Robust Implementation of Finite Automata by Recurrent RBF Networks

Michal Šorel[1] and Jiří Šíma[2⋆]

[1] Institute of Information Theory and Automation,
Academy of Sciences of the Czech Republic, Pod vodárenskou věží 4,
182 07 Prague 8, Czech Republic, michal.sorel@centrum.cz
[2] Institute of Computer Science, Academy of Sciences of the Czech Republic,
Pod vodárenskou věží 2, 182 07 Prague 8, Czech Republic, sima@cs.cas.cz
and Institute for Theoretical Computer Science (ITI), Charles University, Prague

Abstract. In this paper a recurrent network, which consists of $O(\sqrt{m}\log m)$ RBF (radial basis functions) units with maximum norm employing any activation function that has different values in at least two nonnegative points, is constructed so as to implement a given deterministic finite automaton with m states. The underlying simulation proves to be robust with respect to analog noise for a large class of smooth activation functions with a special type of inflexion.

1 RBF and Perceptron Neural Networks

The so-called *radial basis functions* (RBF) were first introduced in the context of interpolation problems [22] which are now one of the main research fields in numerical analysis. These functions were later exploited in the design of feedforward neural networks [3, 18, 21, 23] which are called *RBF networks* and represent an important alternative to the classical *perceptron networks* [17]. Unlike the perceptrons that classify unbounded halfspaces, the RBF units are localized around centers in the areas of bounded width. Hence, also the learning heuristics for RBF networks may be different and their application fields are usually complementary to that of perceptrons [9].

The computational properties of traditional perceptron networks seem to be well understood now. The computational power of the finite recurrent discrete-time neural networks with saturated-linear activation function increases with the Kolmogorov complexity of real weight parameters [2]. For example, such networks with integer weights (corresponding to binary-state networks with threshold units) are equivalent to finite automata [14], while those with rational weights can simulate arbitrary Turing machines [12, 24]. In addition, finer descriptive measures were introduced for the so-called *neuromata* which are binary-state neural networks implementing finite automata. A size-optimal neuromaton with $\Theta(\sqrt{m})$ perceptron units can be constructed as to simulate a given deterministic finite automaton with m states [11, 12]. Similarly, a regular language described by

⋆ Research supported by GA AS CR Grant B2030007.

a regular expression of length d can be recognized by a size-optimal neuromaton with $\Theta(d)$ perceptrons [26]. These neuromaton constructions were generalized for analog-state recurrent neural nets with a large class of sigmoidal activation functions so that the respective simulations are stable in a sense that potentially infinite computations are robust with respect to the analog noise [15, 20, 25].

One motivation for analog implementations of neuromata issues from real-world applications in which partial prior knowledge (i.e. several automaton transitions) is known which should be refined and completed by training data. However, most of the learning heuristics (e.g. gradient methods) operate in an analog state space and hence, the analog neuromaton implementation can serve as an initial network configuration for the underlying learning procedure. The resulting transition rules may be extracted after the neuromaton is adapted to training data. This approach has widely been applied [4, 5, 7, 16, 19, 20, 27, 28], especially for second-order neural nets which are also related to recurrent RBF networks [6]. The analog implementations of neuromata offer other advantages besides their use in gradient-based training algorithms; they also permit analog VLSI implementation, the foundations necessary for the universal approximation theories, the interpretation of outputs as posteriori probability estimates, etc. [20]

Unlike the perceptron networks, the computational capabilities of recurrent RBF networks have not been fully analyzed yet. Partial theoretical results along this direction have been achieved for hybrid recurrent networks consisting of both the RBF and perceptron layers which exhibited successful experimental results for inductive inference of regular grammars [6, 8].

This paper investigates the issue of implementing deterministic finite automata by recurrent RBF networks. In Section 2, the recurrent RBF network of size $O(\sqrt{m \log m})$ units with maximum norm employing any activation function that has different values in at least two nonnegative points, is constructed which simulates a given deterministic finite automaton with m states. Thus the recurrent RBF networks prove to be at least as powerful as finite automata and hence, they can be called the *RBF neuromata*. In Section 3, the underlying simulation is shown to be stable for more practical imprecise analog representations of binary states for a large class of smooth activation functions with a special type of inflexion which includes the most common Gaussian activation function. Finally, some open problems are mentioned in Section 4.

2 Construction of RBF Neuromata

We will first specify the model of a *recurrent RBF network*. The network consists of s *RBF units* indexed as $1, \ldots, s$, which are connected into a generally cyclic oriented graph called *architecture* where each edge (i, j) leading from unit i to j is labeled with real parameter c_{ji}. Denote by $\boldsymbol{c}_j = (c_{ji_1}, \ldots, c_{ji_{n_j}}) \in \mathbb{R}^{n_j}$ the *center* of RBF unit j where $i_1, \ldots, i_{n_j}$ are all the n_j incoming units incident to j. Each RBF unit j is also associated with its real positive *width* parameter $b_j > 0$.

The computational dynamics of the recurrent RBF network determines the evolution of the *network state* $\boldsymbol{y}^{(t)} = (y_1^{(t)}, \ldots, y_s^{(t)}) \in \mathbb{R}^s$ for discrete time instants $t = 0, 1 \ldots$ as follows. At the beginning of the computation, the network is placed in an *initial state* $\boldsymbol{y}^{(0)} \in \mathbb{R}^s$. At discrete time $t \geq 0$, the *excitation* $\xi_j^{(t)} = \|\boldsymbol{x}_j^{(t)} - \boldsymbol{c}_j\|/b_j \geq 0$ of unit j is proportional to the distance between the corresponding *input vector* $\boldsymbol{x}_j^{(t)} = (y_{ji_1}^{(t)}, \ldots, y_{ji_{n_j}}^{(t)}) \in \mathbb{R}^{n_j}$ and its center $\boldsymbol{c_j}$, where the *maximum norm* $\|\boldsymbol{x}\| = \|\boldsymbol{x}\|_\infty = \max_{i=1,\ldots,n} |x_i|$ for $\boldsymbol{x} = (x_1, \ldots, x_n) \in \mathbb{R}^n$ will be exploited in our construction. At the next time instant $t+1$, an *activation function* $\sigma : \mathbb{R} \longrightarrow \mathbb{R}$ is applied to $\xi_j^{(t)}$ only for the RBF units $j \in \alpha_{t+1}$ from a given subset α_{t+1} in order to determine their new *states (outputs)* $y_j^{(t+1)} = \sigma(\xi_j^{(t)})$ in parallel whereas the remaining ones do not change their states, i.e. $y_j^{(t+1)} = y_j^{(t)}$ for $j \notin \alpha_{t+1}$. A typical example of the activation function is the *Gaussian function* $\sigma_G(\xi) = e^{-\xi^2}$.

For the RBF neuromaton recognizing a language $L \subseteq \{0,1\}^\star$ over binary alphabet an input/output protocol is defined as follows. For a general activation function producing real outputs an *analog representation* $r : \{0,1\} \longrightarrow \mathbb{R}$ such that $r(0) \neq r(1)$, is introduced to encode the binary values 0 and 1 by the real states $r(0)$ and $r(1)$ of RBF units, respectively. The analog representation will also be used for Boolean vectors, i.e. $\boldsymbol{r}(z_1, \ldots, z_n) = (r(z_1), \ldots, r(z_n))$ for $\boldsymbol{z} \in \{0,1\}^n$, $n \geq 1$. Thus a binary input string $\boldsymbol{w} = w_1 \ldots w_k \in \{0,1\}^k$, $k \geq 0$ is presented bit after bit to the RBF neuromaton by means of an *input unit* j_{inp} whose state $y_{j_{inp}}^{(\tau(i-1))} = r(w_i)$ is externally set (and e.g. clamped), regardless of the RBF network computational dynamics, to the respective input-bit representation $r(w_i)$ at time instant $\tau(i-1)$ where an integer constant $\tau \geq 1$ is the *time overhead* for processing a single input bit. Then an *output unit* j_{out} signals at time τk whether the input word $\boldsymbol{w}$ belongs to language L, i.e. $y_{j_{out}}^{(\tau k)} = r(1)$ for $\boldsymbol{w} \in L$ and $y_{j_{out}}^{(\tau k)} = r(0)$ for $\boldsymbol{w} \notin L$. Similar definitions of language recognition by perceptron neuromata appeared in [1, 12, 24, 26].

In the following lemma the implementation of a universal NAND gate over multiple literals by the RBF unit with an analog representation of Boolean values is presented which is used for the RBF neuromaton construction in the subsequent theorem.

Lemma 1. *Assume a real function σ is defined in at least two different non-negative points $0 \leq \xi_1 < \xi_2$ so that $\sigma(\xi_1) \neq \sigma(\xi_2)$. Let $n \geq 1$ be an integer and $I \subseteq \{1, \ldots, n\}$. Denote by $g : \{0,1\}^n \longrightarrow \{0,1\}$ a Boolean function defined as $g(z_1, \ldots, z_n) = NAND(\tilde{z}_1, \ldots, \tilde{z}_n)$ where $\tilde{z}_i = z_i$ for $i \in I$ and $\tilde{z}_i = \bar{z}_i$ for $i \notin I$ are literals. Then the n-input RBF unit with the center $\boldsymbol{c} = (c_1, \ldots, c_n)$ such that $c_i = c$ for $i \in I$ and $c_i = \bar{c}$ for $i \notin I$ and width $b > 0$ where*

$$c = \frac{\xi_2\sigma(\xi_2) - \xi_1\sigma(\xi_1)}{\xi_2 - \xi_1}\,, \quad \bar{c} = \frac{\xi_2\sigma(\xi_1) - \xi_1\sigma(\xi_2)}{\xi_2 - \xi_1}\,, \quad b = \frac{|\sigma(\xi_2) - \sigma(\xi_1)|}{\xi_2 - \xi_1}\,, \tag{1}$$

employing the maximum norm and activation function σ, computes g within the analog representation $r(0) = \sigma(\xi_1)$ and $r(1) = \sigma(\xi_2)$, i.e. for any $\boldsymbol{z} \in \{0,1\}^n$

$$\sigma\left(\frac{\|\boldsymbol{r}(\boldsymbol{z}) - \boldsymbol{c}\|}{b}\right) = r(g(\boldsymbol{z}))\,. \tag{2}$$

Proof. It follows from (1) that

$$|\sigma(\xi_1) - \bar{c}| = |\sigma(\xi_2) - c| = b\xi_1 < b\xi_2 = |\sigma(\xi_1) - c| = |\sigma(\xi_2) - \bar{c}| \,. \tag{3}$$

Further denote by $\boldsymbol{a} = (a_1, \ldots, a_n) \in \{0,1\}^n$ a Boolean vector so that $a_i = 1$ for $i \in I$ and $a_i = 0$ for $i \notin I$ which is the only input that yields $g(\boldsymbol{a}) = 0$. Hence, for any $1 \le i \le n$ either $a_i = 1$ and $c_i = c$, or $a_i = 0$ and $c_i = \bar{c}$. Thus for $\boldsymbol{z} = \boldsymbol{a}$ it holds $\sigma(\|\boldsymbol{r}(\boldsymbol{a}) - \mathbf{c}\|/b) = \sigma(b\xi_1/b) = \sigma(\xi_1) = r(0)$ according to (3). On the other hand, for any $\boldsymbol{z} \in \{0,1\}^n$, $\boldsymbol{z} \neq \boldsymbol{a}$ there exists $1 \le i \le n$ such that either $z_i = 1$ and $c_i = \bar{c}$, or $z_i = 0$ and $c_i = c$ which gives $\sigma(\|\boldsymbol{r}(\boldsymbol{z}) - \mathbf{c}\|/b) = \sigma(b\xi_2/b) = \sigma(\xi_2) = r(1)$ due to (3). □

Theorem 2. *Let σ be a real function that assigns different values $\sigma(\xi_1) \neq \sigma(\xi_2)$ to at least two nonnegative points $0 \le \xi_1 < \xi_2$ which establish the analog representation $r(0) = \sigma(\xi_1)$ and $r(1) = \sigma(\xi_2)$. Given a deterministic finite automaton with m states, an equivalent RBF neuromaton of size $O(\sqrt{m \log m})$ units with maximum norm and activation function σ, can be constructed that recognizes the same regular language for the analog representation r within the time overhead $\tau = 3$ per each input bit.*

Proof. Set Q of m states of a given deterministic finite automaton can be enumerated so that each $q \in Q$ is binary encoded within $\lceil \log_2 m \rceil$ bits. The underlying transition function $\delta : Q \times \{0,1\} \longrightarrow Q$ producing the new automaton state $q_{new} = \delta(q_{old}, z) \in Q$ from an old one $q_{old} \in Q$ and a given input bit $z \in \{0,1\}$ can be viewed as a vector Boolean function $\boldsymbol{f}_\delta : \{0,1\}^{\lceil \log_2 m \rceil + 1} \longrightarrow \{0,1\}^{\lceil \log_2 m \rceil}$ in terms of binary encoding of states. It can be shown by techniques from [10] that any vector Boolean function $\boldsymbol{f} : \{0,1\}^n \longrightarrow \{0,1\}^p$ can be implemented by an optimal-size Boolean circuit of depth 3 with $\Theta(\sqrt{p2^n})$ gates computing NAND functions over multiple literals. According to Lemma 1 the underlying NAND gates can be realized by RBF units with the maximum norm and activation function σ within the analog representation r. Hence, the automaton transition function $\boldsymbol{f}_\delta$ can be implemented by a three-layered feedforward RBF network of size $O(\sqrt{m \log m})$ units which is used for constructing the respective RBF neuromaton as follows.

The inputs to the first layer are taken from the third-layer units storing current neuromaton state within analog representation r and from a single external input unit j_{inp} which is updated by encoding the next bit of input string $\boldsymbol{w} = w_1 \ldots w_k$ once every $\tau = 3$ time instants. In the beginning, the third-layer units encode the initial automaton state $q_0 \in Q$ and $y^{(0)}_{j_{inp}} = r(w_1)$ represents the first input bit. Set α_t of RBF units which update their states at time instant $1 \le t \le \tau k$ corresponds to the ℓth layer for $t = \tau(i-1) + \ell$ when the ith bit w_i of the input string is being processed. This means that the binary encoding of new state is computed in the period of 3 parallel updates in which layer after layer takes its part. Finally, at time instant τk, the input string $\boldsymbol{w}$ of length k is read and accepted iff the third-layer units encode an accepting automaton state which is indicated by an additional bit in the binary encoding of automaton states corresponding to the output unit j_{out} whose state $y^{(\tau k)}_{j_{out}} = r(1)$ iff $\boldsymbol{w}$ is accepted. This completes the desired construction of the RBF neuromaton. □

3 Robust RBF Neuromata

In Section 2 the RBF neuromaton is implemented only for precise analog representations of binary values assuming noise-free operations over real numbers. However, the infinite precision of real number representation is far from being realistic and an analog noise (e.g. rounding errors) could deteriorate the RBF neuromaton computations. Therefore, in this section the underlying construction is analyzed to find out conditions under which the respective RBF neuromaton is robust when the analog representation of binary values 0 and 1 is released to be within specific disjoint small intervals around real states $r(0)$ and $r(1)$, respectively.

For the robust RBF neuromaton construction it suffices to show that the implementation of NAND gates by RBF units according to Lemma 1 preserves the prescribed precision of analog representation. More precisely, the underlying computation of Boolean function g by an RBF unit (with activation function σ defined in a sufficiently large domain) is said to be *ε-stable*, $0 < \varepsilon < (r(0) + r(1))/2$, if for every binary input $\boldsymbol{z} \in \{0,1\}^n$ and for any $\boldsymbol{x} \in \mathbb{R}^n$ of its analog representations within precision ε, i.e. $\max_{i=1,\dots,n} |r(z_i) - x_i| = \|\boldsymbol{r}(\boldsymbol{z}) - \boldsymbol{x}\| \leq \varepsilon$, the output precision of RBF unit is preserved:

$$\left| \sigma\left(\frac{\|\boldsymbol{r}(\boldsymbol{z}) - \boldsymbol{c}\|}{b}\right) - \sigma\left(\frac{\|\boldsymbol{x} - \boldsymbol{c}\|}{b}\right) \right| \leq \varepsilon \,. \tag{4}$$

A large class of differentiable activation functions for which Lemma 1 provides a robust implementation of NAND gates by RBF units will be characterized.

Theorem 3. *Let σ be a real function that is differentiable within sufficiently large neighborhoods of two positive points $0 < \xi_1 < \xi_2$ where $\sigma(\xi_1) \neq \sigma(\xi_2)$. Then the computation of g by the RBF unit in Lemma 1 is ε-stable ($\varepsilon \leq \xi_1 b$) if*

$$\forall \xi \in D(\xi_1, \xi_2, \varepsilon) = \left[\xi_1 - \frac{\varepsilon}{b}, \xi_1 + \frac{\varepsilon}{b}\right] \cup \left[\xi_2 - \frac{\varepsilon}{b}, \xi_2 + \frac{\varepsilon}{b}\right] \quad |\sigma'(\xi)| \leq b \tag{5}$$

and it is not ε-stable if $|\sigma'(\xi)| > b$ for all $\xi \in D(\xi_1, \xi_2, \varepsilon)$.

Proof. First suppose that σ satisfies (5) and let $\boldsymbol{z} \in \{0,1\}^n$ and $\boldsymbol{x} \in \mathbb{R}^n$ be such that $\|\boldsymbol{r}(\boldsymbol{z}) - \boldsymbol{x}\| \leq \varepsilon$. It follows from the triangular inequality that

$$\|\boldsymbol{r}(\boldsymbol{z}) - \boldsymbol{c}\| - \|\boldsymbol{r}(\boldsymbol{z}) - \boldsymbol{x}\| \leq \|\boldsymbol{x} - \boldsymbol{c}\| \leq \|\boldsymbol{r}(\boldsymbol{z}) - \boldsymbol{c}\| + \|\boldsymbol{x} - \boldsymbol{r}(\boldsymbol{z})\| \,. \tag{6}$$

Then divide (6) by $b > 0$ and recall from the proof of lemma 1 that $\|\boldsymbol{r}(\boldsymbol{z}) - \boldsymbol{c}\|/b$ equals either ξ_1 for $\boldsymbol{z} = \boldsymbol{a}$, or ξ_2 for $\boldsymbol{z} \neq \boldsymbol{a}$ which together with $\|\boldsymbol{r}(\boldsymbol{z}) - \boldsymbol{x}\| \leq \varepsilon$ implies $\|\boldsymbol{x} - \boldsymbol{c}\|/b \in D(\xi_1, \xi_2, \varepsilon)$. Therefore by the mean value theorem there exists $\zeta \in D(\xi_1, \xi_2, \varepsilon)$ such that

$$\sigma\left(\frac{\|\boldsymbol{r}(\boldsymbol{z}) - \boldsymbol{c}\|}{b}\right) - \sigma\left(\frac{\|\boldsymbol{x} - \boldsymbol{c}\|}{b}\right) = \sigma'(\zeta)\left(\frac{\|\boldsymbol{r}(\boldsymbol{z}) - \boldsymbol{c}\|}{b} - \frac{\|\boldsymbol{x} - \boldsymbol{c}\|}{b}\right) \tag{7}$$

that yields the desired (4) since $|\sigma'(\zeta)| \leq b$ from (5) and $\big|\|\boldsymbol{r}(\boldsymbol{z}) - \boldsymbol{c}\| - \|\boldsymbol{x} - \boldsymbol{c}\|\big| \leq \varepsilon$ according to (6).

On the other hand, suppose $|\sigma'(\xi)| > b$ for all $\xi \in D(\xi_1, \xi_2, \varepsilon)$ and let $\boldsymbol{z} \in \{0,1\}^n$. Choose $\boldsymbol{x}_0 = \boldsymbol{r}(\boldsymbol{z}) + \varepsilon(\boldsymbol{r}(\boldsymbol{z}) - \boldsymbol{c})/\|\boldsymbol{r}(\boldsymbol{z}) - \boldsymbol{c}\|$. Clearly $\|\boldsymbol{r}(\boldsymbol{z}) - \boldsymbol{x}_0\| = \varepsilon$ and thus $\boldsymbol{x}_0 \in \mathbb{R}^n$ is an analog representation of $\boldsymbol{z}$ within precision ε. In addition, $\|\boldsymbol{r}(\boldsymbol{z}) - \boldsymbol{c}\| - \|\boldsymbol{x}_0 - \boldsymbol{c}\| = \varepsilon$ which implies $\|\boldsymbol{x}_0 - \boldsymbol{c}\|/b \in D(\xi_1, \xi_2, \varepsilon)$ since $\|\boldsymbol{r}(\boldsymbol{z}) - \boldsymbol{c}\|/b \in \{\xi_1, \xi_2\}$. Therefore again by the mean value theorem (7) there exists $\zeta_0 \in D(\xi_1, \xi_2, \varepsilon)$ such that

$$\left| \sigma\left(\frac{\|\boldsymbol{r}(\boldsymbol{z}) - \boldsymbol{c}\|}{b}\right) - \sigma\left(\frac{\|\boldsymbol{x}_0 - \boldsymbol{c}\|}{b}\right) \right| = |\sigma'(\zeta_0)| \frac{\varepsilon}{b} > \varepsilon \tag{8}$$

which means that the computation of g is not ε-stable according to (4). □

Condition (5) that characterizes the differentiable activation functions for which the presented RBF neuromaton is robust includes parameter b which is defined in (1) by the function values again. In the following lemma a useful characterization of smooth functions, which implies (5), is formulated. Namely, these functions must have a special type of an inflexion point at which either the increasing function changes from convex to concave part, or conversely for the decreasing one.

Lemma 4. *Let σ be a real function that has the continuous derivative within $(\alpha, \beta) \subseteq [0, \infty)$. Assume that there exist $\xi_0 \in (\alpha, \beta)$ with $\sigma'(\xi_0) \neq 0$ and $\Delta > 0$ such that σ is either convex within $(\xi_0 - \Delta, \xi_0)$ and concave within $(\xi_0, \xi_0 + \Delta)$ if $\sigma'(\xi_0) > 0$, or concave within $(\xi_0 - \Delta, \xi_0)$ and convex within $(\xi_0, \xi_0 + \Delta)$ when $\sigma'(\xi_0) < 0$. Then there exist $\alpha < \xi_1 < \xi_2 < \beta$ and $\varepsilon > 0$ such that $\sigma(\xi_1) \neq \sigma(\xi_2)$ and for any $\xi \in D(\xi_1, \xi_2, \varepsilon)$ $|\sigma'(\xi)| \leq b = |\sigma(\xi_2) - \sigma(\xi_1)|/(\xi_2 - \xi_1)$ (which corresponds to condition (5) in Theorem 3).*

Proof. Suppose that $\sigma'(\xi_0) > 0$ while for $\sigma'(\xi_0) < 0$ the argument is similar. Since σ' is continuous there exists $0 < \Delta' < \Delta$ such that $\sigma'(\xi) > 0$ for all $\xi \in (\xi_0 - \Delta', \xi_0 + \Delta')$. According to the assumption, σ' is nondecreasing within $(\xi_0 - \Delta', \xi_0)$ and nonincreasing within $(\xi_0, \xi_0 + \Delta')$, and hence σ' has its maximum at ξ_0. Without loss of generality there exists $\gamma > 0$ such that either $\sigma'(\xi) = \sigma'(\xi_0)$ for all $\xi \in (\xi_0 - \gamma, \xi_0 + \gamma)$ (shift ξ_0 appropriately), or $\sigma'(\xi) \neq \sigma'(\xi_0)$ for all $\xi \in (\xi_0 - \gamma, \xi_0) \cup (\xi_0, \xi_0 + \gamma)$. In the former case set $\xi_1 = \xi_0 - \gamma/2$, $\xi_2 = \xi_0 + \gamma/2$, and $\varepsilon = \gamma/4$ for which $\sigma(\xi_1) \neq \sigma(\xi_2)$ and $\sigma'(\xi) = (\sigma(\xi_2) - \sigma(\xi_1))/(\xi_2 - \xi_1)$, i.e. $|\sigma'(\xi)| = b$ for any $\xi \in D(\xi_1, \xi_2, \varepsilon)$ since σ' is positive and constant there. In the latter case choose $u > 0$ within $\sigma'((\xi_0 - \Delta', \xi_0)) \cap \sigma'((\xi_0, \xi_0 + \Delta'))$ which is a nontrivial interval excluding $\sigma'(\xi_0)$. Then there exists $\xi_1 \in (\xi_0 - \Delta', \xi_0)$ and $\xi_2 \in (\xi_0, \xi_0 + \Delta')$ such that $\sigma'(\xi_1) = \sigma'(\xi_2) = u$ due to σ' is continuous. Again $\sigma(\xi_1) \neq \sigma(\xi_2)$ because σ is increasing within $(\xi_0 - \Delta', \xi_0 + \Delta')$. Moreover $\sigma'(\xi) > u$ for all $\xi \in (\xi_1, \xi_2)$ and therefore

$$u(\xi_2 - \xi_1) < \int_{\xi_1}^{\xi_2} \sigma'(\xi) d\xi = \sigma(\xi_2) - \sigma(\xi_1)\,, \tag{9}$$

which implies $0 < \sigma'(\xi_1) = \sigma'(\xi_2) = u < (\sigma(\xi_2) - \sigma(\xi_1))/(\xi_2 - \xi_1)$ and thus there exists $\varepsilon > 0$ such that $|\sigma'(\xi)| < b$ for any $\xi \in D(\xi_1, \xi_2, \varepsilon)$ because σ' is continuous. □

For example, if σ has the first three derivatives and there exists an inflexion point $\xi_0 > 0$ such that $\sigma''(\xi_0) = 0$ and $\sigma'(\xi_0)\sigma'''(\xi_0) < 0$, then the type of this inflexion corresponds to that described by the assumption of Lemma 4. This can be used for checking that the most common Gaussian activation function $\sigma_G(\xi) = e^{-\xi^2}$ can be employed for the robust RBF neuromaton implementation according to Theorem 2. Namely, $\xi_0 = 1/\sqrt{2}$ is the required inflexion point, i.e. $\sigma''_G(\xi_0) = 0$ and $\sigma'_G(\xi_0)\sigma'''_G(\xi_0) < 0$. In addition, it also follows from Theorem 3 that $\xi_1 = \sqrt{\ln(4/3)}$, $\xi_2 = \sqrt{\ln 4}$, i.e. $r(0) = 3/4$, $r(1) = 1/4$ are valid parameters for the corresponding 0.2-stable RBF neuromaton computations.

4 Conclusions and Open Problems

In the present paper, the recurrent RBF network with maximum norm has been exploited for implementing a given deterministic finite automaton within a relatively small network size. For the precise analog representation of binary values a corresponding simulation works for almost any activation function (Section 2). In addition, for a large class of differentiable activation functions, which are characterized in detail, this implementation has proved to be robust with respect to the analog noise in a sense that it preserves a prescribed precision of the underlying representation (Section 3). It follows that the recurrent RBF networks are efficient and reliable analog computational devices that are at least as powerful as finite automata and thus comparable to perceptron networks [15, 20, 25].

It would be interesting to generalize the construction of the RBF neuromaton for general norms including the Euclidean one. Another challenge for further research is to prove the upper bounds on the computational power of analog recurrent RBF networks, e.g. to show that they are able to simulate Turing machines as it is known for analog perceptron networks [12, 13, 24].

References

1. Alon, N., Dewdney, A. K., Ott, T. J. Efficient simulation of finite automata by neural nets. *Journal of the ACM*, **38**, 495–514, 1991.
2. Balcázar, J. L., Gavaldà, R., Siegelmann, H. T. Computational power of neural networks: A characterization in terms of Kolmogorov complexity. *IEEE Transactions of Information Theory*, **43**, 1175–1183, 1997.
3. Broomhead, D. S., Lowe, D. Multivariable functional interpolation and adaptive networks. *Complex Systems*, **2**, 321–355, 1988.
4. Das, S., Mozer, M. C. A unified gradient-descent/clustering architecture for finite state machine induction. In J. Cowan, G. Tesauro, and J. Alspector, editors, *Neural Information Processing Systems*, **6**, 19–26, 1994.
5. Frasconi, P., Gori, M., Maggini, M., Soda, G. A unified approach for integrating explicit knowledge and learning by example in recurrent networks. In *Proceedings of the IEEE International Joint Conference on Neural Networks IJCNN'91*, Seattle, vol. **I**, 881–916, IEEE Press, New York, 1991.
6. Frasconi, P., Gori, M., Maggini, M., Soda, G. Representation of finite state automata in recurrent radial basis function networks. *Machine Learning*, **23**, 5–32, 1996.

7. Giles, C. L., Miller, C. B., Chen, D., Chen, H. H., Sun, G. Z., Lee, Y. C. Learning and extracting finite state automata with second-order recurrent neural networks. *Neural Computation*, **4**, 393–405, 1992.
8. Gori, M., Maggini, M., Soda, G. Inductive inference with recurrent radial basis function networks. In *Proceedings of the International Conference on Artificial Neural Networks ICANN'94*, Sorrento, Italy, 238–241, Springer-Verlag, 1994.
9. Haykin, S. *Neural Networks: A Comprehensive Foundation.* Prentice-Hall, Upper Saddle River, NJ, 2nd edition, 1999.
10. Horne, B. G., Hush, D. R. On the node complexity of neural networks. *Neural Networks*, **7**, 1413–1426, 1994.
11. Horne, B. G., Hush, D. R. Bounds on the complexity of recurrent neural network implementations of finite state machines. *Neural Networks*, **9**, 243–252, 1996.
12. Indyk, P. Optimal simulation of automata by neural nets. In *Proceedings of the Twelfth Annual Symposium on Theoretical Aspects of Computer Science STACS'95*, vol. **900** of LNCS, 337–348, Springer-Verlag, Berlin, 1995.
13. Kilian, J., Siegelmann, H. T. The dynamic universality of sigmoidal neural networks. *Information and Computation*, **128**, 48–56, 1996.
14. Kleene, S. C. Representation of Events in Nerve Nets and Finite Automata. In C. E. Shannon and J. McCarthy, editors, *Automata Studies*, vol. **34** of *Annals of Mathematics Studies*, 3–41, Princeton University Press, NJ, 1956.
15. Maass, W., Orponen, P. On the effect of analog noise in discrete-time analog computations. *Neural Computation*, **10**, 1071–1095, 1998.
16. Manolios, P., Fanelli, R. First-order recurrent neural networks and deterministic finite state automata. *Neural Computation*, **6**, 1155–1173, 1994.
17. Minsky, M. L., Papert, S. A. *Perceptrons.* MIT Press, Cambridge, MA, 1969.
18. Moody, J. E., Darken, C. J. Fast learning in networks of locally-tuned processing units. *Neural Computation*, **1**, 281–294, 1989.
19. Omlin, C. W., Giles, C. L. Training second-order recurrent neural networks using hints. In D. Sleeman and P. Edwards, editors, *Proceedings of the Ninth International Conference on Machine Learning*, 363–368, San Mateo, CA, Morgan Kaufman Publishers, 1992.
20. Omlin, C. W., Giles, C. L. Constructing deterministic finite-state automata in recurrent neural networks. *Journal of the ACM*, **43**, 937–972, 1996.
21. Poggio, T., Girosi, F. Networks for approximation and learning. In *Proceedings of the IEEE*, **78**, 1481-1497, 1990.
22. Powell, M. J. D. Radial basis functions for multivariable interpolation: A review. In J. C. Mason and M. G. Cox, editors, *Proceedings of the IMA Conference on Algorithms for the Approximation of Functions and Data*, RMCS, Shrivenham, UK, 143–167, Oxford Science Publications, 1985.
23. Renals, S. Radial basis function network for speech pattern classification. *Electronics Letters*, **25**, 437–439, 1989.
24. Siegelmann, H. T., Sontag, E. D. Computational power of neural networks. *Journal of Computer System Science*, **50**, 132–150, 1995.
25. Šíma, J. Analog stable simulation of discrete neural networks. *Neural Network World*, **7**, 679–686, 1997.
26. Šíma, J., Wiedermann, J. Theory of neuromata. *Journal of the ACM*, **45**, 155–178, 1998.
27. Tiňo, P., Šajda, J. Learning and extracting initial mealy automata with a modular neural network model. *Neural Computation*, **7**, 822–844, 1995.
28. Zeng, Z., Goodman, R., Smyth, P. Learning finite state machines with self-clustering recurrent networks, *Neural Computation*, **5**, 976–990, 1993.