure that the binary code of automaton's old state is replaced by the new one. In addition, the $p$th neuron in the zeroth layer of $N$ representing the output neuron out $\in V$ signals whether the automaton is in an accepting state (cf. Eq. 4) because the $p$th bit in the binary code of states indicates the final states. Using this approach, a finite automaton can be implemented by an optimal-size neural net [3].

Unfortunately, the second layer of threshold circuit $C$ in Lupanov's construction [7] contains $\Theta(s)$ gates, half of which fire for any input to $C$, which results in an unacceptably high energy consumption $\Omega(s)$ although the energy demands of the remaining three layers are bounded by $O(p) = O(\log s)$. In particular, this second layer is composed of $O(2^{p/2})$ pairs of units such that exactly one neuron of each pair

fires, except for just $2p$ pairs of which both their neurons are active simultaneously. In fact, determining these $2p$ pairs is the core of evaluating the function $\mathbf{f}_\delta$ for a given input.

In order to achieve a low-energy implementation of $A$, this layer of $\Theta(s)$ neurons is properly partitioned into $O(s/e)$ blocks of $O(e)$ units each so that no pair of units is split into two blocks (see Fig. 1). Then, so-called *control units*, one for each block, are introduced which ensure that these blocks are updated successively one by one so that the energy consumption (i.e. the maximum number of simultaneously active neurons) is bounded by $O(e)$, while the time overhead for processing a single input bit possibly increases to $O(s/e)$. In particular, the control units create typically a path along which the control signal is propagated so that only one control unit on the path is active at each time instant. Each such a control unit releases an update of one associated block while the remaining blocks are blocked typically at the zero state (consuming no energy) apart from those $2p$ pairs of which both neurons fire simultaneously. In this way, the energy consumption of $C$ is reduced from $\Omega(s)$ to $O(e+p) = O(e+\log s) = O(e)$ as $e = \Omega(\log s)$. □

Theorem 1 provides an energy-time tradeoff for the size-optimal recurrent networks implementing finite automata, which generalizes the tradeoff results for threshold circuits (see Sect. 1). In particular, the time overhead $\tau$ which is necessary for processing a single input bit decreases when more energy $e$ is supplied to the network. For the full energy $e = \Omega(s)$ we obtain the constant-time overhead.

## 5 The Energy Lower Bound

In this section, we will show lower bounds on the energy complexity of neural networks implementing finite automata. For this purpose, we will employ the technique due to Uchizawa and Takimoto [19] which is based on communication complexity [5]. Assume that $f : \{0,1\}^n \times \{0,1\}^n \longrightarrow \{0,1\}$ is a Boolean function whose value $f(\mathbf{x},\mathbf{y})$ has to be computed by two players with unlimited computational power, each receiving only his/her part of the input $\mathbf{x} \in \{0,1\}^n$ and $\mathbf{y} \in \{0,1\}^n$, respectively, while they wish to exchange with each other the least possible number of bits. In particular, they communicate according to a randomized protocol additionally making use of the same public random bit string. For any error probability $\varepsilon$ satisfying $0 \leq \varepsilon < 1/2$, the *communication complexity* $R_\varepsilon(f)$ of function $f$ is defined to be the maximum number of bits needed to be exchanged for the best randomized protocol to make the two players compute the correct value of $f(\mathbf{x},\mathbf{y})$ with probability at least $1-\varepsilon$, for every input assignment $\mathbf{x}$ and $\mathbf{y}$.

It is well known [5] that almost all Boolean functions $f$ of $2n$ variables have large communication complexity

$$R_\varepsilon(f) = \Omega\left(n + \log\left(\frac{1}{2} - \varepsilon\right)\right) \tag{5}$$

for any error probability $\varepsilon$ such that $0 \leq \varepsilon < 1/2$. An example of a particular function that meets condition (5) is the Boolean inner product $IP_n : \{0,1\}^{2n} \longrightarrow \{0,1\}$, defined as

$$IP_n(x_1,\ldots,x_n,y_1,\ldots,y_n,) = \bigoplus_{i=1}^{n}(x_i \wedge y_i) \tag{6}$$

where $\oplus$ denotes the parity which gives 1 iff the number of satisfied conjunctions $x_i \wedge y_i$ (i.e. $x_i = y_i = 1$), for $i = 1,\ldots,n$, is odd.

On the other hand, Uchizawa and Takimoto [19] proved the upper bound on the communication complexity of Boolean function $f$ in terms of the size, depth, and energy complexity of a feedforward network computing $f$:

**Theorem 2 ([19]).** *If a Boolean function $f : \{0,1\}^{2n} \longrightarrow \{0,1\}$ can be computed by a threshold circuit of size S, depth d, and energy complexity E, then*

$$R_\varepsilon(f) = O\left((E+d)(\log n + (E+1)^d \log S\right) \tag{7}$$

*for error probability*

$$\varepsilon = \frac{1}{2} - \frac{1}{4S^{3(E+1)^d}}\,. \tag{8}$$

The lower and upper bounds on the communication complexity (5) and (7), respectively, are put together in the following lemma:

**Lemma 1 ([13]).** *Let $f : \{0,1\}^{2n} \longrightarrow \{0,1\}$ be a Boolean function of $2n$ variables whose communication complexity satisfies condition (5), which can be computed by a threshold circuit of size S, depth d, and energy complexity E such that $n = O(S)$ and $d = O(E)$. Then $n = O(E^{d+1} \log S)$.*

Now we will formulate the result providing the lower bound $e \geq s^{c/\tau}$ (for some constant $c > 0$ and for infinitely many $s$) on the energy complexity $e$ of a recurrent neural network of size $s$ neurons implementing a given finite automaton with time overhead $\tau$ such that $\tau^\tau = o(s)$. This means the lower bound is valid for less than sublogarithmic time overheads.

**Theorem 3.** *Let $\tau \log \tau = o(\log s)$. There exists a neural network of size s neurons simulating a finite automaton with time overhead $\tau$ per one input bit which needs energy e such that $\log e = \Omega_\infty\left(\frac{1}{\tau}\log s\right)$.*

*Proof.* Let $N$ be a neural network of size $s$ neurons simulating a finite automaton $A$ with time overhead $\tau$ per one input bit. The states of $A$ are represented by the $2^{s-1}$ states of $N$ (excluding the input neuron in) and the transition function of $A$ is computed by $N$ within $\tau$ time steps. Clearly, network $N$ can be "unwound" into a threshold circuit $C$ of depth $d = \tau$ and size $S = \tau s$ which implements the transition function of $A$ so that each layer is a copy of $N$ [10]. Thus, the states of neurons in the $i$th layer of $C$ coincide with the network state $\mathbf{y}^{(k\tau+i)}$ for $0 \leq i \leq \tau$, when the new state $\mathbf{y}^{((k+1)\tau)}$ of $A$ is produced from the old one $\mathbf{y}^{(k\tau)}$ including the current input bit. Hence, the energy complexity of $C$ is a $\tau$ multiple of the energy consumed by $N$, that is, $E = \tau e$.

As component $f_k : \{0,1\}^s \longrightarrow \{0,1\}$ (for $1 \leq k \leq s$) of the transition function defining $A$ can be arbitrary, there is a neural network $N$ simulating $A$ such that $f_k$ implemented by $C$ has large communication complexity satisfying condition (5). Moreover, $n = \lceil s/2 \rceil = O(S)$ and $d = \tau = O(E)$, which meets the remaining assumptions of Lemma 1. It follows that

$$\lceil s/2 \rceil = n = O\left(E^{d+1} \log S\right) = O\left((\tau e)^{t+1} \log \tau s\right) \tag{9}$$

according to Lemma 1. On the contrary, suppose that

$$\log e = o\left(\frac{\log s}{\tau}\right). \tag{10}$$

We will prove that $(\tau e)^{\tau+1} \log \tau s = o(s)$ which contradicts equation (9). For this purpose, it suffices to show that $\log((\tau e)^{\tau+1} \log \tau s) = o(\log s)$. This can be rewritten as $(\tau+1)\log\tau + (\tau+1)\log e + \log\log\tau + \log\log s = o(\log s)$ which follows from the assumption of the theorem and equation (10), completing the argument. □

Theorem 3 provides an energy-time tradeoff in recurrent neural networks from the lower-bound perspective as a counterpart to Theorem 1. In particular, the underlying formula relates the energy demands to the time overhead for processing a single input bit. It follows that the energy complexity in a fixed-size neural network increases exponentially with the frequency of presenting the input bits. In the following corollary we will formulate the lower bounds on energy complexity in terms of the network size for selected cases of sublogarithmic time overhead.

**Corollary 1.**
*1. If $\tau = O(1)$, then $e \geq s^{\delta}$ for some $\delta$ such that $0 < \delta < 1$ and for infinitely many $s$.*
*2. If $\tau = O(\log\log s)$, then $e = \Omega_\infty\left(s^{1/\log^{\delta} s}\right) = \Omega_\infty\left(2^{\log^{1-\delta} s}\right)$ for any $\delta$ such that $0 < \delta < 1$.*
*3. If $\tau = O(\log^{\alpha} s)$ for some $0 < \alpha < 1$, then $e = \Omega_\infty\left(s^{\log\log s/\log^{\delta} s}\right) = \Omega_\infty\left((\log s)^{\log^{1-\delta} s}\right)$ for any $\delta$ such that $\delta > \alpha$.*

*Proof.* 1. For $\tau = O(1)$, the assumption $\tau \log \tau = o(\log s)$ trivially holds and the proposition follows straightforwardly from Theorem 3.
2. For $\tau = O(\log\log s)$, there is $c_u > 0$ such that for all but finitely many $s$ we have $\tau\log\tau \leq c_u(\log\log s)\log\log\log s + (c_u \log c_u)\log\log s = o(\log s)$. According to Theorem 3, there is $c_\ell > 0$ such that $\log e \geq \frac{c_\ell \log s}{c_u \log\log s}$ for infinitely many $s$. On the contrary, suppose that $e = o\left(s^{1/\log^{\delta} s}\right)$ for some $\delta$ satisfying $0 < \delta < 1$, which implies $\log^{1-\delta} s \geq \frac{c_\ell \log s}{c_u \log\log s}$ leading to a contradiction $\frac{\log\log s}{\log^{\delta} s} \geq \frac{c_\ell}{c_u} > 0$.
3. If $\tau = O(\log^{\alpha} s)$ for some $0 < \alpha < 1$, then there is $c_u > 0$ such that for all but finitely many $s$ we have $\tau\log\tau \leq c_u(\log^{\alpha} s)\log\log^{\alpha} s + (c_u \log c_u)\log^{\alpha} s = o(\log s)$. According to Theorem 3, there is $c_\ell > 0$ such that $\log e \geq \frac{c_u}{c_\ell}\log^{1-\alpha} s$ for infinitely many $s$. On the contrary, suppose that $e = o\left(s^{\log\log s/\log^{\delta} s}\right)$ for some $\delta$ satisfying

$\delta > \alpha$, which implies $(\log\log s)\log^{1-\delta} s \geq \frac{c_u}{c_\ell}\log^{1-\alpha} s$ leading to a contradiction $\frac{\log\log s}{\log^{\delta-\alpha} s} \geq \frac{c_u}{c_\ell} > 0$. $\square$

We can compare the lower bounds on energy complexity of simulating the finite automata by neural nets presented in Corollary 1 to the respective upper bounds provided by Theorem 1. For the constant time overhead $\tau = O(1)$, the construction from Theorem 1 achieves the energy consumption of $e = O(s)$, while any simulation requires energy $e \geq s^\delta$ for some constant $\delta$ such that $0 < \delta < 1$ and for infinitely many $s$, according to Corollary 1. Similarly, for the time overhead of $\tau = O(\log^\alpha s)$ where $0 < \alpha < 1$, we have the upper bound of $e = O(s/\log^\alpha s)$ which compares to the lower bound of $e = \Omega_\infty\left(s^{\log\log s/\log^\delta s}\right)$. Clearly, there are still gaps between these lower and upper bounds, respectively, which need to be eliminated.

## 6 Conclusions

We have, for the first time, applied the energy complexity measure to recurrent neural nets. This measure has recently been introduced and studied for feedforward perceptron networks. The binary-state recurrent neural networks recognize exactly the regular languages so we have investigated their energy consumption of simulating the finite automata with the asymptotically optimal number of neurons. We have presented a low-energy implementation of finite automata by optimal-size neural nets with the tradeoff between the time overhead for processing one input bit and the energy varying from the logarithm to the full network size. It appears that the frequency of presenting the input bits can only increase when more energy is supplied to the network. We have also achieved lower bounds for the energy consumption of neural finite automata which are valid for less than sublogarithmic time overheads and are still not tight. It follows that the energy demands in a fixed-size network increase exponentially with the frequency of presenting the input bits. An open problem remains for further research whether these bounds can be improved. In addition, we have so far assumed the worst case energy consumption while the average case analysis would be another challenge.

**Acknowledgements** Author's research was done with institutional support RVO: 67985807 and partially supported by the grant of the Czech Science Foundation No. P202/12/G061.

## References

1. Alon, N., Dewdney, A.K., Ott, T.J.: Efficient simulation of finite automata by neural nets. J ACM **14**(2), 495–514 (1991)
2. Gafaniz, R., Sanches, J.M.: Neuronal electrical activity, energy consumption and mitochondrial ATP restoration dynamics: A physiological based model for FMRI. In: Proceedings of

the ISBI 2011 Eighth IEEE International Symposium on Biomedical Imaging: From Nano to Macro, pp. 341–344, IEEE (2011)
3. Horne, B.G., Hush, D.R.: Bounds on the complexity of recurrent neural network implementations of finite state machines. Neural Netw **9**(2), 243–252 (1996)
4. Indyk, P.: Optimal simulation of automata by neural nets. In: Mayr, E.W., Puech, C. (eds.) Proceedings of the STACS 1995 Twelfth Annual Symposium on Theoretical Aspects of Computer Science. LNCS, vol. 900, pp. 337–348 (1995)
5. Kushilevitz, E., Nisan, N.: Communication Complexity. Cambridge University Press, Cambridge (1997)
6. Lennie, P.: The cost of cortical computation. Curr Biol **13**(6), 493–497 (2003)
7. Lupanov, O.: On the synthesis of threshold circuits. Probl Kibern **26**, 109–140 (1973)
8. Minsky, M.: Computations: Finite and Infinite Machines. Prentice-Hall, Englewood Cliffs (1967)
9. Orponen, P.: Computing with truly asynchronous threshold logic networks. Theor Comput Sci **174**(1-2), 123–136 (1997)
10. Savage, J.E.: Computational work and time on finite machines. J ACM **19**(4), 660–674 (1972)
11. Siegelmann, H.T., Sontag, E.D.: Computational power of neural networks. J Comput Syst Sci **50**(1), 132–150 (1995)
12. Šíma, J.: A low-energy implementation of finite automata by optimal-size neural nets. In: Mladenov, V., Koprinkova-Hristova, P.D., Palm, G., Villa, A.E.P., Appollini, B., Kasabov, N. (eds.) Proceedings of the Twenty-Third International Conference on Artificial Neural Networks (ICANN 2013), LNCS, vol. 8131, pp. 114–121 (2013)
13. Šíma, J.: Energy complexity of recurrent neural networks. Neural Comput **26**(5) (2014)
14. Šíma, J., Orponen, P.: General-purpose computation with neural networks: A survey of complexity theoretic results. Neural Comput **15**(12), 2727–2778 (2003)
15. Šíma, J., Wiedermann, J.: Theory of neuromata. J ACM **45**(1), 155–178 (1998)
16. Suzuki, A., Uchizawa, K., Zhou, X.: Energy and fan-in of threshold circuits computing Mod functions. In: Ogihara, M., Tarui, J. (eds.) Proceedings of the TAMC 2011 Eight Annual Conference on Theory and Applications of Models of Computation. LNCS, vol. 6648, pp. 154–163 (2011)
17. Uchizawa, K., Douglas, R., Maass, W.: On the computational power of threshold circuits with sparse activity. Neural Comput **18**(12), 2994–3008 (2006)
18. Uchizawa, K., Nishizeki, T., Takimoto E.: Energy and depth of threshold circuits. Theor Comput Sci **411**(44-46), 3938–3946 (2010)
19. Uchizawa, K., Takimoto, E.: Exponential lower bounds on the size of constant-depth threshold circuits with small energy complexity. Theor Comput Sci **407**(1-3), 474–487 (2008)
20. Uchizawa, K., Takimoto, E.: Lower bounds for linear decision trees via an energy complexity argument. In: Murlak, F., Sankowski, P. (eds.): Proceedings of the MFCS 2011 Thirty-Sixth International Symposium on Mathematical Foundations of Computer Science. LNCS, vol. 6907, pp. 568-579 (2011)
21. Uchizawa, K., Takimoto, E., Nishizeki, T.: Size-energy tradeoffs for unate circuits computing symmetric Boolean functions. Theor Comput Sci **412**(8-10), 773–782 (2011)