Classification and regression of spatio-temporal signals using NeuCube and its realization on SpiNNaker neuromorphic hardware

DOI:
10.1088/1741-2552/aafabc

Document Version
Accepted author manuscript

Link to publication record in Manchester Research Explorer

Citation for published version (APA):

Published in:
Journal of Neural Engineering

Citing this paper
Please note that where the full-text provided on Manchester Research Explorer is the Author Accepted Manuscript or Proof version this may differ from the final Published version. If citing, it is advised that you check and use the publisher's definitive version.

General rights
Copyright and moral rights for the publications made accessible in the Research Explorer are retained by the authors and/or other copyright owners and it is a condition of accessing publications that users recognise and abide by the legal requirements associated with these rights.

Takedown policy
If you believe that this document breaches copyright please refer to the University of Manchester’s Takedown Procedures [http://man.ac.uk/04Y6Bo] or contact uml.scholarlycommunications@manchester.ac.uk providing relevant details, so we can investigate your claim.
Classification and Regression of Spatio-Temporal Signals using NeuCube and its realization on SpiNNaker Neuromorphic Hardware

Jan Behrenbeck 1, Zied Tayeb 2,3, Cyrine Bhiri 2, Christoph Richter 2, Oliver Rhodes 4, Nikola Kasabov 5, Josafath I. Espinosa-Ramos 5, Steve Furber 4, Gordon Cheng 3 and Jörg Conradt 2

1 Department of Mechanical Engineering, Technical University of Munich, Germany
2 Neuroscientific System Theory, Department of Electrical and Computer Engineering, Technical University of Munich, Germany
3 Institute for Cognitive Systems, Technical University of Munich, Germany
4 School of Computer Science, The University of Manchester, UK
5 Knowledge Engineering and Discovery Research Institute, Auckland University of Technology, AUT Tower, Level 7, cnr Rutland and Wakefield Street, Auckland 1010, New Zealand

E-mail: zied.tayeb@tum.de

Abstract. Objective. The objective of this work is to use the capability of spiking neural networks to capture the spatio-temporal information encoded in time-series signals and decode them without the use of hand-crafted features and vector-based learning and the realization of the spiking model on low-power neuromorphic hardware. Approach. The NeuCube spiking model was used to classify different grasp movements directly from raw surface electromyography signals (sEMG), the estimations of the applied finger forces as well as the classification of two motor imagery movements from raw electroencephalography (EEG). In a parallel investigation, the designed spiking decoder was implemented on SpiNNaker neuromorphic hardware, which allows low-energy real-time processing. Results. Experimental results reveal a better classification accuracy using the NeuCube model compared to traditional machine learning methods. For sEMG classification, we reached a training accuracy of 85% and a test accuracy of 84.8%, as well as less than 19% of relative root mean square error (rRMSE) when estimating finger forces from six subjects. For the EEG classification, a mean accuracy of 75% was obtained when tested on raw EEG data from nine subjects from the existing 2b dataset from “BCI Competition IV”. Significance. This work provides a proof of concept for a successful implementation of the NeuCube spiking model on the SpiNNaker neuromorphic platform for raw sEMG and EEG decoding, which could chart a route ahead for a new generation of portable closed-loop and low-power neuroprostheses.

Keywords: Surface EMG (sEMG), EEG, NeuCube, Spiking neural networks, SpiNNaker neuromorphic platform, Prosthetic hands. Submitted to: J. Neural Eng.
**Featured Application:** A neuroprosthesis is a device that has a direct interface with the nervous system and supplements or substitutes functionality in the patient’s body. Regarding the increasing consumer base of amputees, neuroprosthetic research has gained momentum over the last decades. However, current neuroprostheses still exhibit various drawbacks, such as high power consumption, low computational speed and consequently limited control capabilities. This work investigates the realization of the NeuCube spiking algorithm to efficiently decode different hand poses and associated forces from sEMG signals as well as the classification of motor imagery movements from EEG without hand-crafted features. This spike-based neuronal processing algorithm yields significantly higher flexibility and inherent run-time adaptation, well beyond traditional hard-coded signal processing methods. In a parallel investigation, this work explores also the implementation of this decoding algorithm on dedicated neuromorphic hardware, which allows low-energy real-time processing to create portable closed-loop neuroprosthetic devices.

![Figure 1: An overview of the proposed system, including sMG acquisition and processing, spiking decoding algorithms, spiNNaker neuromorphic hardware and a prosthetic hand.](image)

1 **Introduction**

There has been a rapid rise in the use of sEMG analysis for surface myoelectric signal pattern recognition, which is commonly investigated for the control of upper-limb prostheses, human-assisting robots, as well as rehabilitation devices [1, 2, 3, 4, 5, 6]. Recent studies have focused on how to increase the level of comfort and effectiveness of these devices. Particularly, these studies have tended to focus on reducing the latency period between intention and the subsequent behavior. With regard to traditional
approaches towards robust pattern recognition, the focus has always been on extracting hand-crafted features, which involves designing engineering features and feeding them into a classifier or a regressor [7, 8, 9]. However, this technique becomes computationally demanding when classifying multiple hand gestures [10, 11] or decoding complex reach-to-grasp movements [12]. Hence, it is not suitable for real-time prostheses control, where a very low latency is always required [13]. This work primarily focuses on the decoding of four different hand poses, namely two-digit pinch, three-digit pinch, a fist and rest from raw sEMG signals using the NeuCube spiking neural network [14]. To further validate the usefulness of the proposed method, the spiking model was also used to decode motor imagery movements using an existing dataset known as 2b EEG dataset from BCI Competition IV [15]. In both case studies, the spiking model was tested on a standard computer and was thereafter implemented on the SpiNNaker neuromorphic hardware [16]. Such a hardware system offers a low power consumption, scalability and a competitive real-time capability, which could enhance real-time prostheses control. The implemented brain-inspired model comprises three stages: An encoding stage, which converts time-series signals into spike trains; a three-dimensional brain-shaped recurrent SNN reservoir, which learns spatio-temporal patterns; and an output classifier, which is trained on class-specific reservoir dynamics and predicts class labels for unknown samples. Overall, this work presents a first attempt to use the advantages of SNN models on a dedicated hardware for sEMG and EEG decoding. In particular, the NeuCube was used to decode different hand gestures and associated forces from sEMG signals without hand-crafted features, and to the best of our knowledge, no similar work has been carried out before. Thus, this work argues that using deep learning of spatio-temporal sEMG/EEG data with SNN can be further investigated to achieve a smooth, accurate and adaptive control of hand prosthetics for complex movements within a larger time window. The remainder of this paper is structured as follows: The first section provides an overview of previous work on sEMG signal decoding using traditional machine learning methods as well as SNN approaches. In addition, previous applications of SpiNNaker and NeuCube are reviewed in this first section. The second section describes the sEMG signal processing as well as the implementation details of the NeuCube model on a standard computer. The neuromorphic implementation of the NeuCube on SpiNNaker is described in Section 3. Obtained results for hand poses, force estimation from sEMG signals and EEG-based motor imagery movements using the NeuCube model are shown in Section 4. Finally, Section 5 enumerates the strengths and weaknesses of the used method and proposes possible future improvements.

1.1 Related work

1.1.1 sEMG signal decoding algorithms
This section provides a review of the different algorithms used to classify sEMG signals spanning from classic machine learning techniques to SNN approaches. In [17], the authors investigated the decoding of sEMG in the early stages of reach-to-grasp
movements (pre-reach-to-grasp) for prosthetic hand control. Five different grasp types were classified, namely precision disk, tripod, thumb-2 fingers, thumb-4 fingers, and ulnar pinch. In the paper, the authors showed that an accuracy of 90% of the final grasp gesture was obtained 0.5 s after the movement onset. In [18], the authors used the NeuCube spiking model to classify six different hand gestures, namely hand closed, hand open, wrist flexion, wrist extension, ulnar and radial deviation. Various feature-extraction techniques were investigated (Mean-absolute value (MAV), Wavelength (WL) and combined features). Overall, a classification accuracy of 95.3% was reached and the authors showed that the spiking approach outperformed classic machine learning methods when using MAV features. It should be noted, however, that an accuracy of only 68.7% was attained when classifying raw sEMG data without hand-crafted features. Table 1 compares different studies for sEMG classification and summarizes the obtained results. For a more comprehensive survey of sEMG classification, readers are referred to [19].

Table 1: Comparison of different sEMG classification studies and their achieved accuracy.

<table>
<thead>
<tr>
<th>Reference</th>
<th>No of classes</th>
<th>Extracted features</th>
<th>Classifier</th>
<th>Accuracy</th>
</tr>
</thead>
<tbody>
<tr>
<td>[20]</td>
<td>7</td>
<td>RMS</td>
<td>Adaptive Fuzzy classifier (ANFIS)</td>
<td>86%</td>
</tr>
<tr>
<td>[21]</td>
<td>10</td>
<td>RMS and WL</td>
<td>Linear discriminant analysis (LDA)</td>
<td>98.87%</td>
</tr>
<tr>
<td>[22]</td>
<td>5</td>
<td>RMS</td>
<td>Support vector machine (SVM)</td>
<td>73%</td>
</tr>
<tr>
<td>[23]</td>
<td>6</td>
<td>MAV and WL</td>
<td>LDA</td>
<td>91.64%</td>
</tr>
<tr>
<td>[24]</td>
<td>3</td>
<td>TD</td>
<td>Fuzzy logic (FL)</td>
<td>97%</td>
</tr>
<tr>
<td>[25]</td>
<td>4</td>
<td>MAV, WL and RMS</td>
<td>Artificial neural networks (ANN)</td>
<td>89.2%</td>
</tr>
<tr>
<td>[26]</td>
<td></td>
<td>Time-Frequency (TF)</td>
<td>SVM</td>
<td>90%</td>
</tr>
<tr>
<td>[27]</td>
<td>4</td>
<td>TF</td>
<td>ANFIS</td>
<td>92%</td>
</tr>
<tr>
<td>[18]</td>
<td>6</td>
<td>MAV, WL</td>
<td>NeuCube</td>
<td>95.33%</td>
</tr>
<tr>
<td>[18]</td>
<td>6</td>
<td>Raw data</td>
<td>NeuCube</td>
<td>68.7%</td>
</tr>
<tr>
<td>[28]</td>
<td>9</td>
<td>spiking neurons</td>
<td>Neural networks</td>
<td>92.3%</td>
</tr>
<tr>
<td>Our work</td>
<td>4</td>
<td>Raw data</td>
<td>NeuCube</td>
<td>84.8%</td>
</tr>
</tbody>
</table>

1.1.2 NeuCube applications

The NeuCube model has been successfully applied and evaluated in several studies. In the context of EEG signal classification, the system was tested to model peri-perceptual brain processes from EEG signal [29]. Doborjeh et al. tested the model in a similar study on fMRI data [30]. Chen et al. and Taylor et al. performed two feasibility studies for the use of NeuCube for motor-imagery based brain-computer interfaces (BCI) resulting in accuracy values of up to 81% for recognition of wrist flexion, extension, and rest [31, 32]. NeuCube provides generally higher performance than traditional classifiers, which shows the potential of the model for sEMG pattern classification [18]. Due to the good results in EEG-BCI studies, the NeuCube algorithm has been also adapted and has been successfully applied to other applications, such as traffic and sleep data.
Classification and Regression of Spatio-Temporal Signals using NeuCube [33], earthquake data [34] as well as to perform ecological forecasts [35]. These positive results motivated our choice of using the NeuCube spiking algorithm for this study.

1.1.3 SpiNNaker neuromorphic hardware applications
SpiNNaker is a massively parallel hardware system, inspired by the human brain, which allows the simulation of large numbers of neurons in a biologically plausible fashion [16]. The SpiNNaker board has already been applied in robotic contexts and connected to other neuromorphic hardware such as the eDVS silicon retina sensor to control an autonomous mobile robot, which can identify and approach a signal in real-time [36]. In other works, SpiNNaker was used to control a musculoskeletal robot in real-time [37] as well as decoding motor imagery movements from EEG signal [38]. Furthermore, SpiNNaker was used for audio samples [39] and handwritten digits classification [40].

2 Materials and Methods

2.1 sEMG signals recording
sEMG data were recorded from four healthy subjects: three right-handed males and one left-handed female. Subjects were between 22 and 24 years old and claimed to have no record of neurological disorders. Before this study, they had never performed tasks involving myoelectric control of any external devices. Subjects were recruited to perform a series of hand movements, namely 2-digit grasp, 3-digit grasp, fist and rest (hand open). The grasping tasks were performed on an easy to grasp cylindrical object as shown in Figure 2. Each trial consisted of five seconds of movement execution followed by a five-second break. During each trial, the subject was asked to perform one of the four postures, which were selected from a discrete uniform distribution, with two levels of force (high, low). Between two sessions, five minutes were given to the subject in order to avoid muscle fatigue. All recording sessions took place in our lab and were set-up as shown in Figure 2b. Each of the three recording sessions consisted of 12 trials separated by short breaks to avoid muscle fatigue, yielding a total number of 36 trials for each subject. A total number of 108 trials for each movement was acquired from all the three sessions for each subject. Data used in this example study as well as the experimental paradigm are made publicly available with the gumpy toolbox [41]. To measure the applied force, force sensing resistors (FS402) were placed on all the fingertips, and a voltage divider and an Arduino Uno were used to acquire the global force value, which was set as the sum of each individual measured finger force. As we were mostly interested in the global force applied on the object, we chose to send the sum of the forces applied by each finger to get reliable force information. Furthermore, our Arduino board was only able to send 2000000 bits/second. However, if we wanted to send the voltage information of each finger to the computer with a frequency of 512 Hz, the number of bits we had to send per second was bigger than 2000000. sEMG data were recorded and sampled at 512 Hz using a g.USBamp system. Four pairs of
electrodes, which cover the muscles of the forearm in a ring-like fashion were used, and the reference electrode was placed on the elbow bone. It should be noted that liquid-gel ECG electrode (model: 50 x 48 mm) were used for sEMG recording.

![Image of sEMG recording paradigm](image.png)

Figure 2: sEMG recording paradigm. (a) Different hand gesture renderings prompting subjects. (b) Recording setup of sEMG signals during grasp movements.

### 2.2 Data preprocessing

sEMG signals were processed using the gumpy.signal module in the gumpy BCI toolbox [41]. sEMG signals were band-pass filtered between 20 and 450 Hz using a 4th order zero-phase Butterworth filter and notch filtered at 50 Hz to remove the power line interference. Afterwards, two normalization steps were performed. First, recorded signals were normalized between 0 and 1. Second, the computed mean of the signal was subtracted from each channel and then divided by the standard deviation. Epochs of 200 time points were extracted, yielding a total of 144 epochs for each session, which were thereafter divided into training and validation samples. Due to multiple zeros at the beginning of each trial, the first 500 recorded time points were discarded for the classification task. This delay can be explained as a result of the average time taken by the subject to correctly perform the task and the comfort of a recording session.

### 2.3 NeuCube

The mechanism of neuronal communication can be rebuilt through mathematical models to help create more efficient solutions to pattern recognition when it comes to complex and large signals such as spatial-/spectral-temporal data (SSTD). NeuCube is a 3-D SNN reservoir biologically inspired by the human brain that can perform both classification and regression tasks on complex signals, which was proposed by N. Kasabov [14]. At its core, NeuCube has some similarities with a liquid state machine [42], but it differs from it in many aspects: the structure is brain-inspired; mapping of input variables typically
preserves their spatial location; and both unsupervised and supervised learning are applied. The NeuCube architecture comprises two basic parts, a recurrent 3D SNN reservoir, also called SNNcube, and a readout function which is also realised as an SNN. It performs pattern recognition as follows: First, a spatio-temporal (or spectral) input signal \( x(t) \) within a time window is acquired and encoded with some encoding function \( e : N \times R^n \rightarrow N \times B^n \) into a discrete time series of binary spikes, known as spike trains. Afterwards, the SNNcube is stimulated with the input spike train and learns the interaction between the input signals in an unsupervised mode. The SNNcube creates spatio-temporal patterns of connections as a result of this learning. The learned spatio-temporal patterns are transferred into another signal using a function \( l : N \times B^n \rightarrow N \times B^p \), which integrates the spiking dynamics of the SNNcube. Due to the internal structure and properties of the SNNcube, different input samples induce different dynamic reactions in the SNNcube that are captured by a function \( s : N \times B^p \rightarrow R^p \) creating a spatio-temporal pattern of connectivity for each input sample. Finally, a readout function \( r : R^p \rightarrow \{0,1,...,N\} \) is trained on the spatio-temporal patterns activated in the SNNcube by the input data samples to represent N input classes. In the end, NeuCube can predict the class \( c \) of a new input signal \( x \) as \( c(x) = r(s(l(e(x)))) \). Figure 3 provides an overview of NeuCube’s modules and how they were used for sEMG hand pose classification. More information about the NeuCube architecture can be obtained from [14, 43, 44].

Figure 3: NeuCube’s structure and its different modules used for sEMG signal classification.

2.3.1 Spatio-temporal signal encoding
To encode sEMG and EEG time-continuous values into spike trains, a temporal
difference (TD) algorithm was used. The algorithm works as follows: Every time step $\tau$, the difference $\Delta v = v(\tau) - v(\tau - 1)$ between the current and the previous value in the sEMG signal is calculated and compared to a threshold. If the absolute difference exceeds the threshold, a spike is generated. The sign of the difference $\Delta v$ defines the sign of the spike and thereby its mode: If the spike is excitatory, the signal rises (+1), if the spike is inhibitory, the signal falls (-1) and if the absolute difference is smaller than the threshold, there is no spike (0).

Other algorithms for data encoding into spikes and decoding can be found in [43]. A widely used algorithm is the Bens Spiker Algorithm (BSA) developed by Schrauwen et al. [45]. The BSA uses solely excitatory spikes and makes use of a finite impulse response (FIR) reconstruction filter. It works as follows: Every instant in time $\tau$ two error metrics are calculated:

\begin{align*}
\text{error}_1 &= \sum_{i=0}^{M} \text{abs} (\text{Signal}_{\text{ori}}(i + \tau) - h(i)) \\
\text{error}_2 &= \sum_{i=0}^{M} \text{abs} (\text{Signal}_{\text{ori}}(i + \tau))
\end{align*}

with $\text{Signal}_{\text{ori}}$ being the original signal, $h$ is the FIR filter window function, and $M$ is the filter size. If the first error metric is smaller than the second minus a threshold, a spike is generated and the filter is subtracted from the input signal; otherwise no spike is emitted.

\subsection{NeuCube initialization and training}

Some Atlas-models have been used to define the spatial characteristics of the brain by describing the structure or a function of the whole brain, mapping of groups of neurons or populations or also defining it for an intended application. These have been elaborated to state changes followed over time brain development, to establish a comparison across individuals of modalities or states for an automatic functional/anatomical labeling. In this work, the Talairach Atlas mapping of the brain [46], also known as the stereotaxic coordinate system, is put into perspective. It is known to be widely used in stereotactic and functional neurosurgery, brain mapping, neuroradiology, medical image analysis and neuroscience education. In our implementation, the spiking neurons in the 3D SNNcube are spatially located according to the brain template Talairach stereotactic atlas [46] by setting 1471 leaky integrate-and-fire (LIF) neurons each representing 1 cm$^3$ of brain tissue. In terms of number of neurons the initialization of the SNNcube is scalable from hundreds to hundreds of millions of neurons [14]. A visualization of the input neurons matching the number (4) of sEMG bipolar channels is presented in supplementary videos 1, 2 and 3. It is worth noting that we noticed a decrease in the performance in most of the simulations with dispersed input neurons. After the neurons are positioned, the synaptic connections are initialized in a small world connectome (SWC), which means that close neurons are connected with a higher probability than neurons that are further apart [14]. For every neuron pair $N_i = (x_i, y_i, z_i)$ and $N_j = (x_j, y_j, z_j)$, the Euclidean
distance is computed, then, a connection probability $P_{i,j}$ is calculated according to Equation 3.

$$P_{i,j} = \begin{cases} C \times e^{-(d_{i,j}/\lambda)^2} & \text{if } d_{i,j} \leq d_{\text{thresh}} \\ 0 & \text{otherwise} \end{cases}$$

(3)

with $C$ specifying the maximum connection probability, $d_{\text{thresh}}$ representing the maximum relative distance for connected neurons and $\lambda$ defining the small world connection radius. A $\lambda$ of 2.5 units was found to be the most effective for the 3D cube initialization.

2.3.3 Unsupervised training using spike timing dependent plasticity (STDP)

The traditional Hebbian implementation from [47] is used, where weight updates are calculated based on relative spike timing. From the perspective of a synapse, if a presynaptic spike is shortly followed by a postsynaptic spike, potentiation occurs; while reversal of this spike ordering leads to depression. The magnitude of this weight change is calculated according to Eq. 4, where $A$ defines the learning rate, and $\tau$ the time window over which plasticity is active. Only consecutive pairs of spikes are considered during the plastic update, and the weight change is made in a multiplicative manner by scaling the original weight within the limits $w_{\text{min}} \leq w \leq w_{\text{max}}$. This learning rule is implemented on the SpiNNaker hardware according to the pre-event driven model described in [48].

$$\Delta w_{\text{post}}(\Delta t) \over w_{\text{pre}} = \begin{cases} A_+ \times e^{\Delta t/\tau_+} & \text{if } \Delta t < 0 \\ -A_- \times e^{-\Delta t/\tau_-} & \text{if } \Delta t \geq 0 \end{cases}$$

(4)

The weight change in STDP is illustrated in Figure 4. STDP training parameter definitions and their values can be found in Appendix 1.

Figure 4: Weight change in STDP. The relative weight change $\frac{\Delta w_{\text{post}}(\Delta t)}{w_{\text{pre}}}$ at a synapse due to a single pre- and post-synaptic spike with spike time difference $\Delta t$.

In general, we see that using STDP in the training phase allows neurons, which represent the sEMG electrode sites, to have more high-weight synaptic connections
than the other neurons. Overall, NeuCube applies STDP learning to strengthen synapses that are involved in the creation of class-specific neuronal connectivity sets and weakens connections that take no role in class-specific neuron pattern creation. Thereby, discriminative spatio-temporal correlations are amplified and isolated. Interestingly, the spatially meaningful neuron structure and its complex connectivity enable the SNNcube to filter class-specific spatio-temporal features from sEMG data by creating class-specific sets of firing neurons and isolating these sets with STDP-learning. The learned input patterns can be as deep as needed in terms of time points that are measured for every sample. In our experiments we have used 200 time points for each sample, represented by ~ 400 milliseconds of sEMG data. How deep in time the learned patterns should be depends on the task and the size of the SNNcube. There is no limit in principle, meaning that an SNNcube can learn longer time sequences, e.g. minutes, representing complex movements. This is a dramatic departure from the traditional sEMG machine learning methods that use feature-vector based learning.

2.3.4 Classification Module

The third stage realizes a classification/regression function $s : N \times B^p \to R^p$, which converts the activity of the SNNcube into output classes using readout function $r : R^p \to \{0, 1, ...N\}$, which is trained with and finally classifies the state vectors. In this stage, dynamic evolving SNN (deSNN) [49] was used due to its efficiency in spatial and temporal pattern recognition as well as its accurate spatial-temporal event prediction. Overall, the deSNN network is based on the comparison between the synaptic weights of a newly created output neuron that represents the new spatial-temporal pattern for recall, and the connection weights of the existing neurons created during training. The deSNN algorithm comprises two phases: the training phase and the classification phase. In the training phase, for each training sample, new output neurons are created and connected to the reservoir neurons. Then, their spike trains are fed into the reservoir. The deSNN sets the initial weight of the connections from the reservoir to the output neuron according to the rank order (RO) rule [50]:

$$w_{j,i} = \alpha \cdot \text{mod}^{\text{order}(j,i)}$$

(5)

where: $w_{j,i}$ is the weight from a presynaptic (reservoir) neuron $j$ to a postsynaptic (output) neuron $i$; $\alpha$ is a learning parameter that in a partial case it is equal to 1; $\text{mod} = (0, 1)$ is a modulation factor that defines the importance of the order of the first spike; and $\text{order}(j, i)$ is the order (the rank) of the first spike emitted from neuron $j$ to neuron $i$ among all neurons in the reservoir, and has an initial value of zero and increases according to the spike order of other reservoir neurons. The deSNN adapts a synapse $w_{j,i}$ at every simulation time according to:

$$\Delta w_{j,i} = \begin{cases} d_+, & \text{if } j \text{ spikes} \\ d_-, & \text{otherwise} \end{cases}$$

(6)
where $d_+$ and $d_-$ are positive and negative small values (drift parameters). After weight adaptation, the K-nearest neighbors (KNN) is trained on the provided data. In the classification phase, every new sample utilizes the same learning strategy as for output neurons and the new validation sample is assigned to the class that is predicted by the KNN classifier. Figure 5 visualizes the creation of a classification neuron and its class assignment by the KNN classifier. The KNN classifier can use different features to fit the data. It can use the initial or final weight vectors, the total number of spikes per reservoir neuron, or a user-specified combination of the aforementioned. Furthermore, the classification phase can be paused to train the model on new samples from potentially new classes.

Figure 5: Classification of a neuron C. The weights between classification neuron C and reservoir neurons $N_i$ is calculated. The resulting weight vector is compared to all weight vectors in the deSNN employing a KNN classifier. The class of the closest output neuron $O_j$ determines the class prediction of the sample.

### 2.4 Implementation of NeuCube on SpiNNaker

This subsection describes the implementation of the NeuCube model on SpiNNaker and its motivation.

#### 2.4.1 Motivation and Neuromorphic hardware architecture

NeuCube is a specialized spiking model for modelling and recognition of complex spatio-temporal signal correlations using a network of spiking neurons. The execution of such artificial networks in real-time means simulating the asynchronous spiking behavior and synaptic communication between thousands of neurons simultaneously. However, simulating SNNs on standard computers requires substantial computational overhead [51], largely annihilating the benefits of using spiking models in certain applications, such as prosthetic control. Neuromorphic computers are hardware systems [51] developed specifically to simulate large networks of spiking neurons in real-time with low energy consumption often by collocating memory and processing. Thus, neuromorphic
computers achieve a speedup and support massively parallel data exchange in a brain-inspired fashion. We utilized the cross-platform high-level language PyNN[52] to implement the NeuCube algorithm on the SpiNNaker platform. SpiNNaker [16] uses a highly-parallel architecture to simulate large SNNs efficiently. This hardware system relies on regular microprocessors to maintain the flexibility and scalability of ordinary computers and at the same time avoid the Von Neumann Bottleneck when simulating large numbers of neurons. A 4-node SpiNNaker machine consisting of 64 processors and capable of simulating up to 10,000 neurons, was sufficient to simulate our network. In fact, our choice of using the small SpiNNaker board was highly motivated by its compact size, and power efficiency, and therefore its suitability to be integrated in a future prosthetic device.

2.4.2 Implementation details

The NeuCube was implemented using the open source Python package for neural networks specification PyNN [52]. In addition, the KNeighborsClassifier class from the SciKit-Learn library [53] was used as part of the deSNN implementation. The implementation comprises three stages: an encoding stage, which converts sEMG and EEG signals into spike trains; a three dimensional brain-shaped recurrent Spiking Neural Network reservoir, which learns spatio-temporal correlations from the data; and an output classifier, which is trained on class-specific reservoir dynamics and predicts class labels for unknown samples. The spike encoding stage has been implemented in Python and runs on the host computer. It converts normalized sEMG and EEG samples into sets of spike trains. It should be noted that the TD algorithm described in Section 3.3.1 was chosen and implemented.

The 3D reservoir implementation was split into two parts: the generation of the network structure; and the simulation of the network, which includes both training and classification. The network generation, meaning the positioning and connection of neurons, was executed on the host computer. Thereafter, the network was mapped to and trained on a SpiNNaker machine using the sPyNNaker software [54]. The third stage employs a modified version of the deSNN algorithm described in Section 3.5 for the final classification task and can be separated into network generation and sample classification. Stage 3 runs on the host computer. First, each neuron in the output stage is connected to each reservoir neuron of the second stage. The weights are initially calculated using the RO rule [50] and are adjusted using the spike driven synaptic plasticity (SDSP) rule [55]. As a result, every output neuron contains three feature vectors of 1471 elements (the total number of reservoir neurons) based on its connections to the reservoir neurons: one for the initial weights, one that contains the total number of spikes on each synapse, and one for the final weights. These weight vectors represent the feature vectors, which are used as inputs to train a KNN classifier from the SciKit-Learn library [53].
3 Results

3.1 Hand pose recognition from sEMG

In this section, we show the obtained NeuCube classification results of four hand movements recorded from six different subjects. To benchmark our obtained results, we compared the obtained results with two different machine learning classifiers KNN and Support Vector Machine (SVM) using three different feature extraction methods, namely mean absolute value (MAV), root-mean square (RMS), Wavelength (WL), as well as with raw data directly. For classic machine learning classifiers, the principal component analysis (PCA) technique was performed with 95% preserved variance. Through all six subjects, a mean test accuracy of $(84.8 \pm 4.0)\%$ was achieved using NeuCube on raw sEMG data. Interestingly, the NeuCube model outperforms the KNN algorithm by a large margin (33%) while classifying the raw data and shows comparable performance to state of the art methods, which use hand-crafted features. Overall, the obtained results confirm the evidence proposed before in [18], which shows that pattern recognition with standard machine learning algorithms does not perform as well in classification tasks without hand-crafted features. Thus, it reinforces the usefulness of using SNN algorithms, and in particular the NeuCube model when working directly with raw data (as shown in Figure 6). More importantly, the obtained results for classifying raw sEMG data are compared favourably to similar work presented before in [18], where an overall accuracy of only 68% was reached for classification of six hand movements collected from a single subject. Results are depicted in Figure 6. It should be noted that a total of 432 epochs for each subject were used and the balanced accuracy was chosen as an evaluation metric for the trained model. Data were divided into 70% for training and 30% for testing. A 5-fold cross validation was performed on training data to validate the model. Figure 6 shows that the NeuCube model provides comparable approach to classic methods with hand-crafted features and outperforms KNN with raw data. As model performance cannot be assessed solely on classification accuracy, precision and recall performance measures were also computed. Results are presented in the confusion matrix in Figure 7. The confusion matrix presented in Figure 7 shows that the NeuCube model could classify the fist well. However, as shown in Figure 7, the 2-digit pinch, 3-digit pinch and hand-open movements were confused for all the subjects.

3.2 Force estimation

In addition to the recognition of different hand gestures, we also investigated estimation of the applied force (the force distribution across fingers), when executing the different hand movements. For that, a regressor implemented in the NeuCube software was used. Force values from groups of 200 consecutive time points from each subject, yielding a total of 34 extracted trials per session ($\sim 0.4$s) were fed as input into the NeuCube regressor. We wish to mention that the regression was performed similar to the aforementioned classification task. A root mean square error (RMSE) of 1.29 N
Figure 6: Comparison of the obtained classification results using NeuCube on the raw data and the K-nearest neighbors (KNN) and support vector machine (SVM) with three different feature extraction methods, namely wavelength (WL), root mean square (RMS) and mean-absolute value (MAV). The polar bar plot shows the accuracy range (mean ± standard deviation) achieved by the seven models for each of the six subjects. The lower panel subsumes for each algorithm the nine mean accuracies achieved, black bars indicate the median result.

(rRMSE=18.8%) was obtained. Both the TD and BSA were tested and overall, the BSA algorithm provides better performance compared to the TD. BSA [45] was used instead of the TD algorithm to encode the force values with a fixed threshold of 0.5N.
Results are shown in Figure 8. To measure the performance of the validation set, the RMSE was used.

### 3.3 EEG results

To further evaluate the performance of the proposed spiking model, we tested the NeuCube model on the Graz EEG data set B from the BCI Competition 2009 [56]. The data consist of three bipolar recordings (C3, Cz, and C4) sampled at 250 Hz and two classes, namely the MI of left and right hand. A notch filter at 50 Hz was applied in order to remove power line interference. Second, data were band-pass filtered between 2 and 60 Hz using a 5th order zero-phase Butterworth filter. Epochs of 4 seconds were extracted, yielding a total number of 300 trials. Figure 9 shows that the spiking model outperforms both LDA, Naive Bayes and logistic regression on raw EEG data for all nine subjects and shows a comparable performance to the logistic regression with alpha and beta band power features. The same approach used for sEMG when dividing data into training, validation and test was used for EEG data. Through all subjects, a mean test
Figure 8: Regression results for force estimation. The pairs plot shows the histogram and the scatter plot. The histogram on the diagonal allows to see the distribution of each single variable while the scatter plots show the relationship (or lack thereof) between the predicted and true forces.

accuracy of 75% was achieved using the NeuCube model on raw data. For individual subjects, mean accuracies of 94.87%, 88.46% and 88.89% were attained for subjects B04, B05 and B09, respectively, whereas a mean accuracy between 58% and 77.5% was reached for the other six subjects. We wish to mention that the same parameters and configuration used for sEMG was chosen for the EEG decoding task.

3.4 Neuromorphic Results

In this section, we present the overall obtained results of the implementation of the NeuCube model on SpiNNaker for sEMG signal decoding. It is important to mention that model calibration remains a major challenge, when testing a complex system, such as NeuCube. Due to the complex interplay between different stages, and the multitude of parameters involved in the process as well as the probabilistic uncertainty that is introduced with every reservoir creation, it becomes hard to analyze
and establish the influence of individual parameters on overall system performance. Hence, the functionality of individual stages of NeuCube was tested first to validate our implementation. Thereafter, the performance of the NeuCube model as an sEMG classifier was validated. The recorded sEMG signals collected from six different subjects and sampled at 512 Hz were used to assess performance. The same preprocessing
described before in Section 2.2, was performed on the recorded data. 200 data points were fed into the encoding stage of NeuCube. The input neurons were positioned to maximize the spatial distribution within the reservoir. Noticeably, the different hand poses induced different amplitude patterns on the selected electrodes, which resulted in a clear and perceivable separation within the reservoir. Although the general class-specific firing pattern was clearly visible during the network’s simulation, there were samples that did not match their respective class pattern, which could be explained by some encoding errors. Moreover, it is important to highlight that minor parameter changes resulted in a worse classification performance, which shows the importance as well as the effect of network parameter tuning. Data were divided into 70% for training and 30% for testing. A 5-fold cross validation was performed on the training data to test the capability of the model. After a brief period of network tuning, a mean accuracy of 77% across all subjects was achieved. Overall, obtained results with NeuCube on SpiNNaker are inferior to those obtained on a standard PC (77% versus 84.8%). One intuitive reason for that is the higher number of parameters to tune in our NeuCube-SpiNNaker implementation compared to the small number of high-level parameters in the provided Matlab-NeuCube software, tested on a conventional computer. Given that the neuromorphic results were attained without any fine-tuning of the parameters, better performance is undoubtedly achievable with further optimization of the NeuCube’s implemented modules. It is important to highlight that the neuromorphic implementation runs in real-time with about 1-4W power draw on SpiNNaker.

4 Discussion

4.1 NeuCube is a suitable model for classification and regression of hand movements and associated forces from raw sEMG and EEG data

In this work, we showed that spiking neural networks, particularly the NeuCube spiking model, could be used to efficiently and accurately decode different hand poses as well as related finger forces from sEMG without hand-crafted features. Additionally, we showed that NeuCube was able to classify motor imagery (MI) movements without hand-crafted features. Several classic decoding algorithms have been proposed and used in the past [57]. In addition, Lobov et al. [28] proposed a hybrid network model involving bio-mimetic spiking neurons for extracting features and a classic artificial neural network for pattern classification. However, the novelty of this work resides in the direct decoding of recorded time-series signals without extracting discriminative features using a pure biologically inspired decoding algorithm, thereby allowing for the whole spatio-temporal pattern of hand movement to be learned in a deep learning mode. Overall, our obtained results do not appear to be in accordance with previous results in [14] where an accuracy of only 68% was achieved (84.8% in this work). Aside from the higher number of classes (6 classes) used in that study, another intuitive reason for
this difference, is the fine-tuning process of NeuCube’s parameters. We observed that these parameters have to be carefully chosen and tuned in order to reach good results, which could explain the difference in the obtained results between the two studies. Hence, we empirically selected and tuned the model parameters during the conduction of this work. Overall, results showed that NeuCube clearly outperformed the classic KNN machine learning algorithm when used without hand-crafted features, and showed comparable performance to the KNN with well-known discriminative features for sEMG decoding. Furthermore, the NeuCube outperforms classic machine learning methods when classifying raw EEG data and provides similar performance to state of the art band-power features with the logistic regression classifier. Notwithstanding these good results, an intra-variability between the six sEMG participants is clearly visible, which is in accordance with previous studies on sEMG signal classification [58]. Sweat, slight changes of electrode position during recording, muscle fatigue and the difference between muscle tissues of the participants could drastically decrease the quality of the recorded sEMG signal and therefore deteriorate classification accuracy. Last, we estimated the applied finger pressure force when performing the different hand poses. Despite the good performance (relative error of 18.8%), it is thought considerable improvements could be achieved. Using more accurate pressure sensors as well as further optimizing the NeuCube regressor’s parameters should be addressed in future work.

4.2 Successful implementation of the NeuCube model on SpiNNaker neuromorphic hardware

To complete our investigation of the potential use of spiking models, we implemented the NeuCube on SpiNNaker, a massively parallel neuromorphic system. We validated the overall implementation by successfully classifying the same sEMG data described in the first part of this work. As shown before, neuromorphic computing provides real-time performance and lower power draw (1-4W) compared to the traditional computing approach, which shows lower speed (latency of 220 sec) and higher power consumption. Despite such advantages of neuromorphic computing, the use of spiking models on neuromorphic systems to decode time-series signals has been rarely investigated [41, 59, 60]. First, we encoded the recorded sEMG into spikes using the TD algorithm. In addition, the second stage was designed such that every neuron can conceptually represent some spatio-temporal information from the input data based on its connectivity and spatial arrangement. Overall, the simulation showed that the second stage was successfully validated and a separation between the different sEMG classes is clearly visible. Nevertheless, the model was able to classify the presented sEMG samples with significantly higher-than-chance accuracy and, thus, the overall system was conceptually validated. It should be noted, however, that the STDP learning rule was disabled in the NeuCube-SpiNNaker implementation. STDP requires more computation time, more resources and hence a larger SpiNNaker system, which is not suitable for our ultimate prosthetic hand control application. Given our successful decoding of
both EEG and sEMG data, we can expect that integrating both, collected during same
data collection process, could improve further the classification and regression results
and would allow for more complex movements to be learned. With this regard, one
appropriate future test would be to use the hybrid-brain-computer interface (hBCI)
by combining EEG and sEMG [61], which may further validate this implementation.
Additionally, it is important to highlight that since we have used PyNN [52], the same
implementation should run on other contemporary and future neuromorphic and neuro-
inspired hardware platforms [62, 63]. Last, as was shown previously in Section 1.1.2,
the huge number of applications where NeuCube could be used as well as the versatility
of such a model could make our proposed approach applicable across different domains.
Thus, further investigation of our provided proof-of-concept could inspire the use of a
NeuCube-SpiNNaker implementation in new different applications.

5 Conclusion

In this work, we used NeuCube spiking model to classify four hand poses and estimate
related finger forces from raw sEMG and EEG signals. In addition, a successful
implementation of the proposed spiking model on dedicated neuromorphic hardware
was performed. Overall, the NeuCube model showed better performance compared to
traditional methods and achieved a mean test accuracy of 84.8% among six different
subjects. Moreover, a mean accuracy of 75% was obtained when classifying motor
imagery movements from EEG collected from nine different subjects. In addition,
NeuCube’s regressor successfully estimated associated forces with a mean RMSE of
1.29 N (≈18.8%). Furthermore, the neuromorphic NeuCube implementation on the
SpiNNaker neuromorphic platform was successfully validated and the implemented
model reached a mean classification accuracy of 77%, when classifying the four classes.
Taken together, our results could accelerate the development of the next generation of
low-power, portable and intelligent prosthetic hands.

6 Supplementary materials

Supplementary Videos:
S1: https://www.youtube.com/watch?v=Vb224UDHNSU
S2: https://www.youtube.com/watch?v=fRZBH1QmEyA
S3: https://www.youtube.com/watch?v=fRZBH1QmEyA

The source code of the NeuCube-SpiNNaker implementation is made publicly
available at https://github.com/behrenbeck/NeuCube_SpiNNaker. All the sEMG
processing scripts are made publicly available at https://github.com/gumpy-hybridBCI.
7 Author contributions

All authors contributed to this research. ZT designed the research and wrote the paper. JB implemented NeuCube on SpiNNaker and assisted in writing the paper. CB and ZT did the recording and the processing of sEMG data as well as the implementation of the NeuCube on a standard computer. JR and AG contributed in the implementation of the NeuCube on a standard computer. CR, OR, JR and NK contributed in the implementation of NeuCube on SpiNNaker and assisted in writing the paper. PC, CR, OR, NK, SF, GC and JC revised the manuscript. JC supervised this research.

8 Funding

This work was supported in part by Ph.D. grant of the German Academic Exchange Service (DAAD). Research and development of the SpiNNaker software stack is supported by the EU ICT Flagship Human Brain Project (HBP SGA2 H2020 785907).

9 Acknowledgments

The authors would like to thank Remi Laumont and Maxime Kirgo for technical assistance.

10 Conflicts of interest

The authors declare that the research was conducted in the absence of any commercial or financial relationships, which could be construed as a potential conflict of interest.

11 Appendix

<table>
<thead>
<tr>
<th>STDP Training parameters</th>
<th>Definition</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Potential leak rate</td>
<td>Leak in membrane potential of a spiking neuron when it doesn’t fire</td>
<td>0.002</td>
</tr>
<tr>
<td>Firing Threshold</td>
<td>Threshold membrane potential beyond which the neuron fires a spike</td>
<td>0.5</td>
</tr>
<tr>
<td>STDP rate</td>
<td>Learning rate of the STDP learning</td>
<td>0.01</td>
</tr>
<tr>
<td>Training round</td>
<td>Number of iterations for unsupervised learning in the cube</td>
<td>1</td>
</tr>
<tr>
<td>Refractory time</td>
<td>Absolute time (in time units) during which the neuron will not fire</td>
<td>6</td>
</tr>
<tr>
<td>LDC probability</td>
<td>Probability of creating a long distance connection</td>
<td>6</td>
</tr>
</tbody>
</table>
References


Classification and Regression of Spatio-Temporal Signals using NeuCube


[35] Enmei Tu, Nikola Kasabov, Muhaini Othman, Yuxiao Li, Susan Worner, Jie Yang, and Zhenghong Jia. Neucube (st) for spatio-temporal data predictive modelling with a case study on ecological data. In Neural Networks (IJCNN), 2014 International Joint Conference on, pages 638–645.


Classification and Regression of Spatio-Temporal Signals using NeuCube


