3096

On the Non-Learnability of a Single Spiking Neuron

Jiří Šíma

Institute of Computer Science,

Academy of Sciences of the Czech Republic,

P. O. Box 5, 18207 Prague 8, Czech Republic, *sima@cs.cas.cz*

Jiří Sgall

Mathematical Institute,

Academy of Sciences of the Czech Republic,

Žitná 25, 11567 Prague 1, Czech Republic, *sgall@math.cas.cz*

Abstract

We study the computational complexity of training a single spiking neuron $\mathcal{N}$ with binary coded inputs and output which, in addition to adaptive weights and a threshold, has adjustable synaptic delays. A synchronization technique is introduced so that the results concerning the non-learnability of spiking neurons with binary delays are generalized to arbitrary real-valued delays. In particular, the consistency problem for $\mathcal{N}$ with programmable weights, a threshold, and delays, and its approximation version are proven to be NP-complete. It follows that the spiking neurons with arbitrary synaptic delays are not properly PAC-learnable and do not allow robust learning unless $RP = NP$. In addition, the representation problem for $\mathcal{N}$, i.e., a question whether an n-variable Boolean function given in DNF (or as a disjunction of $O(n)$ threshold gates) can be computed by a spiking neuron, is shown to be $coNP$-hard.

1 A Spiking Neuron with Synaptic Delays

Neural networks establish an important class of learning models that are widely applied in practical applications to solving artificial intelligence tasks (Haykin, 1999). The prominent position among neural network models has recently been occupied by networks of spiking neurons (Maass & Bishop, 1999). As compared to the traditional perceptron unit (Rosenblatt, 1958) the spiking neuron represents a biologically more plausible model in which the times that pulses need to travel through particular synapses, called delays, are taken into account (Maass, 1997). It is known that the synaptic delays are tuned in biological neural systems through a variety of mechanisms (Gerstner & Kistler, 2002).

We consider a simplified model of a spiking neuron with adjustable weights, a threshold, and delays where pulses are implemented by a step function, rather than a smooth function of a similar shape. This makes easy silicon implementation in pulsed VLSI possible (Maass & Bishop, 1999). In addition, the spiking neuron under consideration is used to compute Boolean functions and thus binary coding is assumed both for the inputs and the output. The computational power of this model was analyzed by Schmitt (1998) while its learning complexity was studied by Maass & Schmitt (1999).

Formally, a *spiking neuron* $\mathcal{N}$ has n *inputs* receiving binary values $x_1, \ldots, x_n \in \{0,1\}$; each input i ($1 \leq i \leq n$) is associated with a real synaptic *weight* $w_i \in \mathbf{R}$ and nonnegative *delay* $d_i \in \mathbf{R}_0^+$, and the last parameter is the output *threshold* $\theta \in \mathbf{R}$. The input value $x_i = 1$ is presented in the form of a unit-length rectangular *pulse (spike)* of height $|w_i|$ (for $w_i < 0$ upside down). This pulse travels through ith synapse in continuous time producing a synaptic time delay d_i. Denote $D_i = [d_i, d_i+1)$ the time interval of unit length during which $\mathcal{N}$ is influenced by the spike from input i. Taking the delay into account, a *postsynaptic potential* $x_i(t) \in \mathbf{R}$ at ith input to $\mathcal{N}$ at continuous time $t \geq 0$ can be expressed as

$$x_i(t) = \begin{cases} w_i & \text{for } x_i = 1 \text{ and } t \in D_i = [d_i, d_i + 1) \\ 0 & \text{otherwise.} \end{cases} \tag{1.1}$$

The *(membrane) potential* of $\mathcal{N}$ at time instant $t \geq 0$ is then determined as the sum of current postsynaptic potentials:

$$\xi(t) = \sum_{i=1}^{n} x_i(t) \,. \tag{1.2}$$

Neuron $\mathcal{N}$ fires as soon as potential $\xi(t)$ reaches threshold θ. Thus, $\mathcal{N}$ outputs $y = 1$ if $\xi(t) \geq \theta$ for some $t \geq 0$ and $\mathcal{N}$ outputs $y = 0$ if $\xi(t) < \theta$ for all $t \geq 0$. Thus, for given weights $w_1, \ldots, w_n$, delays $d_1, \ldots, d_n$, and threshold θ, neuron $\mathcal{N}$ implements Boolean function $y_{\mathcal{N}} : \{0,1\}^n \longrightarrow \{0,1\}$ defined above. Also denote by $\mathcal{F}_{\mathcal{N}}$ the class of all Boolean functions computable by spiking neurons. Note that $\mathcal{N}$ coincides with the classical perceptron when all synaptic delays are zero.

We give several results concerning computational complexity of training a single spiking neuron $\mathcal{N}$ with programmable weights, a threshold, and delays. In

Section 2, the so-called consistency problem is shown to be NP-complete for $\mathcal{N}$ with arbitrary real-valued delays. This generalizes the previous NP-completeness proof that was valid only for binary (or bounded discrete) delays, and solves an open problem (Maass & Schmitt 1999; Šíma, 2003). Assuming $RP \neq NP$, it follows that the spiking neurons with binary coded inputs and outputs are not properly PAC-learnable and also that they do not allow robust learning. This might be a little surprising in comparison to classical perceptrons: perceptrons are not properly PAC-learnable for binary weights (Pitt & Valiant, 1988) but they become learnable when real-valued weights are allowed. Thus, in the learning complexity of neurons, the delays play a different role than the weights and the threshold. In addition, only the unit weights and bounded delays are exploited in our proof and the weights together with the threshold need not be modifiable. Hence, our non-learnability result holds also for the spiking neurons restricted to bounded real-valued delays and/or unit weights and/or a fixed threshold. Finally, the representation problem for spiking neurons is proven to be $coNP$-hard in Section 3.

A preliminary version of this article (Šíma, 2003) considered perceptrons with delays having analog input values while the results presented now are valid for spiking neurons with binary coded inputs.

2 A Single Spiking Neuron Is Not Learnable

The computational complexity of training a spiking neuron can be analyzed by using the *consistency (loading) problem* (Judd, 1990) which is the problem of finding the neuron parameters for a given training task so that the function computed by neuron is perfectly consistent with all training data. Thus, a *training set* contains m *training examples*, each composed of n-dimensional input $\mathbf{x}_k$ from $\{0,1\}^n$ labeled with the desired scalar output value b_k from $\{0,1\}$ corresponding to *negative* and *positive examples*:

$$T = \{(\mathbf{x}_k; b_k) \mid \mathbf{x}_k \in \{0,1\}^n,\, b_k \in \{0,1\},\, k = 1, \ldots, m\} \,. \qquad (2.1)$$

The decision version for the consistency problem is formulated as follows:

Consistency Problem for Spiking Neuron (CPSN)

Instance: A training set T for spiking neuron $\mathcal{N}$ having n inputs.

Question: Are there real-valued weights $w_1, \ldots, w_n \in \mathbf{R}$, threshold $\theta \in \mathbf{R}$, and nonnegative delays $d_1, \ldots, d_n \in \mathbf{R}_0^+$ for $\mathcal{N}$ such that $y_{\mathcal{N}}(\mathbf{x}) = b$ for every training example $(\mathbf{x}; b) \in T$?

The consistency problem is related to the *PAC (Probably Approximately Correct)* learning model (Valiant, 1984) which is defined as follows. The example oracle $EX(f, D)$ for a Boolean *target function* $f : \{0,1\}^n \longrightarrow \{0,1\}$ with respect to distribution $D : \{0,1\}^n \longrightarrow [0,1]$ returns the training example $(\mathbf{x}; f(\mathbf{x}))$ where $\mathbf{x}$ is drawn from $\{0,1\}^n$ according to distribution D. Denote by $f \Delta g = \{\mathbf{x} \in \{0,1\}^n \mid f(\mathbf{x}) \neq g(\mathbf{x})\}$ the symmetric difference between f and g. The spiking neuron is *properly PAC-learnable* if there is a learning algorithm L such that for any target function $f \in \mathcal{F}_{\mathcal{N}}$ and for any distribution D, given $n \geq 1$, accuracy $\varepsilon > 0$, and confidence $\delta > 0$ as input, L has access to oracle $EX(f, D)$, runs in time polynomial in n, $1/\varepsilon$, $1/\delta$, and produces weights $w_1, \ldots, w_n$, threshold θ, and delays $d_1, \ldots, d_n$ of spiking neuron $\mathcal{N}$ which with probability at least $1 - \delta$ satisfies

$$\sum_{x \in f \Delta y_{\mathcal{N}}} D(\mathbf{x}) < \varepsilon \,. \qquad (2.2)$$

An efficient algorithm for the consistency problem is required within the proper PAC learning framework (Blumer, Ehrenfeucht, Haussler, & Warmuth, 1989). In addition, the VC-dimension of the problem has to be polynomial; this requirement is satisfied by the common neural network models (Anthony & Bartlett, 1999; Roychowdhury, Siu, & Orlitsky, 1994; Vidyasagar, 1997), in particular, the VC-dimension of spiking neuron $\mathcal{N}$ with n inputs is $\Theta(n \log n)$

(Maass & Schmitt, 1999). These results on polynomiality of the VC-dimension encouraged proposals of several learning heuristics for networks of spiking neurons, e.g., *spike-propagation* (Bohte, Kok, & La Poutré, 2000). On the other hand, *NP*-hardness of the consistency problem implies that the neuron is not properly PAC-learnable under generally accepted complexity-theoretic assumption $RP \neq NP$ (Pitt & Valiant, 1988). An almost exhaustive list of such *NP*-hardness results for feedforward perceptron networks including single units was presented by Šíma (2002).

For ordinary perceptrons with *zero delays*, i.e., $d_i = 0$ for $i = 1, \ldots, n$, the consistency problem is solvable in polynomial time by linear programming although this problem restricted to binary weights is *NP*-complete (Pitt & Valiant, 1988). However, already for *binary delays* $d_i \in \{0, 1\}$ the consistency problem becomes *NP*-complete, even for spiking neurons having fixed weights (Maass & Schmitt, 1999). This implies that neuron $\mathcal{N}$ with binary delays is not properly PAC-learnable unless $RP = NP$. The result generalizes also to *bounded discrete delay* values $d_i \in \{0, 1, \ldots, k\}$ for fixed $k \geq 2$. For the spiking neurons with unbounded delays, however, *NP*-hardness of the consistency problem was listed among open problems (Maass & Schmitt, 1999; Šíma, 2003).

In this section we prove that the consistency problem is *NP*-complete for a single spiking neuron $\mathcal{N}$ with *arbitrary real-valued delays*, which answers the previous open question. The result appears at first sight a little surprising since, for example, the consistency problem for the perceptron with zero delays becomes polynomial-time solvable when we go from binary to real-valued weights. This shows that the delays have different impact on the learning complexity of neurons than the weights and the threshold.

Furthermore, our *NP*-completeness result holds even in a restricted model with bounded delays, unit weights, and a fixed threshold—in fact, it holds for any intermediate model between this restricted model and the general model stated above. This follows immediately from a natural robustness of the combinatorial proof which uses sets of training examples which, on one hand, if representable, then they can be represented by neurons in the restricted class, and on the other hand, if not representable, then cannot be represented even by the general spiking neurons. Thus our results are valid for a wide variety of models and may be more applicable to realistic biological systems.

For the proof a *synchronization technique* is introduced whose main idea can be described as follows. Let $A \subseteq \{1, \ldots, n\}$ be a subset of inputs and let $(\sum_{i \in A} \mathbf{e}_i \, ; \, 1) \in T$ be a consistent positive example written as 0-1 linear combinations of basis vectors $\mathbf{e}_i \in \{0,1\}^n$, that is all the items of $\mathbf{e}_i$ are zero but its ith component equals 1. The simultaneous consistency of negative examples $(\sum_{i \in B} \mathbf{e}_i \, ; \, 0) \in T$ for every $B \subseteq A$ such that $|B| = |A| - 1$ implies $\bigcap_{i \in A} D_i \neq \emptyset$. In this way it can be ensured that $\mathcal{N}$ is simultaneously influenced by the spikes from inputs A, which is then exploited for the synchronization of the input spikes.

Theorem 1 *The consistency problem for spiking neuron (CPSN) is* NP*-complete.*

In addition, this holds also for the model of spiking neurons with any combination of the following restrictions: (i) *bounded delays, when the values d_i are restricted to any real interval strictly longer than 1;* (ii) *unit weights, that is $w_i \in \{0,1\}$ (or any set containing $\{0,1\}$); and* (iii) *a fixed threshold $\theta \geq 3$.*

Proof: The fact that CPSN belongs to *NP* has already been stated by Maass & Schmitt (1999). In order to achieve the *NP*-hardness result, the following variant of the *set splitting* problem which is known to be *NP*-complete (Garey & Johnson, 1979) will be reduced to CPSN in polynomial time.

3Set-Splitting Problem (3SSP)

Instance: A finite set $S = \{1, \ldots, s\}$ of s elements and a collection C of subsets of S such that $|c| = 3$ for all $c \in C$.

Question: Is there a partition of S into two disjoint subsets S_1 and S_2, i.e., $S = S_1 \cup S_2$ and $S_1 \cap S_2 = \emptyset$, such that all $c \in C$ satisfy $c \not\subseteq S_1$ and $c \not\subseteq S_2$?

The 3SSP problem was also used for proving the result restricted to binary delays (Maass & Schmitt, 1999). The synchronization technique generalizes the proof to arbitrary delays.

Given a 3SSP instance (S, C), we construct a training set T for spiking neuron $\mathcal{N}$ with $n = 6(s+1)$ inputs. The inputs are indexed $x_{p,i}$ for $p = 0, 1, \ldots, s$, $i = 1, \ldots, 6$. One 6-tuple of inputs $x_{p,i}$ corresponds to each element $p \in S$; in addition the first six inputs $x_{0,i}$ are used for overall synchronization. The same index notation is used for corresponding weights $w_{p,i}$, delays $d_{p,i}$, time intervals $D_{p,i} = [d_{p,i}, d_{p,i}+1)$, and basis vectors $\mathbf{e}_{p,i} \in \{0,1\}^n$, for all pairs of $p = 0, 1, \ldots, s$, $i = 1, \ldots, 6$.

The training set T, written in terms of the basis vectors, is as follows:

$$
\begin{aligned}
(\mathbf{e}_{0,i} + \mathbf{e}_{0,j}\,;\,0) &\qquad \text{for } 1 \le i < j \le 6\,, && (2.3)\\
(\mathbf{e}_{0,i} + \mathbf{e}_{0,j} + \mathbf{e}_{0,k}\,;\,1) &\qquad \text{for } 1 \le i < j < k \le 6\,, && (2.4)\\
\left.\begin{aligned}(\mathbf{e}_{p,1} + \mathbf{e}_{p,2} + \mathbf{e}_{p,3}\,;\,1)\\ (\mathbf{e}_{p,4} + \mathbf{e}_{p,5} + \mathbf{e}_{p,6}\,;\,1)\end{aligned}\right\} &\qquad \text{for } p \in S\,, && (2.5)\\
(\mathbf{e}_{0,i} + \mathbf{e}_{0,j} + \mathbf{e}_{p,k} + \mathbf{e}_{p,\ell}\,;\,0) &\qquad \text{for } p \in S,\ 1 \le i,k \le 3, \text{ and } 4 \le j,\ell \le 6, && (2.6)\\
(\mathbf{e}_{0,i} + \mathbf{e}_{0,j} + \mathbf{e}_{p,k} + \mathbf{e}_{p,\ell}\,;\,1) &\qquad \text{for } \left\{\begin{aligned}&p \in S,\ 1 \le k \le 3,\ 4 \le \ell \le 6, \text{ and}\\ &1 \le i < j \le 3 \text{ or } 4 \le i < j \le 6,\end{aligned}\right. && (2.7)\\
(\mathbf{e}_{p,i} + \mathbf{e}_{q,j} + \mathbf{e}_{r,k}\,;\,0) &\qquad \text{for } \{p,q,r\} \in C \text{ and } 1 \le i,j,k \le 3. && (2.8)
\end{aligned}
$$

The number of training examples is $|T| = 35 + (2 + 3^4 + 2 \cdot 3^3)s + 3^3|C| = O(|S| + |C|)$, and T can be constructed in polynomial time from S and C.

Now we verify that the 3SSP instance has a solution if and only if the corresponding CPSN instance is solvable. First assume that there exists a solution (S_1, S_2) of the 3SSP instance (S, C). Let $\theta = 3$, $w_{p,i} = 1$ for all p and i, and

$$
\begin{aligned}
&d_{0,1} = d_{0,2} = d_{0,3} = 0.5, \quad d_{0,4} = d_{0,5} = d_{0,6} = 1 && (2.9)\\
&d_{p,1} = d_{p,2} = d_{p,3} = 0, \quad d_{p,4} = d_{p,5} = d_{p,6} = 1.5 \qquad \text{for } p \in S_1\,, && (2.10)\\
&d_{p,1} = d_{p,2} = d_{p,3} = 1.5, \quad d_{p,4} = d_{p,5} = d_{p,6} = 0 \qquad \text{for } p \in S_2\,. && (2.11)
\end{aligned}
$$

With these parameters, $\mathcal{N}$ is consistent with training examples 2.3–2.7: Two weights $w_{0,i} = w_{0,j} = 1$ cannot reach threshold $\theta = 3$ for negative example 2.3. For any positive example 2.4, the potential of $\mathcal{N}$ equals the threshold at least for the duration of time interval $[1, 1.5)$ according to 2.9. Examples 2.5–2.7 are verified similarly. For negative example 2.8, recall that any $\{p, q, r\} \in C$ intersects both S_1 and S_2 which implies $D_{p,i} \cap D_{q,j} \cap D_{r,k} = \emptyset$ by 2.10 and 2.11 and hence the potential never reaches the threshold. This completes the argument for the CPSN instance to be solvable.

For the converse, assume that there exist weights $w_{p,i}$, delays $d_{p,i}$, and threshold θ such that $\mathcal{N}$ is consistent with training examples 2.3–2.8. Any consistent negative example ensures $\theta > 0$, since a sufficiently large t is not in any $D_{p,i}$ and then $\xi(t) = 0$ while $\xi(t) < \theta$ is still required to hold. We can multiply all weights and the threshold by $3/\theta > 0$, preserving $y_{\mathcal{N}}$. Thus, without loss of generality, we assume $\theta = 3$ in the rest of the proof.

From the simultaneous consistency of training examples 2.3 and 2.4 it follows by the synchronization technique that

$$w_{0,i} > 0 \qquad \text{for } i = 1, \ldots, 6, \text{ and} \tag{2.12}$$

$$D_{0,i} \cap D_{0,j} \cap D_{0,k} \neq \emptyset \qquad \text{for } 1 \leq i < j < k \leq 6. \tag{2.13}$$

Furthermore, for each $p = 0, \ldots, s$, positive examples $\mathbf{e}_{p,1} + \mathbf{e}_{p,2} + \mathbf{e}_{p,3}$ and $\mathbf{e}_{p,4} + \mathbf{e}_{p,5} + \mathbf{e}_{p,6}$ from 2.4 and 2.5 together with $\theta = 3$ guarantee that there are $i \in \{1, 2, 3\}$ and $j \in \{4, 5, 6\}$ such that $w_{p,i} \geq 1$ and $w_{p,j} \geq 1$. Since training set T is invariant under permutations of $\{\mathbf{e}_{p,1}, \mathbf{e}_{p,2}, \mathbf{e}_{p,3}\}$, resp. of $\{\mathbf{e}_{p,4}, \mathbf{e}_{p,5}, \mathbf{e}_{p,6}\}$, we may assume without loss of generality that

$$w_{p,1} \geq 1 \quad \text{and} \quad w_{p,4} \geq 1 \quad \text{for every } p = 0, \ldots, s. \tag{2.14}$$

For any $p \in S$, consider the positive example $\mathbf{e}_{0,1} + \mathbf{e}_{0,2} + \mathbf{e}_{p,1} + \mathbf{e}_{p,4}$ from 2.7, and the negative examples $\mathbf{e}_{0,1} + \mathbf{e}_{0,2}$ from 2.3 and $\mathbf{e}_{0,2} + \mathbf{e}_{0,4} + \mathbf{e}_{p,1} + \mathbf{e}_{p,4}$ from 2.6. Together with the fact that all the involved weights are positive due to 2.12

and 2.14, these examples imply that $D_{0,1}$ intersects $D_{p,1} \cup D_{p,4}$. A symmetric argument shows that $D_{0,4}$ intersects $D_{p,1} \cup D_{p,4}$ as well.

Since $D_{0,1} \cap D_{0,4} \neq \emptyset$ by 2.13 and $\mathbf{e}_{0,1} + \mathbf{e}_{0,4} + \mathbf{e}_{p,1} + \mathbf{e}_{p,4}$ is a negative example 2.6, the interval $D_{p,1}$ is disjoint with $D_{0,1} \cap D_{0,4}$, as the relevant weights are at least 1 according to 2.14; by convexity this implies that $D_{p,1}$ can intersect at most one of $D_{0,1}$ and $D_{0,4}$. Similarly for $D_{p,4}$. Thus $D_{0,1}$ intersects exactly one of $D_{p,1}$ and $D_{p,4}$ and $D_{0,4}$ intersect the other one.

Finally, define the splitting of $S = S_1 \cup S_2$ so that $p \in S_1$ if $D_{p,1}$ intersects $D_{0,1}$ and $p \in S_2$ if $D_{p,1}$ intersects $D_{0,4}$. It remains to prove that (S_1, S_2) is a solution of the 3SSP. Let $\{p, q, r\} \in C$. Suppose for a contradiction that $\{p, q, r\} \subseteq S_j$ for some $j \in \{1, 2\}$. By the definition of (S_1, S_2) this implies that $D_{p,1}$, $D_{q,1}$, $D_{r,1}$ all intersect $D_{0,i}$ for the same $i \in \{1, 4\}$. Since $D_{p,1}$, $D_{q,1}$, $D_{r,1}$ all are all disjoint with $D_{0,1} \cap D_{0,4} \neq \emptyset$ and all the intervals have the same length, this implies that $D_{p,1} \cap D_{q,1} \cap D_{r,1} \neq \emptyset$. However, this contradicts the consistency of the negative example $\mathbf{e}_{p,1} + \mathbf{e}_{q,1} + \mathbf{e}_{r,1}$ from 2.8. Thus (S_1, S_2) solves the 3SSP instance (S, C).

It remains to verify that the result holds even in the restricted models, as specified in the statement of the theorem. If a training set corresponds to a negative input of 3SSP, then by the previous proof, it cannot be represented by the general spiking neurons. It remains to verify that training sets corresponding to a positive input of 3SSP can be represented by the restricted spiking neurons. Their representation in the proof already uses bounded delays from the set $\{0, 0.5, 1, 1.5\}$ and it is easy to verify that the delays can be replaced by $\{d, d+\delta, d+1, d+1+\delta\}$ for arbitrary $d \geq 0$ and $\delta \in (0, 1)$ and thus in particular any real interval of length larger than 1 is sufficient. The construction already uses unit weights and a fixed threshold $\theta = 3$. For a fixed threshold $\theta > 3$, it is easy to use a padded variant construction: add some number of new inputs and new training examples that force the new weights to be at least 1, and let these new inputs be 1 in all the old examples; details are omitted. $\square$

Corollary 1 *If* RP $\neq$ NP*, then a single spiking neuron* $\mathcal{N}$ *with programmable weights, a threshold, and delays is not properly PAC-learnable. In addition, this holds also for a spiking neuron with bounded delays and/or unit weights and/or a fixed threshold.*

A single spiking neuron $\mathcal{N}$ can compute only very simple Boolean functions (Schmitt, 1998). Therefore the consistency problem frequently has no solution: for no setting of weights, a threshold, and delays, function $y_{\mathcal{N}}$ is consistent with all the training examples. In this case, one would be satisfied with a good approximation in practice, that is with the neuron parameters yielding a small training error. For example, in the incremental learning algorithms (e.g., Fahlman & Lebiere, 1990) that adapt single neurons before these are wired to a neural network, an efficient procedure for minimizing the training error is crucial to keep the network size small for successful generalization. Thus the decision version for the approximation problem is formulated as follows:

Approximation Problem for Spiking Neuron $\mathcal{N}$ (APSN)

Instance: A training set T for spiking neuron $\mathcal{N}$ and a positive integer k.

Question: Are there real-valued weights $w_1, \ldots, w_n \in \mathbf{R}$, threshold $\theta \in \mathbf{R}$, and nonnegative delays $d_1, \ldots, d_n \in \mathbf{R}_0^+$ for $\mathcal{N}$ such that $y_{\mathcal{N}}(\mathbf{x}) \neq b$ for at most k training examples $(\mathbf{x}; b) \in T$?

Obviously, the consistency problem is a special case of the approximation problem for $k = 0$. Thus theorem 1 implies that the approximation problem for spiking neuron (APSN) is *NP*-complete which was previously proved separately (Šíma, 2003). Also for the perceptrons with *zero delays* for which the consistency problem is polynomial-time solvable, several authors proved that the approximation problem is *NP*-complete (Höffgen et al., 1995; Roychowdhury, Siu, & Kailath, 1995) even if the threshold is assumed to be zero (Amaldi, 1991; Johnson & Preparata, 1978). Within the PAC framework, the *NP*-hardness of the approximation problem implies that the neuron does not allow *robust learn-*

ing unless $RP = NP$ (Höffgen, Simon, & Van Horn, 1995). Recall that in robust learning target function f can be arbitrary Boolean function and condition 2.2 is then replaced with

$$\sum_{x \in f \Delta y_{\mathcal{N}}} D(\mathbf{x}) < \inf_{g \in \mathcal{F}_{\mathcal{N}}} \sum_{x \in f \Delta g} D(\mathbf{x}) + \varepsilon \, . \tag{2.15}$$

Thus we have the following corollary:

Corollary 2 *If* RP $\neq$ NP*, then a single spiking neuron $\mathcal{N}$ with programmable weights, a threshold, and delays does not allow robust learning. In addition, this holds also for a spiking neuron with bounded delays and/or unit weights and/or a fixed threshold.*

3 The Representation Problem

In this section we deal with the *representation (membership) problem* for spiking neurons:

Representation Problem for Spiking Neuron $\mathcal{N}$ (RPSN)

Instance: A Boolean function f in DNF (disjunctive normal form).

Question: Is f computable by a single spiking neuron $\mathcal{N}$, i.e., are there real-valued weights $w_1, \ldots, w_n \in \mathbf{R}$, threshold $\theta \in \mathbf{R}$, and nonnegative delays $d_1, \ldots, d_n \in \mathbf{R}_0^+$ for $\mathcal{N}$ such that $y_{\mathcal{N}}(\mathbf{x}) = f(\mathbf{x})$ for every $\mathbf{x} \in \{0,1\}^n$?

The representation problem for perceptrons with zero delays, known as the *linear separability* problem, is known to be *coNP*-complete (Hegedüs & Megiddo, 1996). We generalize the *coNP*-hardness result for spiking neurons with arbitrary delays. On the other hand, it is easily seen that the RPSN is in the complexity class Σ_2^p from the polynomial time hierarchy (for a definition, see Balcázar, Díaz, & Gabarró, 1995). Hardness of RPSN for Σ_2^p (or for *NP*) would imply (Aizenstein, Hegedüs, Hellerstein, & Pitt, 1998) that the spiking neurons with arbitrary delays are not learnable with membership and equivalence queries (unless $NP = coNP$); this remains an open problem.

Theorem 2 *The representation problem for spiking neuron (RPSN) is* coNP*-hard and belongs to* Σ_2^p*.*

Proof: The *tautology* problem that is known to be *coNP*-complete (Cook, 1971) will be reduced to RPSN in polynomial time in a similar way as it was done for the linear separability problem (Hegedüs & Megiddo, 1996):

Tautology Problem (TAUT)
Instance: A Boolean function g in DNF.
Question: Is g a tautology, i.e., $g(\mathbf{x}) = 1$ for every $\mathbf{x} \in \{0,1\}^n$?

Thus given a TAUT instance g over n variables $x_1, \ldots, x_n$, we construct a corresponding RPSN instance f over $n+2$ variables $x_1, \ldots, x_n, y_1, y_2$ in polynomial time as follows:

$$f(x_1, \ldots, x_n, y_1, y_2) = (g(x_1, \ldots, x_n) \wedge y_1) \vee (y_1 \wedge \bar{y}_2) \vee (\bar{y}_1 \wedge y_2) \,. \qquad (3.1)$$

For TAUT instance g in DNF, function f can be turned into DNF by one application of distributivity; this DNF is an RPSN instance corresponding to g.

We now show that this TAUT instance is a tautology if and only if the corresponding RPSN instance is solvable. So first assume that g is a tautology. Hence f given by formula 3.1 can be equivalently rewritten as $y_1 \vee y_2$ which is trivially computable by a spiking neuron. On the other hand, assume that there exists $\mathbf{a} \in \{0,1\}^n$ such that $g(\mathbf{a}) = 0$. In this case, $f(\mathbf{a}, y_1, y_2)$ reduces to $XOR(y_1, y_2)$. Using a similar argument as to show that XOR cannot be implemented by a single spiking neuron (Maass & Schmitt, 1999), we show the same for f. (Note that we cannot simply refer to this result as a black box, since the class of functions computed by spiking neurons is not closed under substitution of constants for variables.) Assume for contradiction that f is represented by $\mathcal{N}$ with weights w_1

and w_2 corresponding to the inputs y_1 and y_2, respectively. Since $f(\mathbf{a}, 0, 0) = 0$, $f(\mathbf{a}, 1, 0) = 1$, and $\mathcal{N}$ represents f, we have $w_1 > 0$. On the other hand, since $f(\mathbf{a}, 0, 1) = 1$, $f(\mathbf{a}, 1, 1) = 0$, and $\mathcal{N}$ represents f, we have $w_1 < 0$, a contradiction.

For proving that $RPSN \in \Sigma_2^p$ consider an alternating algorithm for the RPSN that, given f in DNF, guesses polynomial-size representations (Maass & Schmitt, 1999) of weights, a threshold, and delays for spiking neuron $\mathcal{N}$ first in its existential state, and then verifies $y_{\mathcal{N}}(\mathbf{x}) = f(\mathbf{x})$ for every $\mathbf{x} \in \{0, 1\}^n$ in its universal state; note that $y_{\mathcal{N}}(\mathbf{x})$ can be computed in polynomial time since the number of time intervals to be checked grows only linearly. $\square$

Maass & Schmitt observed (1999) that the class of n-variable Boolean functions computable by spiking neurons is *strictly* contained in the class DLT that consists of functions representable as disjunctions of $O(n)$ threshold gates with n inputs (computing Boolean linear threshold functions LT). Thus, class DLT corresponds to two-layer networks with linear number of hidden perceptrons (with zero delays) and one output OR gate. The smallest number of threshold gates in such a representation is called the *threshold number* (Hammer, Ibaraki, & Peled, 1981). It was shown (Schmitt, 1998) that the threshold number of spiking neurons with n inputs is at most $n - 1$ and can be lower-bounded by $\lfloor n/2 \rfloor$. On the other hand, there exists a Boolean function with threshold number 2 that cannot be computed by a single spiking neuron (Schmitt, 1998).

Note that TAUT is *coNP*-hard even if restricted to formulas with a linear number of monomials: For any g in DNF with m monomials, the formula $g \vee x'_1 \vee \ldots \vee x'_m$ with m new variables x'_i is tautology if and only if g is and in addition it has $\Theta(m)$ of both variables and monomials. Since any monomial is a special case of a threshold gate, this DNF formula with linear number of monomials can be further transformed into a DLT formula. It follows that a modified version of RPSN, whose instances are Boolean functions f from DLT (instead of DNF) is also *coNP*-hard. On the other hand, also this modified version is in Σ_2^p, using the same argument as in theorem 2.

4 Conclusion

We have analyzed the computational complexity of training a single spiking neuron with programmable weights, a threshold, and delays. This model covers certain aspects of biological neurons. We have developed a synchronization technique that generalizes the known non-learnability results for arbitrary real-valued synaptic delays. In particular, we have proven that the spiking neurons with binary coded inputs and outputs are not properly PAC-learnable and do not allow robust learning unless $RP = NP$, which solves a previously open problem. In addition, we have shown that it is *coNP*-hard to decide whether a disjunction of $O(n)$ threshold gates, which is known to implement any spiking neuron, can reversely be computed by a single spiking neuron. An open problem remains for further research whether the spiking neurons are learnable with membership and equivalence queries.

Acknowledgments

The presentation of this paper benefited from valuable comments of anonymous reviewers. J.Š.'s research was partially supported by the grant GA CR No. 201/02/1456, the "Information Society" project 1ET100300517, and the Institutional Research Plan AV0Z10300504. J.S.'s research was partially supported by Institutional Research Plan AV0Z10190503 and by Inst. for Theor. Comp. Sci., Prague (projects LN00A056 and 1M0021620808 of MŠMT ČR).

References

Aizenstein, H., Hegedüs, T., Hellerstein, L., & Pitt, L. (1998). Theoretic Hardness Results for Query Learning. *Computational Complexity, 7*(1), 19–53.

Amaldi, E. (1991). On the complexity of training perceptrons. In T. Kohonen, K. Mäkisara, O. Simula, & J. Kangas (Eds.), *Proceedings of the 1st International Conference on Artificial Neural Networks (ICANN'91)* (pp. 55–60). North-Holland, Amsterdam: Elsevier Science Publisher.

Anthony, M., & Bartlett, P. L. (1999). *Neural Network Learning: Theoretical Foundations.* Cambridge, UK: Cambridge University Press.

Balcázar, J. L., Díaz, J., & Gabarró, J. (1995). *Structural Complexity I* (2nd ed.). Berlin: Springer-Verlag.

Blumer, A., Ehrenfeucht, A., Haussler, D., & Warmuth, M. K. (1989). Learnability and the Vapnik-Chervonenkis dimension. *Journal of the ACM, 36*(4), 929–965.

Bohte, M., Kok, J. N., & La Poutré, H. (2000). Spike-prop: error-backpropagation in multi-layer networks of spiking neurons. *Proceedings of the European Symposium on Artificial Neural Networks (ESANN'2000)* (pp. 419–425). Brussels: D-Facto Publications.

Cook, S. A. (1971). The complexity of theorem-proving procedures. *Proceedings of the 3rd Annual ACM Symposium on Theory of Computing (STOC'71)* (pp. 151–158). New York: ACM Press.

Fahlman, S. E., & Lebiere, C. (1990). The cascade-correlation learning architecture. In D. S. Touretzky (Ed.), *Advances in Neural Information Processing Systems (NIPS'89), 2* (pp. 524–532). San Mateo: Morgan Kaufmann.

Garey, M. R., & Johnson, D. S. (1979). *Computers and Intractability: A Guide to the Theory of* NP-*Completeness.* San Francisco: Freeman.

Gerstner, W., & Kistler, W. M. (2002). *Spiking Neuron Models: Single Neurons, Populations, Plasticity.* Cambridge, UK: Cambridge University Press.

Hammer, P.L., Ibaraki, T., & Peled, U. N. (1981). Threshold numbers and threshold completions. In: P. Hansen (Ed.), *Studies on Graphs and Discrete Programming, Annals of Discrete Mathematics, 11 Mathematics Studies, 59* (pp. 125–145). Amsterdam: North-Holland.

Haykin, S. (1999). *Neural Networks: A Comprehensive Foundation* (2nd ed.). Upper Saddle River, NJ: Prentice-Hall.

Hegedüs, T., & Megiddo, N. (1996). On the geometric separability of Boolean functions. *Discrete Applied Mathematics, 66*(3), 205–218.

Höffgen, K.-U., Simon, H.-U., & Van Horn, K. S. (1995). Robust trainability of single neurons. *Journal of Computer and System Sciences, 50*(1), 114–125.

Johnson, D. S., & Preparata, F. P. (1978). The densest hemisphere problem. *Theoretical Computer Science, 6*(1), 93–107.

Judd, J. S. (1990). *Neural Network Design and the Complexity of Learning.* Cambridge, MA: MIT Press.

Karp, R. M. (1972). Reducibility among combinatorial problems. In R. E. Miller, & J. W. Thatcher (Eds.), *Complexity of Computer Computations* (pp. 85–103). New York: Plenum Press.

Maass, W. (1997). Networks of spiking neurons: the third generation of neural network models. *Neural Networks, 10*(9), 1659–1671.

Maass, W., & Bishop, C. M. (Eds.). (1999). *Pulsed Neural Networks.* Cambridge, MA: MIT Press.

Maass, W., & Schmitt, M. (1999). On the complexity of learning for spiking neurons with temporal coding. *Information and Computation, 153*(1), 26–46.

Pitt, L., & Valiant, L. G. (1988). Computational limitations on learning from examples. *Journal of the ACM, 35*(4), 965–984.

Rosenblatt, F. (1958). The perceptron: A probabilistic model for information storage and organization in the brain. *Psychological Review, 65*(6), 386–408.

Roychowdhury, V. P., Siu, K.-Y., & Kailath, T. (1995). Classification of linearly non-separable patterns by linear threshold elements. *IEEE Transactions on Neural Networks, 6*(2), 318–331.

Roychowdhury, V. P., Siu, K.-Y., & Orlitsky, A. (Eds.). (1994). *Theoretical Advances in Neural Computation and Learning.* Boston: Kluwer.

Schmitt, M. (1998). On computing Boolean functions by a spiking neuron. *Annals of Mathematics and Artificial Intelligence, 24*(1-4), 181–191.

Šíma, J. (2002). Training a single sigmoidal neuron is hard. *Neural Computation, 14*(11), 2709–2728.

Šíma, J. (2003). On the complexity of training a single perceptron with programmable synaptic delays. *Proceedings of the 14th International Conference on Algorithmic Learning Theory (ALT'2003), LNAI 2842* (pp. 221–233). Berlin: Springer-Verlag.

Valiant, L. G. (1984). A theory of the learnable. *Communications of the ACM, 27*(11), 1134–1142.

Vidyasagar, M. (1997). *A Theory of Learning and Generalization.* London: Springer-Verlag.