LETTER **Communicated by Sander M. Bohte**

Spiking Neural Networks for Cortical Neuronal Spike Train Decoding

Huijuan Fang
huijuan.fang@gmail.com
Key Laboratory for Image Processing and Intelligent Control of Education Ministry of China, Department of Control Science and Engineering, Huazhong University of Science and Technology, Wuhan 430074, China, and College of Information Science and Engineering, Huaqiao University, Xiamen 361021, China

Yongji Wang
wangyjch@mail.hust.edu.cn
Key Laboratory for Image Processing and Intelligent Control of Education Ministry of China, Department of Control Science and Engineering, Huazhong University of Science and Technology, Wuhan 430074, China

Jiping He
jiping.he@asu.edu
Department of Control Science and Engineering, Huazhong University of Science and Technology, Wuhan 430074, China, and Harrington Department of Bioengineering and the Center for Neural Interface Design, Arizona State University, Tempe, AZ 85287, U.S.A.

Recent investigation of cortical coding and computation indicates that temporal coding is probably a more biologically plausible scheme used by neurons than the rate coding used commonly in most published work. We propose and demonstrate in this letter that spiking neural networks (SNN), consisting of spiking neurons that propagate information by the timing of spikes, are a better alternative to the coding scheme based on spike frequency (histogram) alone. The SNN model analyzes cortical neural spike trains directly without losing temporal information for generating more reliable motor command for cortically controlled prosthetics. In this letter, we compared the temporal pattern classification result from the SNN approach with results generated from firing-rate-based approaches: conventional artificial neural networks, support vector machines, and linear regression. The results show that the SNN algorithm can achieve higher classification accuracy and identify the spiking activity related to movement control earlier than the other methods. Both are desirable characteristics for fast neural information processing and reliable control command pattern recognition for neuroprosthetic applications.

1 Introduction

To rehabilitate or replace impaired or lost motor functions for people who have suffered neurological diseases and injuries, it is now technically feasible to establish an alternative communication and control channel between motor cortical commands and artificial actuators (Donohue, 2002; Nicolelis, 2003; Taylor, Helms Tillery, & Schwartz, 2002; Velliste, Perel, Spalding, Whitford, & Schwartz, 2008; Wolpaw, Birbaumer, McFarland, Pfurtscheller, & Vaughan, 2002). For this purpose, it is important to find an efficient decoding scheme to process the neural signals obtained from the brain and convert them to control commands to operate motor prosthetic devices.

Biological neurons communicate with each other through action potentials that are normally recorded and represented in the form of a sequence of spikes. The most commonly adopted scheme to represent the information in the spike trains is the perievent histogram (creating a frequency profile, that is, counting the average number of spikes in a sequence of small time windows—around 20–50 ms—on spikes trains of repeated trials). Then different computational algorithms have been used to extract the useful information from the perievent histograms, among which the most popular ones are population vector algorithms (PVA), linear filter models, Bayesian inference techniques, artificial neural networks (ANN), and support vector machines (SVM) (Fang, Wang, Huang, & He, 2006; Georgopoulos, Schwartz, & Kettner, 1986; Lin & Si, 1997; Schwartz, Taylor, & Helms Tillery, 2001; Wessberg et al., 2000; Wu, Gao, Bienenstock, Donoghue, & Black, 2005). These decoding algorithms have been applied to predict hand motion from the firing rates of multiple motor cortical neurons and for control of prosthetic devices with various levels of success (Hochberg et al., 2006; Paninski, Fellows, Hatsopoulos, & Donoghue, 2004; Taylor et al., 2002).

However, the success of these decoding algorithms based on frequency information alone is limited. It has been widely recognized that the results of a decoding algorithm based on the histogram or frequency profile are variable and may depend on the event alignment and window size (bin). It is readily recognizable that the exact timing between spikes in each such window, no matter how small, varies from trial to trial, and the process of creating the histogram is to remove such variability. The timing difference between spikes may contain important information in the neural coding. Increasingly experimental evidence suggests that the rate coding concept is too simplistic to describe complex computation and information representation in brain neuronal networks. For instance, visual pattern analysis and classification in nonhuman primates can be carried out in just 100 ms (Perrett, Rolls, & Caan, 1982; Thorpe & Imbert, 1989), in spite of the fact that the network involves a minimum of 10 synaptic stages from the retina to the temporal lobe. This means that each individual processing stage would need to operate in less than 10 ms, whereas the firing rates of neurons involved in these computations are usually below 100 Hz. Thus, the

average processing time in each stage is so short that few neurons will be able to produce more than one spike during the signal propagation process. Further experimental evidence has accumulated to demonstrate that many biological neural systems use the timing of spikes to encode information (Kempter, Gerstner, van Hemmen, & Wagner, 1996; Lestienne, 1996). The evidence of precise temporal correlations between pulses of different neurons (Lestienne, 1996) and stimulus-dependent synchronization of the activity in populations of neurons (Eckhorn et al., 1988; Singer, 1994) is inconsistent with the rate coding. These results suggest that the exact timing of spikes, or the dynamic process of producing action potentials from individual neurons, may play an important role in information transfer between neurons and a coding scheme, taking into consideration that spike timing might be more biologically plausible than the rate coding. These experimental results from neurobiology have led to the investigation of a new neural network model different from the ANN, spiking neural networks (SNN) (Maass, 1997a; Natschläger & Ruf, 1998; Buonomano & Merzenich, 1999; Joshi & Maass, 2005; Wang, Hou, Zou, Tan, & Cheng, 2008).

Spiking neural networks differ from the traditional ANN model in that spiking neurons in SNN propagate information by the timing of individual spikes rather than by the rate of spikes as in ANN. Incorporating this temporal inherent property closer to biological neurons, the SNN models theoretically have more computational power than ANN models with sigmoidal activation functions (Maass, 1997b). In this study, we examined whether SNNs were more powerful than ANNs in preserving or extracting more information from recorded neuronal activities while monkeys performed reach-to-grasp tasks. We would like to extract arm movement direction and hand orientation intent from the timing of spike trains using the SNN-based algorithm and compare its performance with those from other commonly used algorithms. The SNN model in this study is used to decode information from the spike trains for use in brain-computer interface (BCI) and similar applications, not proposed to be a model for neural processing in biological brains.

The motor cortical neural signals were recorded simultaneously with the kinematics of arm movement while the monkeys performed three-dimensional (3D) reach-to-grasp tasks. The raw spike trains were used as input data directly into an SNN model, eliminating the need to transform spike trains into binned firing rates in commonly used approaches that would exclude temporal information in the neural signals for decoding. An error backpropagation (BP) learning rule is applied to train the SNN algorithm. For comparison, the same neuronal data were also decoded using the ANN trained with the BP rule, the SVM algorithm, and linear regression as in Hung, Kreiman, Poggio, and DiCarlo (2005).

The rest of this letter is organized as follows. The biological experimental design and neural data acquisition methods are described in section 2. Then a feedforward layered spiking neural network model is introduced to learn

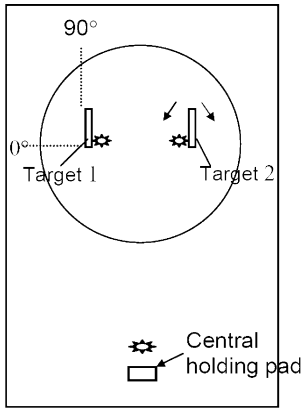

Figure 1: Front view of the experimental apparatus. It shows two targets and the target orientation definition.

the temporal patterns of neural spike trains recorded from the motor cortex. In section 3 we compare the results of analyzing the same spike trains by SNN, ANN, SVM, and linear regression methods. The discussion is in section 4, and it is followed by conclusions in section 5.

2 Methods

2.1 Biological Experimental Design. The more detailed description of the experiment can be found in Fan (2006) and Fan and He (2006). Briefly, rhesus monkeys were trained to perform the visually guided 3D reach-to-grasp task. As shown in Figure 1, the apparatus for grasp consists of a central holding pad and two rectangular bars positioned at approximately shoulder height in the frontal plane. Each of the two bars (left and right) is fitted with touch sensors on both sides and can be rotated to three orientations (45°, 90°, 135°) for grasp. The behavioral events for each movement trial started with monkeys' fingers resting on the central holding pad when the central light was on for a random duration (200–700 ms). Then (1) the central light went off, and one of the two target lights turned on. (2) The monkeys needed to release the central holding pad, reach for the indicated target, and (3) make a whole-hand grasp so that both sides of the bar were contacted to turn off the target light. (4) After the target light went off, the animals would receive a reward. This completed a successful trial.

Target orientation changed randomly after every three successful trials to each target. Eighteen trials for one combination of target location and orientation were obtained for most neurons studied. The average duration of visual cue reaction time (CRT), which is from the illumination of the target light to the central pad release, was 230 $\pm$ 90 (SD) ms. The average

duration of movement time (MT), which is from central pad release to target hit, was 263 ± 62 (SD) ms.

2.2 Neural Data Acquisition and Preprocessing. The activity of single neurons in the hand and arm area of the contralateral motor cortex was recorded extracellularly by multichannel microelectrodes when monkeys performed the 3D reach-to-grasp task. We selected neurons showing high correlation with movement direction or target orientation by a systematic classification.

Among 839 recorded neurons from one monkey, 613 were found to be task related (t-test, $p < 0.05$) and were further analyzed. Of the 613 task-related neurons, 100 (16.3%) were correlated only with movement directions, as they showed statistical significant directional variation (ANOVA, $p < 0.05$) but independent of different target orientations. Thus, the 100 direction-related-only neurons were used for the movement direction prediction described below, and we considered only their data recorded in the trials to the target orientation 45 degrees. We also found a different subpopulation of 97 neurons (15.8% of 613) to be correlated only with target orientations, as they showed statistical significant orientational variation (ANOVA, $p < 0.05$) which was unaffected by movement direction. So we selected the 97 orientation-related-only neurons for the target orientation prediction described below and considered only their data recorded in the trials to the left target.

2.3 Network of Spiking Neurons. The dynamic behavior of an artificial spiking neuron in SNN is closer to the biological neuron than that in ANN. The input and output of a spiking neuron in SNN are described by the chain of precise firing times of the spikes received and emitted, respectively. The behavior of each spiking neuron is modeled according to a simple version of the spike response Model (SRM) (Gerstner & Kistler, 2002), a phenomenological model of neuronal dynamics that formally describes how the incoming spike trains are processed by a neuron to produce a new spike train leaving the neuron. The general idea is depicted in Figure 2.

The state of spiking neuron j in SRM is described by its membrane potential $u_j(t)$. When this potential reaches a threshold θ, the neuron fires a pulse, so-called action potential, or spike that is described by its spike time $t_j^{(f)}$. The output of neuron j is thus fully characterized by the array of spiketimes,

$$\mathcal{F}_j = \left\{ t_j^{(f)}, 1 \le f \le n \right\}, \tag{2.1}$$

where n denotes the number of spikes. The spike train is chronologically ordered, so if $1 \le f < g \le n$, then $t_j^{(f)} < t_j^{(g)}$.

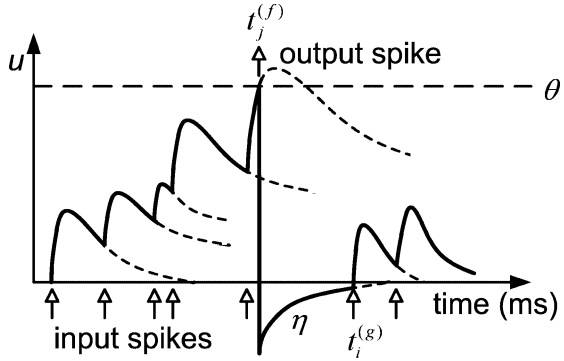

Figure 2: Membrane potential $u(t)$ of a neuron as a sum of weighted PSP due to input spikes $t_i^{(g)}$ (arrow). If the threshold is reached, an output spike $t_j^{(f)}$ is emitted, and a negative kernel η is added so that the potential is reset.

The potential of neuron j is influenced not only by the spikes of its presynaptic neurons Γ_j but also by the spikes produced itself:

$$u_j(t) = \sum_{t_j^{(f)} \in \mathcal{F}_j} \eta\left(t - t_j^{(f)}\right) + \sum_{i \in \Gamma_j} \sum_{t_i^{(g)} \in \mathcal{F}_i} \sum_k w_{ji}^k \varepsilon\left(t - t_i^{(g)} - d_{ji}^k\right), \tag{2.2}$$

where w_{ji}^k is the weight of synapse k from neuron i to neuron j and d_{ji}^k denotes the corresponding delay. The function ε describes a standard postsynaptic potential (PSP) caused by a presynaptic spike and can be described by the following spike response function:

$$\varepsilon(s) = \left[\exp\left(-\frac{s}{\tau_m}\right) - \exp\left(-\frac{s}{\tau_s}\right)\right] \mathcal{H}(s), \tag{2.3}$$

where $\mathcal{H}(s)$ denotes the Heaviside step function: $\mathcal{H}(s) = 0$ for $s \leq 0$ and $\mathcal{H}(s) = 1$ for $s > 0$. The time constants τ_m and τ_s determine the rise and decay of the function.

A neuron's membrane potential will drop instantly to the resting potential after it emits a spike. When the refractory period of a real biological neuron is simulated, a condition is set to make it more difficult for the neuron to generate a second spike shortly after. Since the waveform of a spike is always the same, the exact time course of the action potential carries no information. The event "spike" is fully characterized by the firing time $t_j^{(f)}$. So we neglect the form of the action potential and use a function η to model the negative spike afterpotential,

$$\eta(s) = -\eta_0 \exp\left(-\frac{s}{\tau_r}\right) \mathcal{H}(s), \tag{2.4}$$

where $\eta_0 > 0$ and τ_r is the time constant.

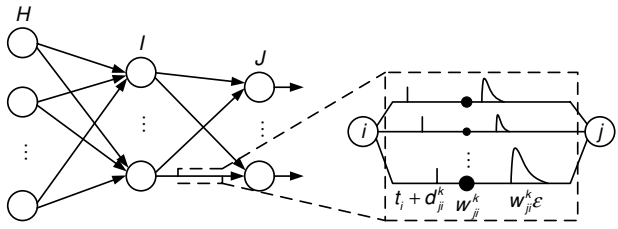

Figure 3: Feedforward spiking neural network. Every connection consists of multiple delayed synapses that all have an adjustable weight, as indicated by the magnification on the right.

The network architecture adopted in this analysis is a fully connected feedforward network with multiple delays per connection, as described in Figure 3. The first layer H, called the input layer, acts as the input of the network. Therefore, the input neurons in H are not involved in real processing; rather, they send experimentally generated spikes into the network. The middle layer I is the hidden layer.

There could be any number of hidden layers. The last layer J is the output layer. The spike trains of these neurons form the output of the network. The set of connections between two layers of neurons can also be seen as a layer. It is the convention to denote a layered network by the number of these connection layers, not by the number of layers of neurons. A two-layer network with one hidden layer and a one-layer network without the hidden layer are used in this study to examine the performance improvement achievable by introducing more complexity (more hidden layers).

There are multiple synapses per connection, similar to biological neurons. Every synapse has an adjustable weight and a different delay, as described in Natschläger and Ruf (1998) and used to test the learning algorithm of SNN in Bohte, Kok, and La Poutré (2002), Schrauwen and Van Campenhout (2004), and Booij and Nguyen (2005). This model of neuronal connection is not supposed to mirror biological reality, merely to provide a substrate for learning different timing relationships. These different delays from different synapses provide a way for the presynaptic neurons to influence the postsynaptic neuron at different times to simulate synaptic actions on a longer timescale than the time interval of the spike response and on a more detailed level. For simplicity we assume, as in Natschläger and Ruf (1998), Bohte et al. (2002), and Booij and Nguyen (2005), that every connection consists of a fixed number of synapses, represented by l delays: $D = \{d^k, 1 \le k \le l\}$.

2.4 Learning Algorithm. The learning algorithm has to change the weights of the synapses in such a manner as to minimize the difference between the actual output and the desired output pattern. The network

could also be trained by changing other parameters, like the synaptic delays (Schrauwen & Van Campenhout, 2004). Here, we choose to keep them fixed. We use the same gradient descent learning rule as in Booij and Nguyen (2005), which is based on the error backpropagation rule for conventional neural networks and can cope with spiking neurons that emit more than one spike. In addition, we use momentum in combination with the adaptive learning rate to update weights Fang, Wang, and Liu (2007).

The network error E is defined to be the sum of the squared error of the first spike of the output neurons J, so later spikes of these neurons are ignored:

$$E = \frac{1}{2}\sum_{j\in J}\left(t_j^{(1)} - t_j^d\right)^2, \tag{2.5}$$

where t_j^d denotes the desired spike time. For example, to perform a movement direction classification for the experiment described in section 2.2, we can define two different desired output spike times, t_j^l and t_j^r to signify the two directions "left" and "right." The weight change for synapse k from neuron h to neuron i in the nth iteration is denoted by

$$\Delta w_{ih}^k(n) = -\kappa_{ih}^k(n+1)\frac{\partial E(n)}{\partial w_{ih}^k(n)} + \alpha\Delta w_{ih}^k(n-1), \tag{2.6}$$

where α is the momentum parameter. The adaptive learning rate κ_{ih}^k is indexed by $n+1$ rather than n to indicate that its update occurs before the w_{ih}^k update.

The delta-bar-delta learning rule is used to adjust the learning rate (Jacobs, 1988):

$$\Delta\kappa_{ih}^k(n) = \begin{cases} a & \text{if } S_{ih}^k(n-1)D_{ih}^k(n) > 0 \\ -b\kappa_{ih}^k(n) & \text{if } S_{ih}^k(n-1)D_{ih}^k(n) < 0 \\ 0 & \text{otherwise} \end{cases} \tag{2.7}$$

$$D_{ih}^k(n) = \frac{\partial E(n)}{\partial w_{ih}^k(n)} \tag{2.8}$$

$$S_{ih}^k(n) = (1-\xi)D_{ih}^k(n) + \xi S_{ih}^k(n-1), \tag{2.9}$$

where a, b, and ξ are all positive parameters. The typical ranges for the parameters are $10^{-4} \le a \le 0.1$, $0.1 \le b \le 0.5$, and $0.1 \le \xi \le 0.7$. Since the error surface for multilayer networks can be complex, choosing a learning rate can be difficult. Delta-bar-delta is a heuristic algorithm for modifying the learning rate as training progresses. For each weight, the gradient at the current time step is compared with the gradient at the previous step. If the

gradient is in the same direction, the current operating point may lie on a relatively flat portion of the error surface. In such a situation, the learning rate should be increased by a in order to reduce the number of iterations to move across the flat portion of the error surface. If the gradient is in the opposite direction, the current operating point may lie on a curved portion of the error surface. In such a situation, the learning rate should be decreased by a factor of b in order to prevent the weight adjustment from oscillating.

Because neuron i can fire multiple times and all of these firing times depend on the weight, the derivative of the network error with respect to the weight in equation 2.6 can be given by

$$\frac{\partial E}{\partial w_{ih}^k} = \sum_{t_i^{(f)} \in \mathcal{F}_i} \frac{\partial E}{\partial t_i^{(f)}} \frac{\partial t_i^{(f)}}{\partial w_{ih}^k}. \tag{2.10}$$

The first factor on the right-hand side of equation 2.10 is calculated depending on where a neuron is located in the network. For output neuron $j \in J$, the derivative of the network error with respect to a spike can be obtained from equation 2.5:

$$\frac{\partial E}{\partial t_j^{(1)}} = t_j^{(1)} - t_j^d. \tag{2.11}$$

For nonoutput neuron $i \notin J$, the derivative of the network error with respect to a spike depends on all spikes of all its immediate neural successors Γ^i:

$$\frac{\partial E}{\partial t_i^{(f)}} = \sum_{j \in \Gamma^i} \sum_{t_j^{(g)} \in \mathcal{F}_j} \frac{\partial E}{\partial t_j^{(g)}} \frac{\partial t_j^{(g)}}{\partial t_i^{(f)}}. \tag{2.12}$$

This means that the error at the last spike of the network has to be computed first, then to the spike preceding the last, and so on, in effect backpropagating the error. The derivative of a postsynaptic spike with respect to a presynaptic spike can be derived using equation 2.2 and the results in Bohte et al. (2002):

$$\begin{aligned} \frac{\partial t_j^{(g)}}{\partial t_i^{(f)}} &= \frac{\partial t_j^{(g)}}{\partial u_j(t_j^{(g)})} \frac{\partial u_j(t_j^{(g)})}{\partial t_i^{(f)}} = \frac{-1}{\partial u_j(t_j^{(g)})/\partial t_j^{(g)}} \\ &\times \left[- \sum_{t_j^{(l)} \in \mathcal{F}_j} \eta'(t_j^{(g)} - t_j^{(l)}) \frac{\partial t_j^{(l)}}{\partial t_i^{(f)}} \quad - \sum_k w_{ji}^k \varepsilon'(t_j^{(g)} - t_i^{(f)} - d_{ji}^k) \right]. \end{aligned} \tag{2.13}$$

This is a recursive function and should be calculated in the time order of spike trains.

The first derivative on the right-hand side of equation 2.13 can also be derived using equation 2.2:

$$\frac{\partial u_j\left(t_j^{(g)}\right)}{\partial t_j^{(g)}} = \sum_{t_j^{(l)} \in \mathcal{F}_j} \eta'\left(t_j^{(g)} - t_j^{(l)}\right) + \sum_{i \in \Gamma_j} \sum_{t_i^{(f)} \in \mathcal{F}_i} \sum_k w_{ji}^k \varepsilon'\left(t_j^{(g)} - t_i^{(f)} - d_{ji}^k\right). \tag{2.14}$$

The second factor on the right-hand side of equation 2.10 can be obtained as in equation 2.13:

$$\frac{\partial t_i^{(f)}}{\partial w_{ih}^k} = \frac{\partial t_i^{(f)}}{\partial u_i(t_i^{(f)})} \frac{\partial u_i(t_i^{(f)})}{\partial w_{ih}^k} = \frac{-1}{\partial u_i(t_i^{(f)})/\partial t_i^{(f)}} \times \left[- \sum_{t_i^{(g)} \in \mathcal{F}_i} \eta'\left(t_i^{(f)} - t_i^{(g)}\right) \frac{\partial t_i^{(g)}}{\partial w_{ih}^k} + \sum_{t_h^{(l)} \in \mathcal{F}_h} \varepsilon\left(t_i^{(f)} - t_h^{(l)} - d_{ih}^k\right) \right]. \tag{2.15}$$

This is again a recursive function, since earlier spikes $t_i^{(g)} < t_i^{(f)}$ also depend on the same weight and influence the potential.

2.5 Enforcing Learning Rule in SNN. If the membrane potential of a neuron does not reach the threshold, the neuron will not generate a spike, just as a biological neuron will not produce an action potential. However, this will result in a problem in SNN: all the learning rules fail once a neuron stops firing. Several heuristic methods have been proposed to handle the problem. One approach is to increase the weight by a small amount or lower the threshold of the postsynaptic neuron (Moore, 2002; Schrauwen & Van Campenhout, 2004). Another approach is to start learning with such high weights that the neurons will initially fire for every input pattern and use a small enough learning rate to prevent output neurons from stopping firing (Booij & Nguyen, 2005). In this study, we used the second approach. The initial range of weights for each layer can be different to handle the weighted sum of different number of presynaptic neurons. Moreover, we used delta-bar-delta learning rules to update learning rates for the weights to deal with the complex error surface.

2.6 Simulation Setup. According to the experiment and data preprocessing, spike times for each trial were referenced to the central pad release time. In our neural networks, we used 50 ms worth of spike trains for two

reasons: (1) the 50 ms bin size is sufficient to analyze the temporal structure in neural spike trains to simulate fast cortical computation, and (2) the interspike intervals of most neurons were less than 50 ms. Thus the bin size is not too short to provide an estimate of the average firing rate (Fan, 2006). We got the spike train data from a 50 ms window at the selected position for each neuron and each trial. Then the time stamps of the extracted spike trains were translated into 0 to 50 ms and input to the spiking neural network. Since some extracted spike trains might not include any spike, we inserted an initiation spike (fired at $t = 0$ ms) in each input spike train to designate the reference start time.

The parameters for the spiking neuron model were the membrane threshold $\theta = 1$, the time constants of the standard PSP $\tau_m = 4$ ms, $\tau_s = 2$ ms, and the time constant of η function $\tau_r = 4$ ms. A one-layer network without the hidden layer and a two-layer network with 12 hidden neurons (including one inhibitory neuron generating only negative sign PSP) were used in this study. Every connection between spiking neurons consisted of 31 synapses with delays $\{1, 3, \ldots, 61\}$ ms. Only positive weights were allowed. The momentum parameter α was set to 0.2. The parameters for the delta-bar-delta learning rule were $a = 10^{-4}$, $b = 0.1$, and $\xi = 0.5$. In the training phase, the early stopping method was used to improve generalization performance, and the maximum number of training epochs was set to 250.

The output layer of the network used to predict the movement direction consisted of only one spiking neuron, which was trained to fire an early spike at $\hat{t} = 51$ ms if the movement direction was "left" and a late spike at $\hat{t} = 61$ ms if the movement direction was "right." In the testing procedure, the output was evaluated as follows. If the output spike was closer to the early spike time ($t < 56$ ms), then the classification of the network was "left," and if the output spike, was closer to the late spike time ($t \geq 56$ ms) or there was no output spike the classification of the network was "right."

The network for target orientation prediction included three output neurons. Output classification was encoded according to a winner-take-all paradigm where the neuron coding for the respective class was designated an early firing time (51 ms) and all others a considerably later one (61 ms). A classification was deemed to be correct if the neuron that fired earliest corresponds to the neuron required to fire first.

To compare with the SNN-based temporal pattern classification, we used the ANN trained with BP rule and the SVM method to analyze the average firing rates of the 50 ms worth of spike trains. The firing rates were computed for each neuron by counting the number of spikes recorded within the 50 ms time window and divided by 50 ms. All neurons in ANN used a tanh-sigmoid transfer function. The firing rates were normalized to have zero mean and unit standard deviation by linear transformation before training. One output neuron was used in the network of movement direction prediction. The "left" class was represented by an output of $+1$ and the "right" class by -1. In the test stage, if the output value was greater than 0,

the classification was "left." And if the output value was less than or equal to 0, the classification was "right." As for target orientation prediction, three output neurons were corresponding to three target orientations. The output of the neuron coding for the respective orientation was designated a value of 1 and all others a value of 0. The orientation to which the corresponding neuron had the maximum output was the predicted target orientation. For the SVM algorithm, we used the radial basis function (RBF) kernel and used the grid search technique to find the best parameters of SVM.

To demonstrate categorization of spike trains that is fed into the SNN, we additionally used the linear regression as in Hung et al. (2005) applied to 50 ms of data chopped up into four 12.5 ms pieces. Each neuron would have four input attributes. Responses from multiple neurons were concatenated as input to the decoding classifier.

2.7 Movement Direction Prediction. First, we used the bin size 50 ms and systematically varied the window position (from 200 ms before movement onset to movement onset) to extract the neuronal spike trains. Here, the 250 ms range during CRT is most predictive for the task of movement direction prediction. Next, a single-input one-layer SNN (or ANN) was performed between the neuronal spike trains (or firing rates) and the movement directions to find the optimal window position for each direction-related-only neuron. The window position that minimized the mean squared error (MSE) was the optimal window position for each neuron. Then, from the results of the optimal window position for each neuron, we chose the optimal window position for a group of neurons to run analyses of neuronal ensembles.

2.8 Target Orientation Prediction. For target orientation prediction, we first used the bin size 50 ms and systematically varied window positions (from 200 ms before movement onset to 200 ms after) that were used to extract the neuronal spike trains. Here, the 450 ms range during CRT and MT is most predictive for the task of target orientation prediction. Next, a single-input one-layer SNN (or ANN) was performed between the neuronal spike trains (or firing rates) and the target orientations to find the optimal window position for each orientation-related-only neuron. Then, from the results of optimal window position for each neuron, we chose the optimal window position for a group of neurons to run analyses of neuronal ensembles.

3 Results

3.1 Prediction of Movement Directions. Figure 4 shows the distribution of the optimal window position for 100 direction-related-only neurons that calculated by SNN and ANN, respectively.

During the reaction time, it takes time for the intention to be filtered down the neural path to activate muscles. Thus, motor cortical neurons

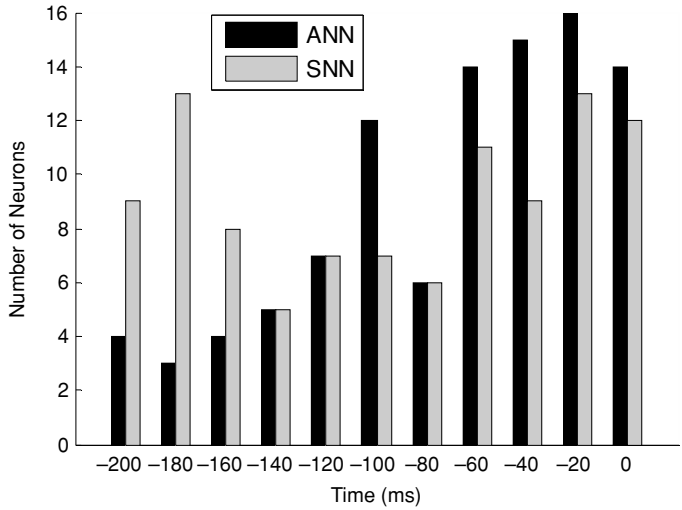

Figure 4: Distribution of the optimal window positions for 100 direction-related-only neurons that were calculated by SNN and ANN respectively. Time zero was aligned at movement onset (i.e., center release).

start to show activities related to movement control over 80 to 150 ms before the actual movement onset. Due to using average firing rates, however, the result of ANN is more closely clustered to the movement onset in Figure 4. Most direction-related-only neurons (77%) show significantly different firing rate patterns related to different movement directions after −100 ms (time zero was aligned at movement onset), especially at the −20 ms window position. This is consistent with what neurophysiological study indicates in Fan (2006). Unlike ANN, SNN analyzes the raw spike trains without losing temporal information and takes into consideration delays and accumulative action of synapses. So the SNN-based method can identify the motor cortical neuron activities related to movement control earlier than ANN. The optimal window positions calculated by SNN are significantly distributed over the early time window before −100 ms (42%), especially at the −180 ms window position. According to the analysis, we used the SNN and the ANN, respectively, to analyze the activities of the direction-related-only neurons at the −20 ms and −180 ms window position to predict the hand movement direction for further study.

The mean classification accuracy of the 100 direction-related-only neurons in each time window is shown in Figure 5. The result of SNN is better than that of ANN in any window. The mean accuracy of SNN in all windows is 62.28%, while that of ANN is 59.25%.

We first analyzed the neural activity within the time window centered at −20 ms, that is, from 45 ms before to 5 ms after center release. Every

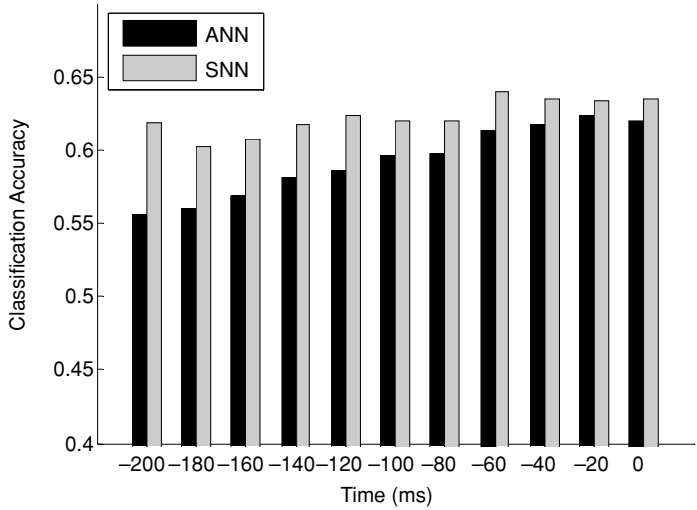

Figure 5: Mean classification accuracy of the 100 direction-related-only neurons in each time window.

neuron has spike trains of 36 trials (18 for "left" and 18 for "right") to the target orientation of 45 degrees. To investigate the case under normal recording conditions, multi-input-neurons were picked up randomly from the 100 direction-related-only neurons. Instead of running the thousands of possible combinations, we randomly picked 20 groups of input neurons and performed one-layer SNN and one-layer ANN analyses for each group. The average classification accuracy was calculated over the 20 groups with SDs. Each data set was randomly partitioned into a training set (50%) and a test set (50%). Such procedures were repeated four times, that is, the accuracy of each group of input neurons was calculated by averaging over the four times. For neuroprosthetic control purposes, it will be feasible to use only about 10 to 20 neurons from all the recorded neurons (Fan, 2006; Wahnoun, He, & Helms Tillery, 2006; Taylor et al., 2002). Furthermore, the results of Fan (2006) suggested that about 10 neurons randomly picked from all movement-related neurons were able to predict hand movement with a very small MSE by the linear regression method. Thus, we used no more than 10 neurons as inputs to the network. Figure 6 shows the classification accuracy averaged over the 20 groups with SDs as a function of the number of neurons. This figure indicates that the SNN, ANN, and SVM methods are all capable of extracting the movement direction from the neural signals in the window located at −20 ms. Moreover, the training and test accuracies of the SNN algorithm are both higher than those of ANN and SVM.

Then, we analyzed the neural activity within the 50 ms window centered at −180 ms—from 205 ms before to 155 ms before center release. According

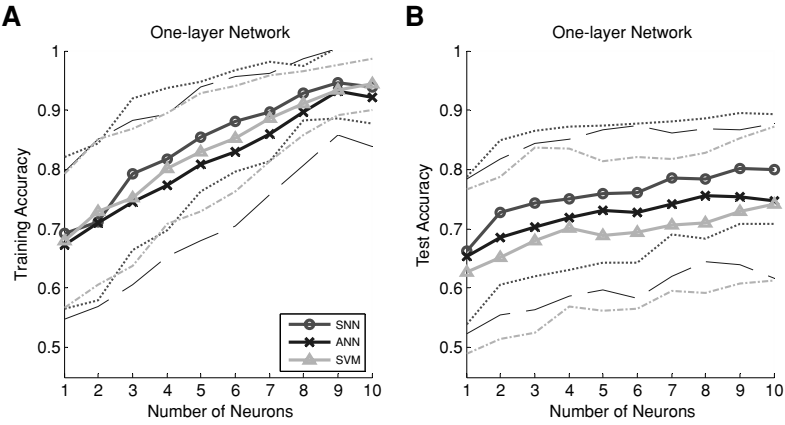

Figure 6: Comparison of classification accuracy of the one-layer SNN, the one-layer ANN, and SVM versus the number of neurons for predicting movement directions in the time window centered at −20 ms (time zero was aligned at movement onset). The accuracy was calculated in the training set (A) and test set (B), respectively. The thick solid curves show the average accuracy of 20 random combination groups. The thin dotted, dashed, and dash-dot curves show the SDs of the SNN, ANN, and SVM methods, respectively.

to the analysis, most neurons' optimal window positions, which were calculated by SNN, located at this time window. Figure 7 shows the rasters of a typical neuron during movements to the left (lower part) and the right (upper part) targets. The spike trains were extracted from a −205 to −155 ms window (zero was aligned at movement onset). In 17 of 36 (47%) trials, the neuron fires just one or no spike. Furthermore, the average number of spikes in this 50 ms time window is 2.22 for the left direction and 2.0 for the right direction. The firing rates are too similar to be distinguished by the ANN. The accuracy of a one-layer ANN analyzing the firing rates is 44.4% even below 50%. However we see from Figure 7 that the temporal patterns of spike trains are significantly different. The distribution of the spike number calculated with a bin of 5 ms averaged from 18 trials for left or right target is shown in Figure 8, which indicates the differences in temporal patterns. A one-layer SNN can recognize the temporal patterns with an accuracy of 80.6%. These results suggest that in the early period of movement preparation, the motor cortical neurons possibly use the timing of single spikes to transfer movement direction information that cannot be decoded by the firing rate's strategy.

From the results of SNN in Figure 4, we selected 10 neurons whose optimal window positions located at −180 ms to analyze further. To predict movement direction using the spike trains of motor cortical neurons, we randomly picked 10 groups of input neurons from the 10 selected neurons

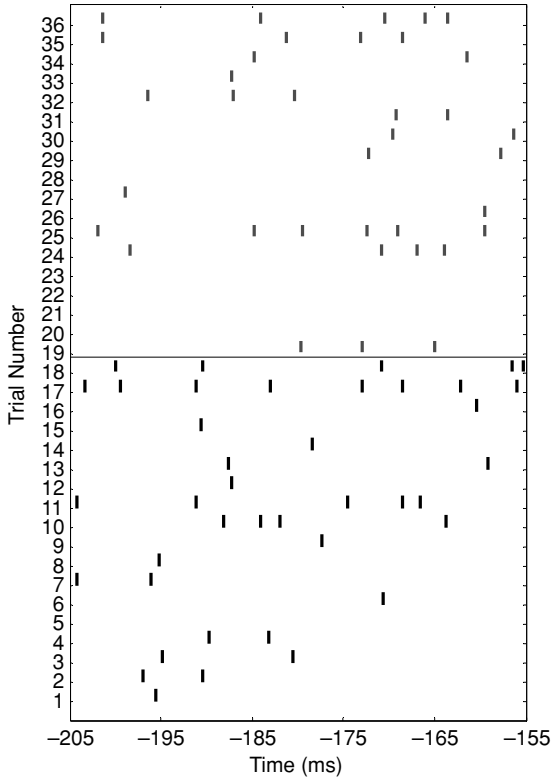

Figure 7: Spike trains of a single neuron during movements to the left (trials 1–18) and the right (trials 19–36) targets. Each spike is denoted by a short vertical bar, with a separate row for each trial. Time zero was aligned at movement onset.

and performed the SNN analysis for each group. The average classification accuracy was calculated over the 10 groups with SDs. The 36 available spike trains were randomly partitioned into a training set (50%) and a test set (50%). Such procedures were repeated four times (i.e., the accuracy of each group of input neurons was calculated by averaging over the four times). Figure 9 shows the classification accuracy averaged over the 10 groups with SDs as a function of the number of neurons. It is obvious that the training and test accuracy using either one-layer or two-layer SNN to analyze the timing of spike trains are all much higher than that using the ANN- and SVM–based approach to analyze the firing rate of spike trains.

For the one-layer network (see Figures 9A and 9B), the training accuracy of the SNN algorithm increases with the number of neurons and can achieve 89.0% by using 10 neurons. However, the change of the training accuracy

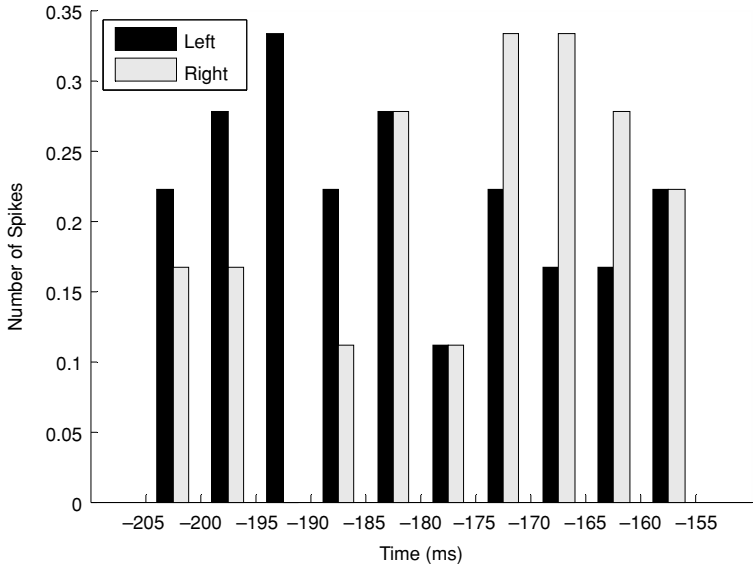

Figure 8: Distribution of the average spike number for the left and the right targets, respectively. The number of spikes was calculated with a bin of 5 ms and averaged from 18 trials. Time zero was aligned at movement onset.

of the ANN algorithm is small, and 10 neurons can classify only 71.0% of the training patterns correctly. The test accuracy of the SNN is also higher than that of ANN. The maximum test accuracy of SNN by using 10 neurons is 82.9%, while that of ANN is 59.9%. For a two-layer network, the SNN's training and test accuracies is higher than that of ANN also. Moreover, the results of the two-layer SNN are better than that of the one-layer SNN. For the two-layer SNN, maximum training and test accuracy can reach 93.5% and 90.4%, respectively, whereas the test accuracy of the two-layer ANN is below even that of the one-layer ANN. The results of the SVM method are close to those of the two-layer ANN. The maximum training and test accuracies of SVM are 80.6% and 59.0%, respectively. The test accuracy of the linear regression method is similar to that of ANN and SVM and is much lower than that of SNN. The maximum test accuracy of the linear regression is 66.7%. However, the linear regression has the highest training accuracy. That is because only 18 samples are used for training the classifier. With five neurons, the linear regression would classify based on the input vector of 20 dimensions, which can completely fit all data in the training set.

3.2 Prediction of Target Orientations. Figure 10 shows the distribution of the optimal window position for 97 orientation-related-only neurons calculated by SNN and ANN, respectively. Similar to the results in Figure 4, the

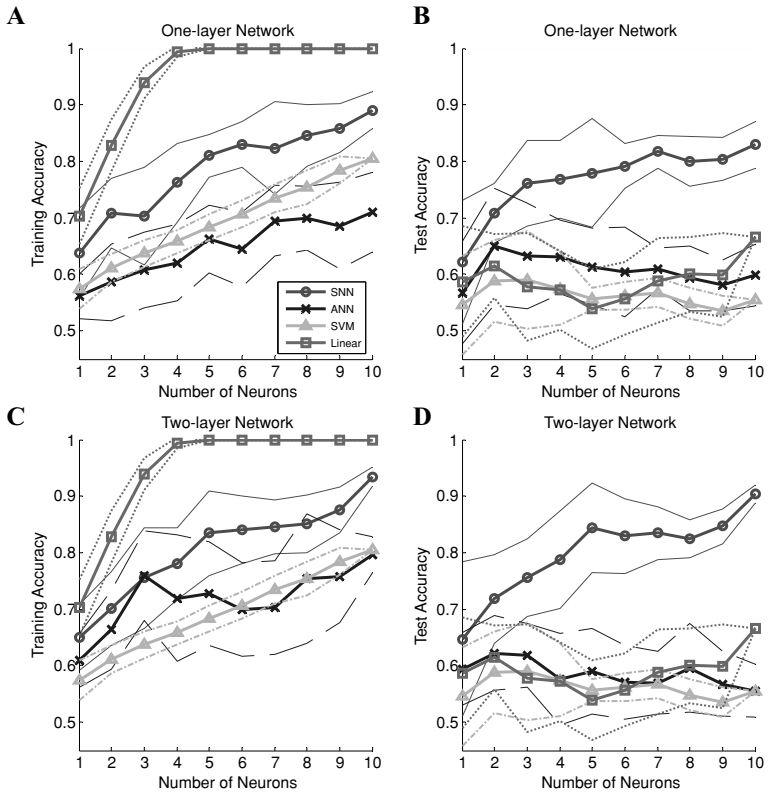

Figure 9: Comparison of classification accuracy of the SNN, ANN, SVM, and linear regression algorithm versus the number of neurons for predicting movement directions in the time window centered at −180 ms (time zero was aligned at movement onset). The accuracy was calculated by a one-layer network (A, B) and two-layer network (C, D), respectively. The thick solid curves show the average accuracy of 10 random combination groups. The thin solid, dashed, dash-dot, and dotted curves show the SDs of the SNN, ANN, SVM, and linear regression methods, respectively.

distribution of optimal windows has a larger early component for SNN than for ANN. Relatively many neurons reached their minimum MSE within a −200 ms window. Thus, we used the SNN, ANN, and SVM approaches, respectively, to analyze the spike train data from 225 ms before center release to 175 ms before center release to predict the target orientation further.

Figure 11 shows the mean classification accuracy of the 97 orientation-related-only neurons in each time window. Like the results of direction-related-only neurons, the accuracy of SNN is higher than that of ANN in any window. Moreover, the advantage of using the SNN-based algorithm

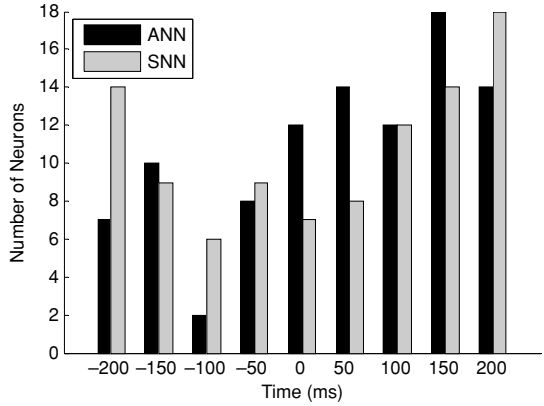

Figure 10: Distribution of the optimal window positions for 97 orientation-related-only neurons calculated by SNN and ANN, respectively. Time zero was aligned at movement onset.

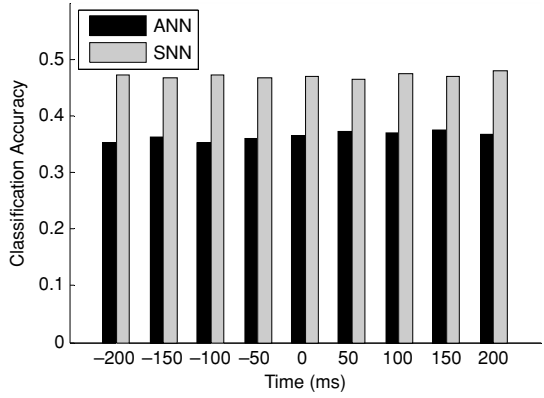

Figure 11: Mean classification accuracy of the 97 orientation-related-only neurons in each time window.

is more obvious than that in the movement direction prediction. The mean accuracy of SNN in all windows is 47.03%, while that of ANN is 36.35%.

From the results of SNN in Figure 10, 10 neurons whose optimal window positions located at −200 ms were selected to predict the target orientation. We randomly picked 10 groups of input neurons from the 10 selected neurons and performed the SNN analysis for each group. The average classification accuracy was calculated over the 10 groups with SDs. The 54 spike trains recorded in the trials to the left target were randomly partitioned into a training set (50%) and a test set (50%). The accuracy of each group of input neurons was calculated by averaging over four times. Figure 12 shows the

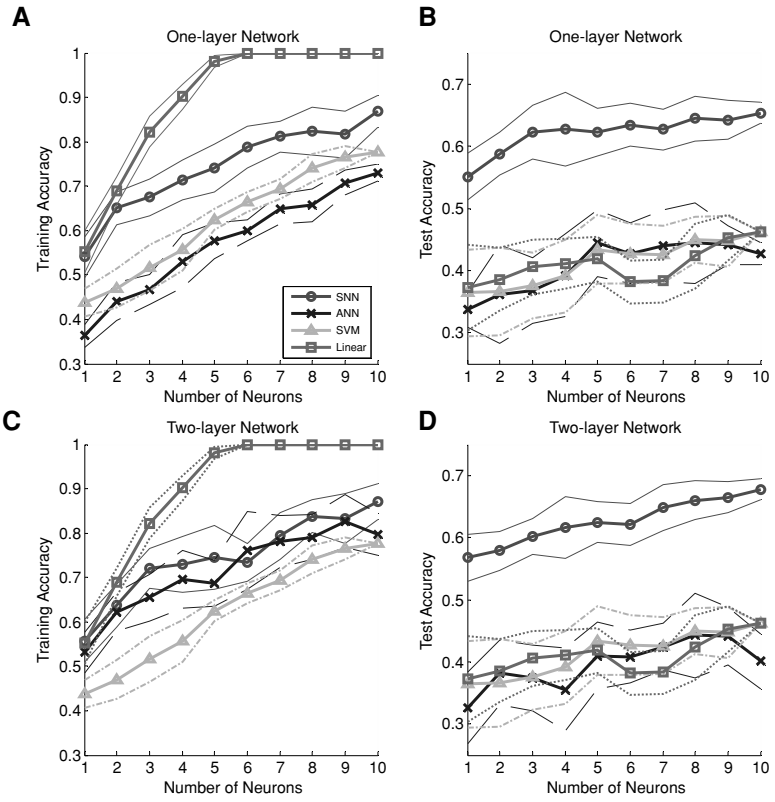

Figure 12: Comparison of classification accuracy of the SNN, ANN, SVM, and linear regression algorithm versus the number of neurons for predicting target orientations in the time window centered at −200 ms (time zero was aligned at movement onset). The accuracy was calculated by one-layer network (A, B) and two-layer network (C, D), respectively. The thick solid curves show the average accuracy of 10 random combination groups. The thin solid, dashed, dash-dot, and dotted curves show the SDs of the SNN, ANN, SVM, and linear regression methods, respectively.

classification accuracy averaged over the 10 groups with SDs as a function of the number of neurons. This figure also indicates that the accuracy of target orientation prediction calculated by the SNN is higher than by the ANN and SVM methods. For the one-layer network (see Figures 12A and 12B), the training and test accuracy of SNN can achieve 86.9% and 65.4%, respectively, while that of ANN can reach but 73.1% and 44.4%, respectively. For the two-layer network (see Figures 12C and 12D), the maximum training and test accuracies of SNN are 87.1% and 67.8%, while those of ANN are 82.7% and 44.3%. The training accuracy of the SVM method is higher than

that of the one-layer ANN but lower than that of the two-layer ANN and the SNN. Additionally, the test accuracies of the SVM is close to that of the ANN method. The maximum training and test accuracies of SVM are 77.8% and 46.3%, respectively. The test accuracy of the linear regression method is similar to that of ANN and SVM and much lower than that of SNN. The maximum test accuracy of the linear regression is 46.3%. The high training accuracy of linear regression is also because of using a small training set.

4 Discussion

From the results, it is clear that spiking neural networks can perform recognition of movement direction and target orientation information from cortical neural spike trains, which are extracted from a short 50 ms bin before movement onset. Thus, the SNN-based algorithm can be used to predict the target location (left or right) and orientation (45°, 90°, or 135°) before the monkey releases the central pad to reach for the indicated target. This result is significant for using cortical neural signals to control motor prosthetic devices to bypass spinal cord injuries.

The comparison between the temporal pattern analysis by SNN and the firing rate analysis by ANN, SVM, and linear regression demonstrates that temporal coding is a viable computational model for fast neural information processing. Nevertheless, the average number of spikes in the 50 ms bin is so few (2.32) that the instantaneous firing rate in the short time window is difficult to estimate correctly on the order of 10 neurons (Gautrais & Thorpe, 1998). These results suggest that the temporal coding is more biologically plausible than the rate coding.

The results of two-layer SNN are better than that of one-layer SNN. Since the one-layer network has only one spiking computational unit (i.e., the only output-neuron), and the multilayer network with more spiking neurons in one or multiple hidden layers has higher computational power (Booij, 2004; Maass, 1997c). In addition, by using the adaptive learning rate method with momentum to speed up training, we obtained an overall better result from SNN than other computational approaches (e.g., ANN, SVM) on a relatively small data set containing only 35 cycles for movement direction prediction and 100 cycles for target orientation prediction. The requirement for data is less when the adaptive learning rate is used in SNN in comparison with studies in other data sets using the SNN method (Bohte et al., 2002).

In the SNN learning algorithm, we considered only the first spike of each output neuron. This is a time-to-first-spike coding strategy for network output based on spike timing. Some researchers have suggested other temporal coding schemes, for example, the latency of the first spike from the stimulus and rank-order code (Thorpe, Delorme, & VanRullen, 2001; Gautrais & Thorpe, 1998), from the time-to-first-spike idea. However, these coding schemes are not feasible for analyzing actual cortical neural signals.

Because of the impact of the neuronal response latency, noise interference, and the error in spike waveforms' sorting, it is difficult to discriminate the first spike that contains all information about movement intention after the onset of stimulus. Nevertheless, the SNN method can extract the spatiotemporal patterns in multiple spikes. It makes the SNN method more robust to analyze actual neuronal spike trains.

The BP learning algorithm was chosen for the SNN in this letter rather than more complicated computing methods, for instance the reservoir method (e.g., liquid state machine, echo state network), because the BP learning algorithm in SNN is similar to BP in ANN, the more popularly used in neural signal analyses, making the comparison straightforward on computational efficiency and power. BP SNN and BP ANN have a similar learning algorithm and network architecture. The most obvious difference between the two methods is the model of artificial neuron in the network. Thus, SNN's better performance is indeed derived from the model of spiking neuron, which is closer to the biological neuron. Moreover, the BP algorithm is our first step in using the SNN method to decode cortical neural spike trains. We will use other computing methods in future research to improve computational efficiency and power.

Another major finding from the results of movement direction and target orientation prediction is that the SNN method based on the precise times of spikes can categorize outcomes earlier than the ANN method based on firing rate. In order to demonstrate this more effectively, we calculated the generalization accuracy in movement direction (see Figures 13A and 13B) and in target orientation prediction (see Figures 13C and 13D) as a function of the window position for the SNN, ANN, and SVM algorithms with 10 neurons. These neurons were randomly picked up from all movement-related neurons or the best combination of neurons distributed at the optimal window positions, which were located at -180 ms for movement direction prediction and -200 ms for target orientation prediction. Figure 13 shows that the earlier the time window location, the better the generalization performance of SNN compared with ANN and SVM. Furthermore, the difference of generalization accuracy is more significant in the best combination. From the results of movement direction prediction in Figures 13A and 13B, we found that after -100 ms, the classification accuracy of ANN and SVM was close to or even exceeded that of SNN. This is because the different firing rate patterns related to different movement directions are more distinct after -100 ms. As for the target orientation prediction, the corresponding significant window position is 50 ms (see Figures 13C and 13D).

5 Conclusion

In this letter, we introduced spiking neural networks to analyze temporal patterns in cortical neural signals. The feedforward layered spiking neural networks trained by backpropagating the temporal error at the output

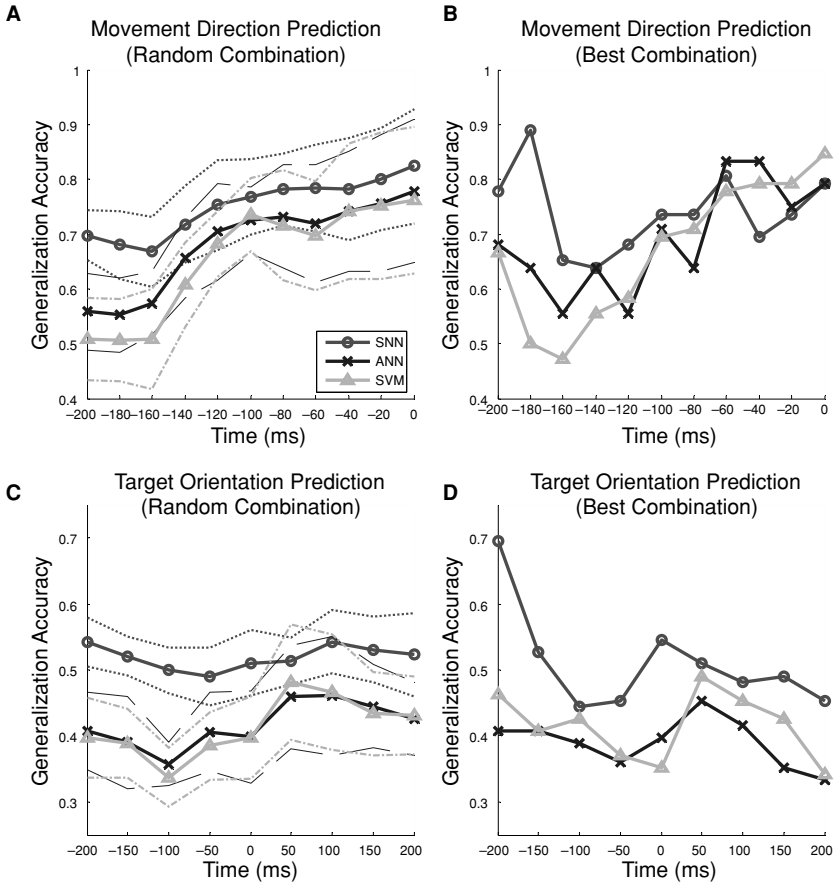

Figure 13: Generalization accuracy of the SNN, ANN, and SVM algorithms as a function of the window position for predicting the movement directions (A, B) and target orientations (C, D) with 10 neurons in the random combination (A, C) of all movement-related neurons or in the best combination (B, D) of neurons distributed at the optimal window positions (−180 ms for the movement direction prediction and −200 ms for the target orientation prediction). In *A* and *C*, the thick solid curves show the average accuracy of 20 random combination groups. The thin dotted, dashed, and dash-dot curves show the SDs of the SNN, ANN, and SVM methods respectively. Time zero was aligned at movement onset.

were used to classify the timing of spike trains, which were recorded from the monkeys' motor cortex when they performed a two-direction three-orientation reach-to-grasp task. The raw spike trains in a short time window (50 ms) were input to the SNN directly to predict the indicated target

location and orientation before movement onset. The results are acceptable. The SNN-based algorithm is indeed capable of learning spike trains and can generalize well, especially by a multilayer network.

The temporal pattern recognition by the SNN algorithm was compared with the firing rate analysis by the ANN, SVM, and the linear regression approach. Our aim is not to compare the computational power between these algorithms but to find a feasible way to decode the information in neural signals. The comparison results are consistent with recent developments in neural coding that temporal coding is a viable code for fast neural information and more biologically plausible than rate coding. The SNN approach could achieve reasonable temporal pattern classification by using a small number of neural signals. On the other hand, the results indicate that the SNN method based on temporal coding can identify the cortical spiking activity related to movement control earlier than the rate-coding-based methods.

The more detailed description of neuronal dynamics by spiking neuron model enables the SNN to decode not only the firing rate pattern of cortical neurons, but also more precise spatiotemporal spike patterns that are not contained in the firing rate of the neurons. The SNN-based method gives us an alternative way to analyze the cortical neural spike trains, directly eliminating the need to transform spike trains into firing rates. It is promising for decoding useful information, which has been neglected by temporal average but related with fast cortical computations.

Acknowledgments

This work is supported in part by National Nature Science Foundation of China under grant 60674105, Doctor Foundation of Ministry of Education of China under grant 20054870013, and Nature Science Foundation of Hubei province of China under grant 2007ABA027. We express our appreciation to the Center for Neural Interface Design of Arizona State University for the data provided from experiments there.

References

Bohte, S. M., Kok, J. N., & La Poutré, H. (2002). Error-backpropagation in temporally encoded networks of spiking neurons. *Neurocomputing, 48*(1–4), 17–37.

Booij, O. (2004). *Temporal pattern classification using spiking neural networks.* Unpublished master's thesis, University of Amsterdam.

Booij, O., & Nguyen, H. T. (2005). A gradient descent rule for spiking neurons emitting multiple spikes. *Information Processing Letters, 95*, 552–558.

Buonomano, D. V., & Merzenich, M. (1999). A neural network model of temporal code generation and position-invariant pattern recognition. *Neural Computation*, *11*, 103–116.

Donohue, J. P. (2002). Connecting cortex to machines: Recent advances in brain interfaces. *Nat. Neurosci., 5* (Suppl.), 1085–1088.

Eckhorn, R., Bauer, R., Jordan, W., Brosch, M., Kruse, W., Munk, M., et al. (1988). Coherent oscillations: A mechanism of feature linking in the visual cortex? *Biol. Cybern., 60*, 121–130.

Fan, J. (2006). *Motor cortical control of hand orientation during 3-D reach-to-grasp.* Unpublished doctoral dissertation, Arizona State University.

Fan, J., & He, J. (2006). Motor cortical encoding of hand orientation in a 3-D reach-to-grasp task. In *Proc. IEEE Engineering in Medicine and Biology Society (EMBS'06)* (pp. 5472–5475). Piscataway, NJ: IEEE.

Fang, H., Wang, Y., Huang, J., & He, J. (2006). Multi-class support vector machines for brain neural signals recognition. In *Proc. of the World Congress on Intelligent Control and Automation (WCICA'06), 2* (pp. 9940–9944). Piscataway, NJ: IEEE.

Fang, H., Wang, Y., & Liu, S. (2007). *Error-backpropagation with adaptive learning rate in spiking neural networks.* Paper presented at the the Second International Conference on Bio-Inspired Computing: Theories and Applications, Zhengzhou, China.

Gautrais, J., & Thorpe, S. (1998). Rate coding versus temporal order coding: A theoretical approach. *BioSystems, 48*, 57–65.

Georgopoulos, A. P., Schwartz, A. B., & Kettner, R. E. (1986). Neuronal population coding of movement direction. *Science, 233*, 1416–1419.

Gerstner, W., & Kistler, W. M. (2002). *Spiking neuron models: Single neurons, populations, plasticity*. Cambridge: Cambridge University Press.

Hochberg, L. R., Serruya, M. D., Friehs, G. M., Mukand, J. A., Saleh, M., Caplan, A. H., et al. (2006). Neuronal ensemble control of prosthetic devices by a human with tetraplegia. *Nature, 442*(7099), 164–171.

Hung, C. P., Kreiman, G., Poggio, T., & DiCarlo, J. J. (2005). Fast readout of object identity from macaque inferior temporal cortex. *Science, 310*, 863–866

Jacobs, R. A. (1988). Increased rates of convergence through learning rate adaptation. *Neural Networks, 1*, 295–307.

Joshi, P., & Maass, W. (2005). Movement generation with circuits of spiking neurons. *Neural Computation*, *17*(8), 1715–1738.

Kempter, R., Gerstner, W., van Hemmen, J. L., & Wagner, H. (1996). Temporal coding in the sub-millisecond range: Model of barn owl auditory pathway. In D. Touretzky, M. Moser, & M. Hasselmo (Eds.), *Advances in Neural Information Processing Systems*, *8* (pp. 124–130). Cambridge, MA: MIT Press.

Lestienne, R. (1996). Determination of the precision of spike timing in the visual cortex of anaesthetised cats. *Biological Cybernetics, 74*, 55–61.

Lin, S., & Si, J. (1997). Self-organization of firing activities in monkey's motor cortex: Trajectory computation from spike signals. *Neural Computation, 9*(3), 607–621.

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

Maass, W. (1997b). Noisy spiking neurons with temporal coding have more computational power than sigmoidal neurons. In M. Mozer, M. I. Jordan, & T. Petsche (Eds.), *Advances in neural information processing systems, 9* (pp. 211–217). Cambridge, MA: MIT Press.

Maass, W. (1997c). Fast sigmoidal networks via spiking neurons. *Neural Computation, 9*(2), 279–304.

Moore, S. C. (2002). *Back-propagation in spiking neural networks.* Unpublished master's thesis, University of Bath.

Natschläger, T., & Ruf, B. (1998). Spatial and temporal pattern analysis via spiking neurons. *Network: Comp. Neural Systems*, *9*(3), 319–332.

Nicolelis, M. A. L. (2003). Brain-machine interfaces to restore motor function and probe neural circuits. *Nat. Rev. Neurosci., 4*, 417–422.

Paninski, L., Fellows, M. R., Hatsopoulos, N. G., & Donoghue, J. P. (2004). Spatiotemporal tuning of motor cortical neurons for hand position and velocity. *J. Neurophysiol., 91*, 515–532.

Perrett, D. I., Rolls, E. T., & Caan, W. C. (1982). Visual neurons responsive to faces in the monkey temporal cortex. *Experimental Brain Research, 47*, 329–342.

Schrauwen, B., & Van Campenhout, J. (2004). Extending spikeprop. In *Proceedings of IEEE International Joint Conference on Neural Networks, 1* (pp. 471–475). Piscataway, NJ: IEEE.

Schwartz, A. B., Taylor, D. M., & Helms Tillery, S. I. (2001). Extraction algorithms for cortical control of arm prosthetics. *Current Opinion in Neurobiology, 11*(6), 701–707.

Singer, W. (1994). The role of synchrony in neocortical processing and synaptic plasticity. In E. Domany, J. L. van Hemmen, & K. Schulten (Eds.), *Models of neural networks II*. Berlin: Springer.

Taylor, D. M., Helms Tillery, S. I., & Schwartz, A. B. (2002). Direct cortical control of 3D neuroprosthetic devices. *Science, 296*(5574), 1829–1832.

Thorpe, S. J., & Imbert, M. (1989). Biological constraints on connectionist modelling. In R. Pfeifer, Z. Schreter, F. Fogelman-Soulie, & L. Steels (Eds.), *Connectionism in Perspective* (pp. 63–92). Amsterdam: Elsevier, North Holland.

Thorpe, S. J., Delorme, A., & VanRullen, R. (2001). Spike-based strategies for rapid processing. *Neural Networks*, *14*, 715–725.

Velliste, M., Perel, S., Spalding, M. C., Whitford, A. S., & Schwartz, A. B. (2008). Cortical control of a prosthetic arm for self-feeding. *Nature, 453*, 1098–1101.

Wahnoun, R., He, J., & Helms Tillery, S. I. (2006). Selection and parameterization of cortical neurons for neuroprosthetic control. *J. Neural Eng., 3*, 162–171.

Wang, X., Hou, Z., Zou, A., Tan, M., & Cheng, L. (2008). A behavior controller based on spiking neural networks for mobile robots. *Neurocomputing*, *71*, 655–666.

Wessberg, J., Stambaugh, C. R., Kralik, J. D., Beck, P. D., Laubach, M., Chapin, J. K., et al. (2000). Real-time prediction of hand trajectory by ensembles of cortical neurons in primates. *Nature, 408*(6810), 361–365.

Wolpaw, J. R., Birbaumer, N., McFarland, D. J., Pfurtscheller, G., & Vaughan, T. M. (2002). Brain-computer interfaces for communication and control. *Clinical Neurophysiology, 113*, 767–791.

Wu, W., Gao, Y., Bienenstock, E., Donoghue, J. P., & Black, M. J. (2005). Bayesian population decoding of motor cortical activity using a Kalman filter. *Neural Computation, 18*(1), 80–118.

Received October 17, 2008; accepted July 13, 2009.