Spike detection and sorting with deep learning

SPIKE DETECTION AND SORTING WITH DEEP LEARNING

Melinda Rácz[1, 2 *], Csaba Liber[1], Erik Németh[2], Richárd Fiáth[2, 3], János Rokai[2, 5], István Harmati[1], István Ulbert[2, 3], Gergely Márton[2, 3, 4]

1 Department of Electrical Engineering and Informatics, Budapest University of Technology and Economics, Budapest, Hungary

2 Faculty of Information Technology and Bionics, Pázmány Péter Catholic University, Budapest, Hungary

3 Institute of Cognitive Neuroscience and Psychology, Research Centre for Natural Sciences, Hungarian Academy of Sciences, Budapest, Hungary

4 Doctoral School on Materials Sciences and Technologies, Óbuda University, Budapest, Hungary

5 Károly Rácz School of PhD Studies, Semmelweis University, Budapest, Hungary

* Corresponding author (e-mail: racz.melinda.9157@gmail.com)

Keywords: spike detection, spike sorting, spike prediction, LSTM, recurrent neural network, convolutional neural network

ABSTRACT

1. Objectives

The extraction and identification of single-unit activities in intracortically recorded electric signals have a key role in basic neuroscience, but also in applied fields, like in the development of high-accuracy brain–computer interfaces.

The purpose of this paper is to present our current results on the detection, classification and prediction of neural activities based on multichannel action potential recordings.

2. Approach

Throughout our investigations, a deep learning approach utilizing convolutional neural networks and a combination of recurrent and convolutional neural networks was applied, with the latter in case of spike detection and the former in cases of sorting and predicting spiking activities.

3. Main results

In our experience, the algorithms applied prove to be useful in accomplishing the tasks mentioned above: our detector could reach an average recall of 69 %, while we achieved an average accuracy of 89 % in classifying activities produced by more than 20 distinct neurons.

4. Significance

Our findings support the concept of creating real-time, high-accuracy action potential based BCIs in the future, providing a flexible and robust algorithmic background for further development.

INTRODUCTION

Brain–computer interfacing (BCI) is a quickly developing multidisciplinary field aiming to bridge the gap between the (human) central nervous system and some environment, enabling the user to control actuators through an artificial medium. To establish robust connection to the central nervous system, high-fidelity control signals are needed [1].

According to our prior knowledge, control signals of the highest quality can be obtained through intracortical recordings. It has been proven that high-density microelectrode arrays (MEAs) provide measurements of high accuracy and the recorded signals can be applied for real-time control of an artificial arm prosthesis [2]-[3]. There is an active research aiming to create more flexible, biocompatible and longer-lasting probes, with results promising safe and robust chronic neural implants [4]-[5].

The waveforms recorded are a combination of action potentials of neurons in the vicinity of the recording site and the cumulated activity of neurons farther from the probe (local field potential or LFP) [6]. While the LFP activity carries relatively small amount of useful information for BCI purposes, the single unit activity (SUA) of the nearby cells can be used to provide robust control for a BCI at a high information transfer rate. Studies show that cells of the motor cortex are cosine-tuned to the direction of motion [7] and the direction [8] and the speed [9] of a movement can be calculated using the SUA from different cell groups in this region. These findings suggest that the key to building high-accuracy BCIs is the efficient detection of neural activities and the algorithmic extraction of SUAs from the recorded intracortical signal.

The algorithmic extraction and categorization of the distinct action potentials have been addressed by many scientists under the name “spike sorting” [10] [11]. The usual structure of sorting algorithms is multi-layered.

Firstly, the detected wideband signal is filtered (usually between 300 and 3000 Hz) to clean data of the lower frequency components corresponding to the LFP and the high-frequency noise. In order to find the action potentials, most of the spike detection algorithms use thresholding [12] [13], where the threshold is given in relation to the estimated standard deviation of the noise [14]. Thresholding has the advantage of being simple, with the drawback of being sensitive to noise. More sophisticated spike detecting methods are also present in the field, such as non-linear energy operator thresholding [15], Teager energy operator thresholding [16], and wavelet decomposition [17].

The second phase consists of feature extraction where the most common feature extraction/dimension reduction method is the principal component analysis (PCA) [18]. PCA captures the largest variations in the data and returns it in a set of orthogonal basis. In this set, every vector contributes to the original data in a proportion given by its eigenvalue. PCA is used alone [19] [20] [21] or in conjunction with other feature extracting methods like Wavelet coefficient [22]. Several other advanced feature extracting methods are used as well, such as wavelet packet coefficient [17] or Wavelet packet decomposition, where the best basis is evaluated by a support vector machine [23].

In the next phase of spike sorting, the extracted features are assigned to a label using a classification algorithm. Those algorithms which assume Gaussian distribution of the feature map perform well only in special cases, when the presented spikes have stationary shape along with uncorrelated noise [11] and high signal-to-noise ratio (SNR). In contrast, methods that do not assume the distribution to be normal tend to cluster spike activity more accurately even in case of lower SNRs. The deviation of the activity distribution from the Gaussian originates from multiple phenomena, such as neuronal burst activity or the displacement of the electrodes in respect of a particular neuron [17]. A commonly used clustering method is the k-means clustering which is used alone [24] or combined with various other methods like template matching [20][12] or Independent Component Analysis [25]. Beside the methods mentioned above, a number of other methods have been applied, e.g. Bayesian classification [11], superparamagnetic clustering [17] or support vector machine classification [26]. Lately,

deep learning was also applied in the form of PCANet [27], however, it did not yield a better sorting performance than the methods mentioned above.

As MEA development facilitates an exponential growth of the number of distinguishable single units, the need for fast and accurate automatic spike sorting algorithms rises [28]. Besides implementing complex solutions to the problem of spike sorting, most of the algorithms require the fine-tuning of several hyperparameters. Measures against the complex and time-consuming hyperparameter-tuning have been taken in form of the development of a fully automated algorithm [29]. While adequate offline detection and classification of action potentials is possible with concurrent technologies [30], there are several issues yet to be overcome for more robust operation. Bursting neurons produce a rapid train of action potentials with decreasing amplitude thus complicating both the detection and the classification of their spikes. Complex waveforms might also appear due to the overlapping of separate single-unit activities. Furthermore, brain-computer interfacing requires devices capable of real-time operation.

In this study we would like to introduce a novel way of spike sorting utilizing convolutional neural networks. Deep neural networks have been tested and shown to outperform the most common unsupervised methods in case of intracranial EEG recordings [31]; and both convolutional [32] and recurrent neural networks [33] were tested on EEG data classification and visualization. Machine learning was also successfully used to provide robust control of a deep brain stimulating device depending on real-time cortical signals [34]. An interesting application of convolutional networks can be found in [35]. The network presented in the paper (SpikeDeeptector) is used for detecting and tracing channels that actual signal can be recorded on, in contrast with the other ones, yielding only artifacts. This approach differs from ours, as our scope was detecting and sorting spikes recorded on all the channels (focusing on the individual spikes instead of the individual channels). Another important difference of our work is that we exploit the spatial information provided by the placement of the electrodes, while SpikeDeeptector uses convolution only for discovering time dependencies along the spike/artifact samples and possible dependencies between different samples. Additionally,

SpikeDeeptector is a convolutional/dense detector, while we use LSTM features for this purpose and CNNs for sorting and predicting activities produced by known units.

Our aim was to create two separate systems: one for the detection of the spiking events and another one that could classify the found spikes using the raw data for input; we also wanted to know if activities can be foreseen (given that neural networks are usually good predictors). We expect that these units would be able to function as integral subsystems of a more complex classifier application providing one step towards future on-line solutions.

METHODS

The general outline of the approach we applied is depicted in Figure 1. The steps mentioned in the figure are described in details throughout sections Methods and Results.

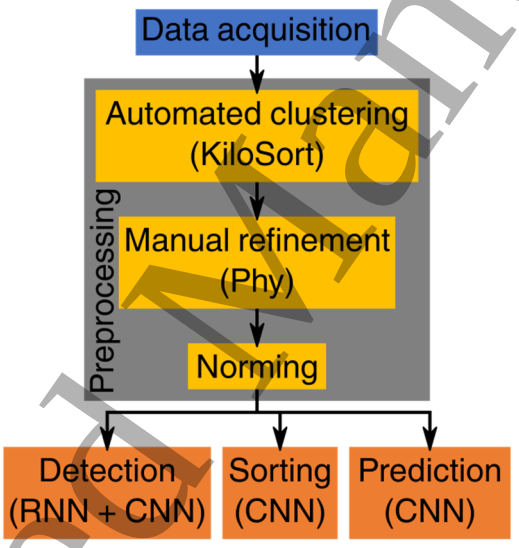

Figure 1. The subsequent steps of our approach. First, data were acquired from the cortex of anesthetized rats using high-density neural probes. Following this, ground data were produced via automatic/manual clustering. These data were feed into the networks after being normed.

1. Data acquisition

In this study, parts of the database obtained in [36] were used (n = 9 high-density cortical recordings, each containing 23–46 well-isolated single units). These datasets were produced using silicon probes with single, 8 mm × 100 µm × 50 µm shanks (length × width × thickness) [37]. These high-density probes contain 128 square-shaped recording sites of dimensions 20 µm × 20 µm with 2.5 µm spacing between the edges of the sites, forming a 32 × 4 sensor array. The densely packed electrodes on the probe allow the recording of the action potentials

('spikes') of single neurons with high spatial resolution. The recording sites have an impedance magnitude of about 50 kΩ measured at 1 kHz [37]. From this point on, the 4 × 32 potential values detected by the recording sites of the probe at a particular time instance would be referred to as a *frame*.

The 128-channel probes were inserted into the trunk region of primary somatosensory cortex of ketamine/xylazine anesthetized Wistar rats (n = 10) to a dorsoventral depth of 1700 µm. This allowed us to obtain the neuronal activity from deeper layers (layers IV–VI) of the cortex, primarily from layer V. In the case of the database used here, the insertion process was taking place at a slow speed (0.002 mm/s) in order to record neuronal signals with high quality. According to [36], neuronal recordings obtained after using this slow insertion speed resulted in the highest and most stable signal-to-noise ratio, the highest single unit yield and the largest spike amplitudes. Additionally, the neuronal loss in the vicinity (< 50 µm) of the probe was found to be the lowest [36]. As reference electrode, a stainless steel needle (inserted in the neck muscle of the animal) was applied.

Wideband (0.1–7500 Hz) spontaneous cortical activity was recorded continuously for 45 minutes after implantation, using an RHD-2000 electrophysiological recording system (Intan Technologies, Los Angeles, CA, USA) connected to a laptop via USB 2.0 data link. The sampling rate on each channel was 20 kHz, with 16-bit resolution.

Given that neural networks require labeled data for deep supervised learning, detected single units of the training and validation sets had to be classified. We obtained ground truth data using a MATLAB-based application for automatic spike sorting called KiloSort [38]. KiloSort is widely applied for spike sorting and benchmarking other sorting algorithms, being able to deliver good results within reasonable run time. Its performance, however, is affected by the parameter settings (i.e. detection threshold, required number of clusters to be distinguished) making revision (merging/splitting certain clusters) by a human expert often necessary. We performed the manual revision phase using a Python-based neurophysiological data analysis package called Phy. Note, that since multiple decisions are made during the whole process involving humans as well, some amount of error/bias is inevitable (e.g. spikes assigned to the wrong cluster, or, with their amplitudes below the threshold, being not detected at all).

The results of the clustering process (i.e. the ground data) were the exact time instances of the negative peak of the spikes (these time instances would be referred to as *marks*) for each single unit cluster in a different file. The spatiotemporal characteristics of the spike waveform of a representative neuron cluster are shown in Figure 2.

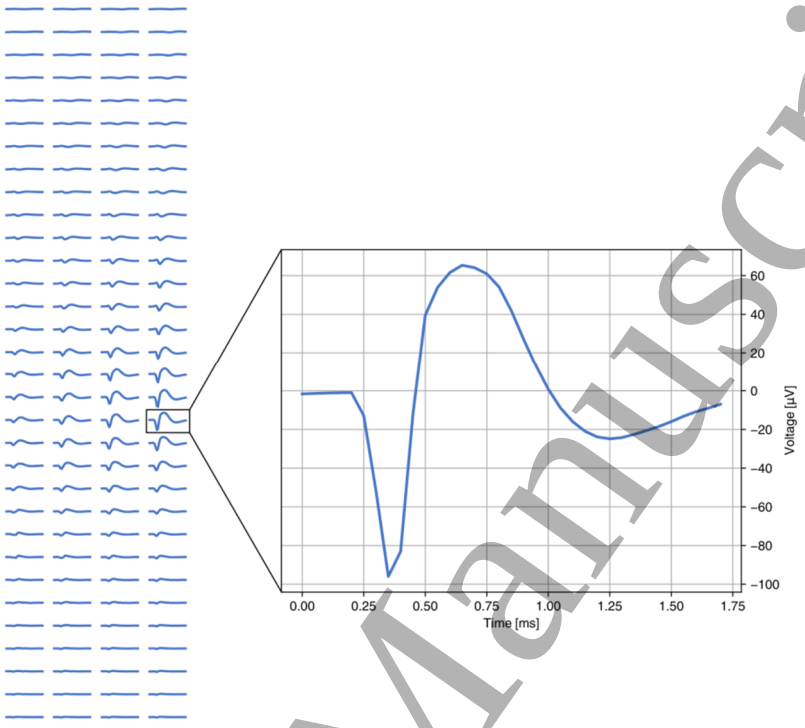

Figure 2. The spatiotemporal profile of the average action potential waveform of a neuron recorded with the 32 × 4 sensor array. The insert on the right shows the spike waveform on the channel which recorded the spike with the highest amplitude.

2. Environment

All the models were written in Python, using TensorFlow, a package developed for the modeling of neural networks and deep learning on CPUs or GPUs.

Sorters and the predictor were implemented utilizing Keras (a high-level API to improve readability and maintainability), while the detector was written using pure TensorFlow.

3. Convolutional neural networks

Convolutional neural networks (CNNs) [39] are neural networks containing convolutional layers that store the representation of the data in the form of convolutional filters. These filters (kernels) slide along the input of the layer, performing a template matching task. Every convolutional layer has its own set of filters (the number of these filters is called the depth of the layer). Data representation is performed in a hierarchical way. Every single kernel is able

to recognize a certain feature, with filters belonging to layers closer to the input detecting simple objects (e.g. edges), while filters in layers closer to the output are capable of extracting more abstract or complex patterns (e.g. faces).

Convolutional layers are often followed by maximum or average pooling layers (usually max pooling applied); these layers implement downsampling (additional dimension reduction) by means of returning the maximum or average value in a small region. Another type of downsampling could be achieved by sliding the kernel along the input taking greater steps (calculating convolution for e.g. every second value of the input), called stride.

CNNs apply fully connected (FC) layers as well, to perform the actual classification of the single instances in the feature space. They are followed by an output layer, generating a one-hot array with as many dimensions as the number of distinct categories available.

4. LSTM cells

Recurrent neural networks (or RNNs) are neural networks in which each cell has an extra feedback loop besides the synapses from the previous layer. This input serves to let the neuron remember its previous output creating a one-iteration long memory.

LSTM (Long short-term memory) cells [40] are special neurons remembering their earlier states by exhibiting an automatically tunable “input”, “output” and “forget” gate filtering the components that will determine the inner state of the cell at a given instance. Using this method the network is also capable to explore and remember long-term dependencies in the data as well thus making the network more robust against outliers in the input.

Combining LSTM cells with convolutional layers one can create networks that are able to extract and classify the major spatiotemporal features of the input. This setup was used for the detection of the spiking activity in the acquired data.

5. Performance metrics

In section Results, three metrics will be used in relation to performance achieved using test data. Positive prediction rate (precision) and recall will be referenced in subsection Detection, while accuracy will be mentioned in Sorting and Prediction.

The *positive prediction rate* of a detector is defined as the quotient of the number of true positives (*TP*, the detected activities) and the number of all selected elements (all the actual activities and some noise erroneously labeled as activity – false positives, *FP*). It is given analytically by equation (1).

$$P = \frac{TP}{TP + FP}. \tag{1}$$

The *recall* of a detector is defined as the quotient of the number of true positives and the number of relevant elements (all the actual activities including the ones detected and the ones erroneously labeled as noise – false negatives, *FN*). It can be calculated using equation (2).

$$R = \frac{TP}{TP + FN}. \tag{2}$$

The *accuracy* of a sorter is defined as the quotient of the sum of the number of the true positives for each cluster and the sum of the number of the activities for each cluster (i.e. a generalization of recall). For its concise form, see equation (3).

$$A = \frac{\sum_{i=1}^{N} TP_i}{\sum_{i=1}^{N} \left(TP_i + FN_i \right)}, \tag{3}$$

where *N* is the number of clusters.

RESULTS

1. Detection

As it has been mentioned, our detector was implemented as a combination of a CNN and an RNN. The input of the system was a data matrix of 32×4 normalized potential values. The spatial feature extraction was performed by 2 consecutive convolutional layers, each having a kernel size of 5×2 and strides of 1×1, extracting 16 and 256 different feature maps (for the first and second layer, respectively). The feature extraction was followed by a max pooling step (kernel size: 2×2, stride: 2×2) producing a feature vector of 3072 dimensions. These kernel sizes were chosen accordingly to the spatial extension of the single unit activities. The consecutive steps and the sizes of the data structures produced are displayed in Figure 3.

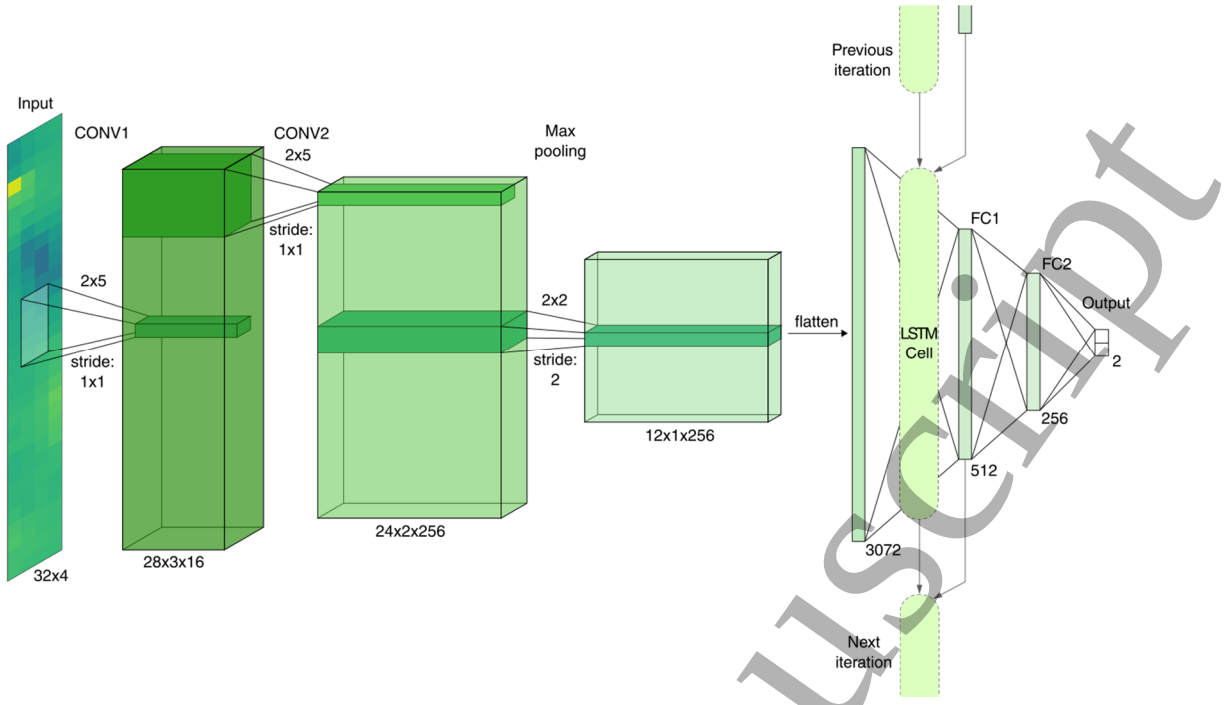

Figure 3. Network architecture for detection.

The LSTM cell of 512 units was trained on these vectors, being followed by a fully connected (FC) layer of 256 neurons. To prevent overfitting, a dropout layer switching 20 % of the neurons in each epoch off was added between the fully connected layers. The output was generated through an FC layer consisting of 2 neurons (one for each category), using a softmax nonlinearity and thus yielding an estimate of the likelihood of each frame containing a spiking event.

For training the LSTM, truncated backpropagation through time (TBPTT) was used [41] with timesteps of 20 datapoints. Training of the network was preliminary tested with 3 different TBPTT steps: 10, 20, and 50. The preliminary tests, which were performed on the nr. 1 dataset showed that TBPTT with 20 timesteps was the most optimal for training our network. According to this, a training sample of 20 datapoints was used.

The training consisted of multiple phases: to validate our network structure and the initial results, a 10-fold cross-validation was performed on the first dataset. The cross-validation training consisted of 100 epochs, with 20 timesteps. The mean recall and positive prediction rate of the cross-validation was 84%, and 30% respectively. The detailed results of the cross-validation process are displayed in Table 1.

K-Fold	1	2	3	4	5	6	7	8	9	10	Mean
R	86.84	86.75	85.63	82.99	84.23	83.50	84.49	83.17	80.57	80.09	83.83
P	18.95	22.34	25.42	23.01	24.89	31.72	33.5	34.76	38.32	43.34	29.62
TP	4542	5468	5783	5857	6060	7376	8054	8387	8442	8786	6875
FP	19421	19002	16960	19595	18279	15877	15984	15738	13587	11486	16592
TN	75348	74694	76286	73347	74526	75289	74483	74177	75935	77543	75162
FN	688	835	970	1200	1134	1457	1478	1697	2035	2184	1367

Table 1. Results of the 10-fold cross-validation after evaluation. For evaluating the results, 10 positive predicted values were considered a detected spike. In the original dataset, every detected spike by KiloSort was labeled as positive in 5 consecutive data points. Our experience was that the network predicted longer positive consecutive sequences; this phenomenon was a manifestation of the great sensitivity of the network to spike-like events. To correctly evaluate the spike-detecting capability of the network, we adjusted the evaluation process accordingly in the evaluation phase of the test dataset.

In the second phase, we trained the proposed network on each of the remaining 8 datasets; each instance of training consisted of 100 epochs on 10,000,000 data samples (8 minutes 20 seconds long recordings) subdivided into training (90 %) and test data (10 %).

The positive-negative data ratio was computed for each of the datasets. Data were divided into 90 batches with each consisting of 20 datapoints according to the number of timesteps used for TBPTT; training samples including no positive frames were treated with less significance, being multiplied by the positive-negative data ratio. This step was necessary because during the early tests the network exhibited a very strong negative bias due to the relatively small number of positive samples.

However, the effectiveness of the previously mentioned method was limited, as under a certain positive-negative data ratio, the network started to exhibit the strong negative bias experienced at the initial trainings. We found that beside the positive-negative data ratio, the mean of the cluster quality given by KiloSort was also a determining factor of the network's capability for learning. The results for all the datasets are shown in Table 2.

Dataset	1*	2	3	4	5	6	7	8	9	Mean
R	83.83	80.87	49.71	81.86	43.20	56.36	73.65	84.36	72.88	69.63
P	29.62	38.74	28.10	6.47	17.87	15.80	14.07	7.49	13.51	19.07
TP	6875	6305	2656	1715	2749	3182	3740	2832	4374	3825
FP	16592	9970	6795	24788	12627	16955	22828	34969	27996	19280
TN	75162	82234	87863	73117	81010	77400	72094	61674	66003	75173
FN	1367	1491	2686	380	3614	2463	1338	525	1627	1721
Spike ratio	0.043	0.034	0.018	0.014	0.021	0.025	0.020	0.007	0.021	0.023
Mean cluster quality	53.44	43.09	44.84	50.55	25.95	41.31	53.07	46.72	40.43	44.38

Table 2. Results for the different datasets. The network performed the poorest when the spike ratio and the mean cluster quality were both low. We found a strong positive correlation between the product of recall and precision, and the product of spike ratio and mean cluster quality (Pearson coefficient: 0.79).

*** - the mean of the results of the 10-fold cross-validation was taken.**

At this point, the misclassifications arising from the partitioning of the data into batches of input still had to be evaluated. We needed to keep in mind that the goal was to identify the intervals in which positive samples were present, instead of perfectly classifying every datapoint; thus taking the accuracy of each interval into account instead of focusing on the classifications of the sample could be a more adequate measure of performance in cases when our system finds the spike with a small offset compared to the labels generated by the KiloSort algorithm or it detects for longer period than as we trained on. An offset of this kind can be seen in Figure 4.

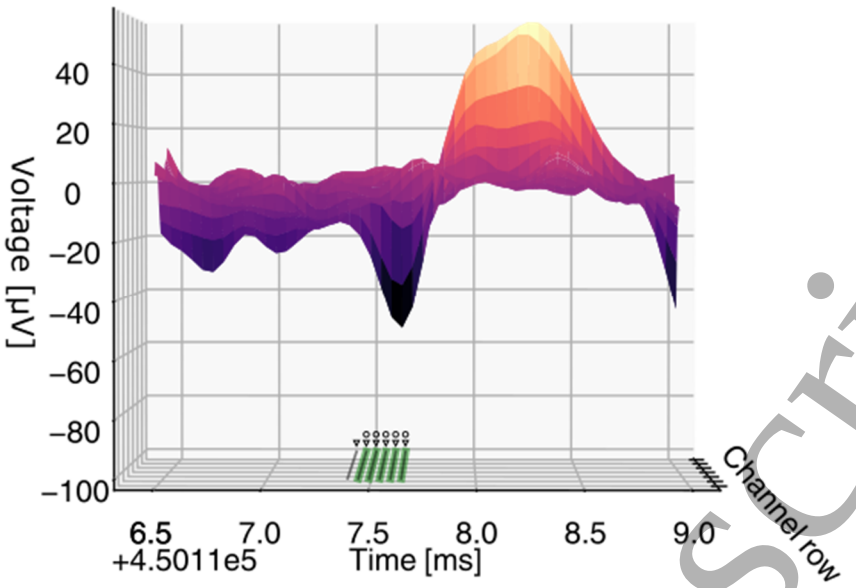

Figure 4. 3-D visualization of an action potential and the corresponding results of the detection. The marked positive interval (thick green lines, marked with small circles hovering above) and the predicted positive datapoints (thin black lines, marked with downward pointing triangles) do not overlap completely, distorting the accuracy of the measurement.

Knowing this our final system was assessed in a way that only a predetermined sequence (n=10) of positive instances were considered to be actual positive data, and we expected our labels to overlap with the ones produced by KiloSort, with this overlap spanning over a fixed size. Naturally this begs the question of temporal accuracy: the network has to be tuned further if the slight offset of the detected spikes or the longer detection time period would spoil the applicability of the system.

During the tests, using the evaluation method described above, we found that the system performed well in case of identifying the true positive intervals, exhibiting 69.63 % mean recall over all datasets in case of the previously determined test data. On the other hand, a large proportion of the positively classified data were actually considered negative by KiloSort, as our mean positive prediction rates lay around 19.07 %.

2. Sorting

The task of spike sorting was performed on activities represented by two- and three-dimensional data.

As two-dimensional data, frames taken at the time instances of spike maximum were used. These time instances roughly correspond to the marks produced by KiloSort, being placed

usually 1-2 frames after the actual voltage peaks. The exact amount of temporal displacement between the spike maxima and the marks is unknown and may vary from dataset to dataset (more precisely, from cluster to cluster). Because of this, sampling was shifted a few frames backwards (i.e. samples were taken a few sampling periods before the marks) in case of all the datasets. This shift is called *time offset*.

Three-dimensional data (called *timeslots*) consisted of a predefined number of frames—*length*—over an arbitrary time window. The difference of the mark and the time instance of the first frame in a particular timeslot (in sampling periods) shall be referred to as the *starting point* of the timeslot.

A diagram explaining the terms above is shown in Figure 5.

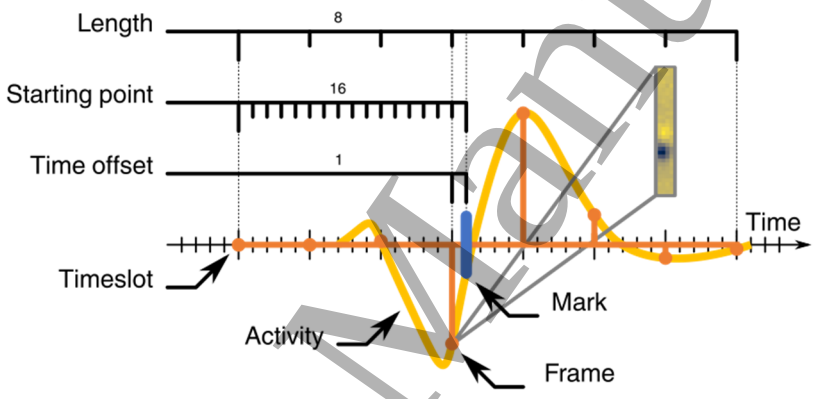

Figure 5. A detailed explanation of the terms used in relation to frames and timeslots. Here an activity is shown (yellow curve) accompanied by a corresponding timeslot (orange ticks). This timeslot consists of eight frames (with every fifth frame taken from the recording); the frame containing the maximum is shown in the insert. The mark is denoted by a blue tick; it follows the peak by one and the first frame of the timeslot by 16 sampling periods.

Signal pre-processing was intended to be simple and fast, therefore data were not filtered. In order to be made tractable, samples were normed between 0 and 1 or -1 and 1 (all the tests were performed on data normed both ways).

As regularization methods, early stopping with validation and batch normalization were applied. Only positive data (i.e. data corresponding to actual activities of some neuron) were used in chronological order: the first 50 % was applied for training, the first and second halves of the rest for validation and test, respectively. Thus, the performance of the networks in case of drift (slow changes in the environment of the probe) could be tested. We assume this yields

a more pessimistic but more realistic estimation than the one could be got using randomized data. To make training faster, samples were shuffled within the (training, validation, test) databases.

The CNNs applied consisted of one convolutional and two dense layers (with the second as the output layer). The padding of the convolutional layer was chosen to be "same", thus the dimensions of data did not change. The depth (i.e. the number of filters) of each layer was fit to the number of clusters (along with the output layer, thus the number of output categories and the number of clusters would be equal; since the latter was known, this could be done). The general structure of the network is shown in Figure 6.

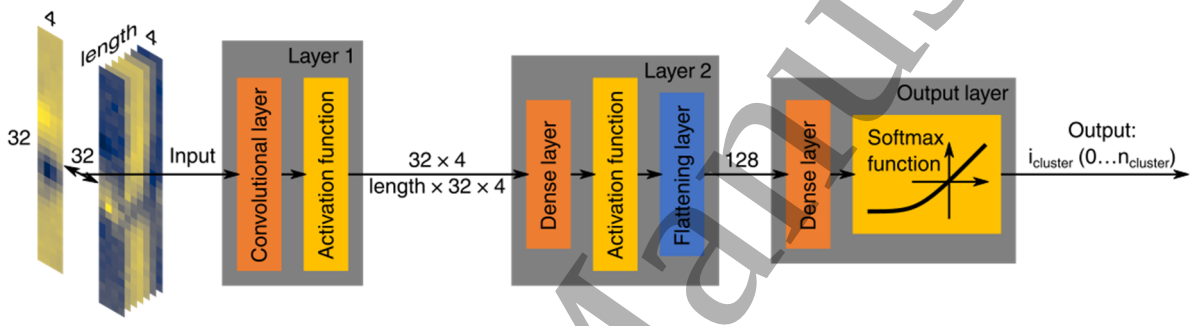

Figure 6. General structure of the networks used for sorting.

The networks we built were slightly different for 2-D and 3-D data processing, with respect to the kernel size of the convolutional layer, the optimizer and the loss; the options chosen can be seen in Table 3. In our experience, these sets of hyperparameters yield the best results given these particular tasks.

	2-D network	3-D network
Kernel size	16 × 2	timeslot length × 4 × 2
Optimizer	Adadelta	RMSprop
Loss function	Cosine proximity	Mean squared error

Table 3. Additional parameters of the networks used.

2.1. Sorting with 2-D input

Initial tests were performed on one of the datasets, containing 23 separate clusters. Different versions of the 2-D network were tested in the terms of nonlinearity, i.e. the activation function and the pooling; as possible variations, linear and rectified linear (ReLU) units and the

presence of 2 × 2 pooling were considered. The four combinations were tested on frames with offsets of 0, 1 and 2. The results are depicted in Figure 7.a).

We found that networks applying no pooling performed slightly better; assuming that activities follow a rather simple pattern and linear regression can be proven sufficient to find the underlying trends, linear activation unit was chosen for all the sorters.

In case of neural networks that use only linear activation functions, the question is raised whether the convolutional and the dense layer could be replaced by only one layer. Three networks were built with the first applying only one convolutional layer, the second having only one dense layer and third using the two type of layers together (all of these apart from the output layer). Only slight differences were experienced regarding the performance of the networks: the combined net yielded the best average accuracy, therefore this architecture was chosen. For details, see Figure 7.b).

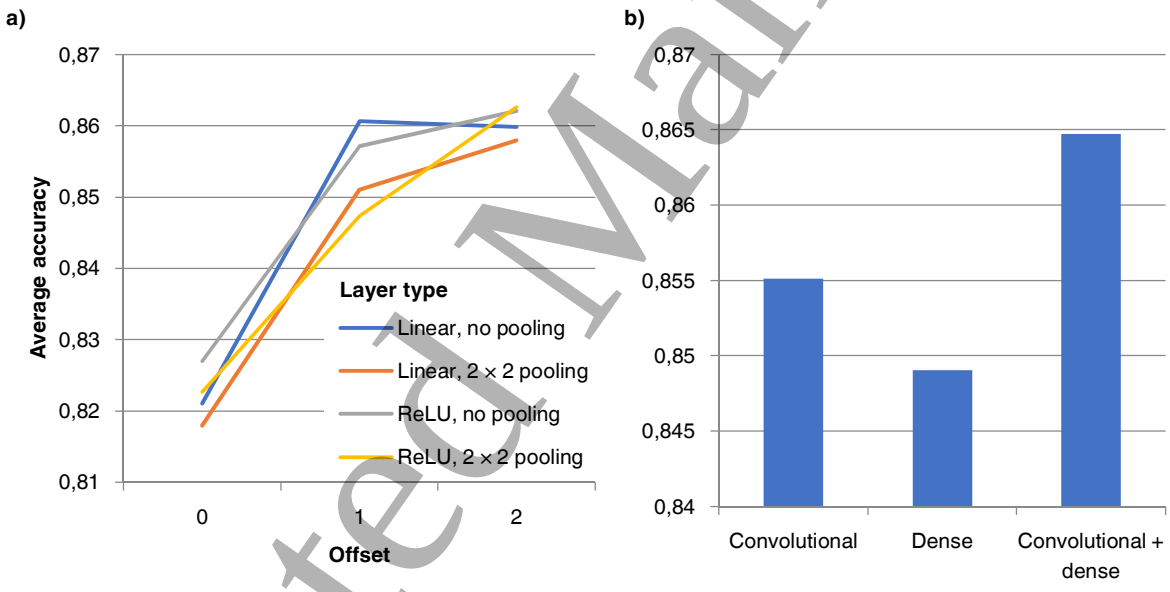

Figure 7. The results of the preliminary tests. a) The average accuracy of the neural network using different activation function/pooling combinations with respect to the offset. b) The performance (average accuracy) of the neural network with respect to the type of the layers applied.

The accuracy of the sorter is strongly dependent on the temporal position of the frame within the spike. The classifier was trained and tested on samples taken at 20 consecutive offset values.

The characteristic curve showing the accuracy with respect to the offset is depicted in Figure 8.

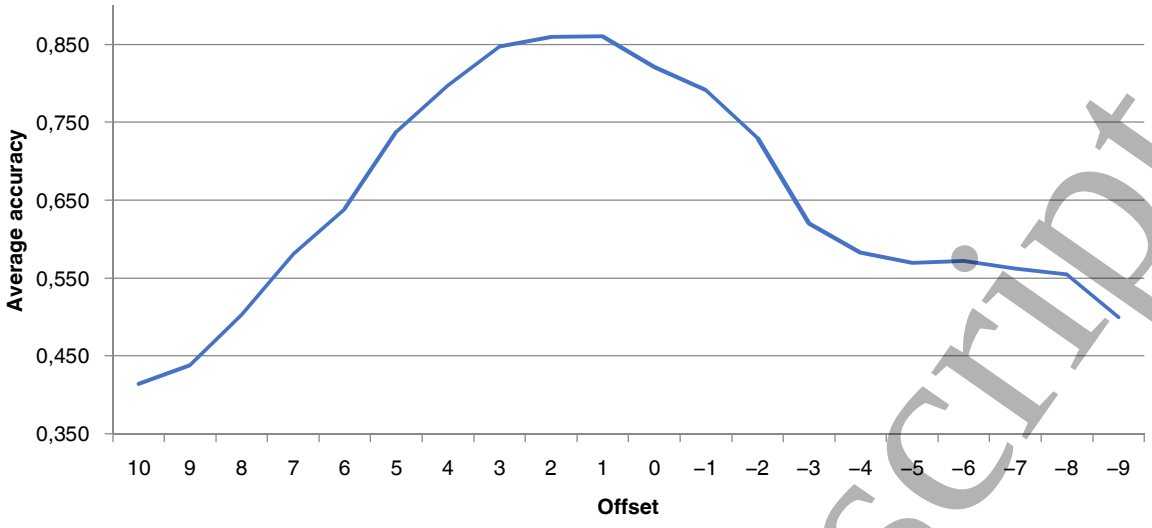

Figure 8. The average accuracy of the network with respect to the offset. At the first trial we used the frame preceding the mark by 10 instances, and in every subsequent test the succeeding frame was taken for training and testing.

According to the characteristics, we found that the best results could be achieved with frames having an offset of 1 or 2 depending on the dataset examined, with the expected accuracy of 0.86.

The greatest accuracy was 0.900 (applying an offset of 2); a confusion matrix is shown in Table 4.

Real \ Predicted	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22
0	2089	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1
1	0	821	0	0	0	0	1	0	4	0	0	171	0	0	0	4	0	0	0	0	0	0	0
2	0	1	312	0	0	0	0	3	1	0	0	0	0	0	0	0	0	2	1	0	0	0	1
3	0	0	0	3	0	0	0	1	0	0	0	0	1	0	0	0	0	0	0	523	5	0	0
4	9	0	0	0	1053	0	2	1	0	1	0	0	0	0	1	3	0	0	0	0	0	0	0
5	0	1	0	0	0	573	0	5	0	0	0	0	2	0	0	0	3	0	0	1	0	0	1
6	6	0	0	0	3	0	3219	0	57	1	0	0	0	0	1	0	0	0	0	0	0	0	0
7	16	0	0	0	3	0	7	1652	1	3	0	0	0	0	0	0	0	3	2	0	0	2	0
8	1	0	0	0	1	0	269	6	2086	1	0	2	2	0	1	0	0	0	0	0	0	0	0
9	4	0	0	0	1	0	2	1	1	472	1	0	0	0	0	0	0	1	0	0	0	0	6
10	2	0	0	0	2	0	5	5	1	0	491	0	0	2	0	5	0	1	0	0	1	1	3
11	162	0	0	0	1	1	7	4	5	0	1	1337	6	0	0	2	0	0	2	0	0	0	1
12	2	6	1	0	7	0	127	33	40	1	2	1	1165	1	3	25	2	6	9	0	0	7	17
13	0	4	0	0	0	0	1	2	0	3	1	0	4	332	1	0	0	1	0	0	0	0	4
14	3	0	0	0	1	0	6	1	0	0	0	0	0	0	520	0	25	1	0	0	0	0	0
15	16	0	0	0	8	0	34	28	11	2	1	6	2	1	3	3096	0	6	0	0	0	0	3
16	0	0	0	2	0	0	2	0	0	0	0	0	3	0	0	2	1171	0	0	2	231	0	0
17	9	0	0	0	5	0	16	4	3	1	0	0	0	0	1	3	0	960	0	0	0	0	2
18	8	0	0	0	1	0	8	25	13	3	2	0	5	0	3	10	0	3	475	0	0	7	4
19	0	0	0	0	1	0	0	0	0	0	0	0	0	2	0	0	1	1	2	369	3	0	0
20	1	299	16	0	1	0	0	11	0	0	0	0	27	8	2	5	1	2	1	5	1227	2	5
21	0	0	0	0	0	0	1	0	0	0	0	0	0	0	1	1	0	0	0	0	0	110	0
22	3	0	0	0	3	0	2	2	1	2	0	0	0	0	0	4	0	0	0	0	0	0	110

0 % 10 % 20 % 30 % 40 % 50 % 60 % 70 % 80 % 90 % 100 %

Table 4. Confusion matrix displaying the results of the 2-D network. The percentage of activities attributed to different neurons relatively to the number of action potentials of a particular single unit is

denoted by shades between blue and yellow; the color coding on the right applies to each row (i.e. each cluster) separately. The misclassified activities exceeding the 10 % of the total number of action potentials of a particular neuron are marked by thick borders.

The misclassifications arise mainly from the spatial proximity and morphological similarity of the particular clusters. The average frames for each neuron can be seen in Figure 9.

Comparing the ones the classifier was not able to sort properly (1 and 11, 3 and 19, 6 and 8, 16 and 20, 11 and 0, 20 and 1), they are not easy to distinguish visually even for the human eye. Note that the percentage of misclassifications is independent from the number of activities of the neurons, even clusters having only a few hundred activities can be sorted properly (assuming they are distinguishable from the others based on their spatial characteristics).

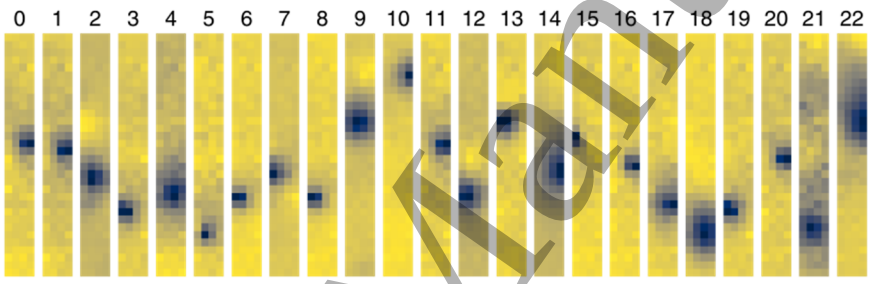

Figure 9. Frames the classification was based on (having an offset of 1). The figures were produced through averaging every activity of the clusters filtered above 300 Hz and under 3 kHz.

The sorters were tested on all datasets described in section Data acquisition, each having 23–46 neurons. In accordance with our prior results, time offsets of 0, 1 and 2 were applied. The average accuracy for each dataset is displayed in Figure 10.

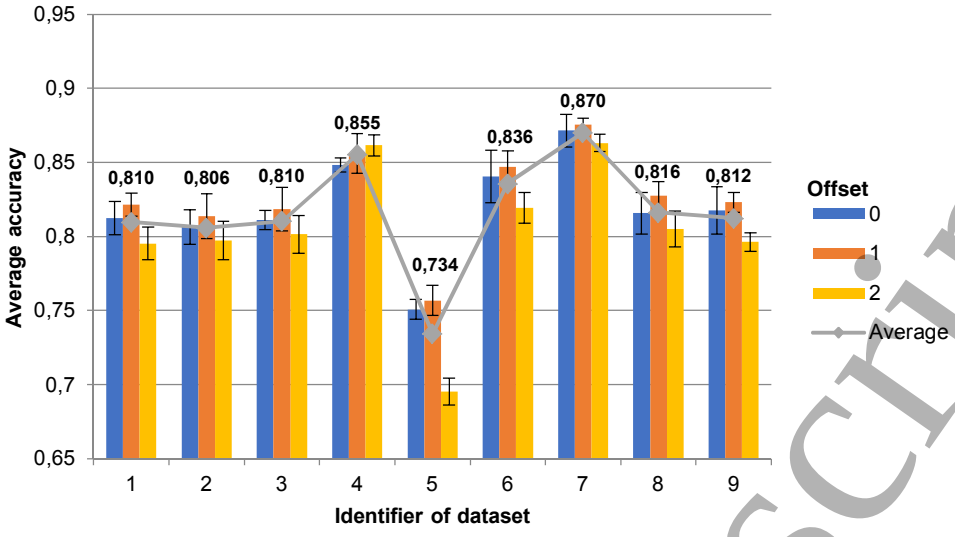

Figure 10. Average accuracy for each offset value with respect to the datasets.

We found that the best results can be achieved with frames having an offset of 1; in this case, the average accuracy is above 0.8 except for dataset 5.

2.2. Sorting with 3-D input

Timeslots consisted of 4–8 frames, each consecutive two being five samples apart, starting at 15–17 frames before the marks. The number of frames in the timeslots is supposed to be small in order to achieve reasonably short training time. Therefore, data had to be decimated in order to maintain the temporal diversity in the samples; we got the best results using every fifth frame.

Certain clusters may contain some characteristic features preceding the spikes, e.g. a fragment from the activity of another neuron might indicate that a spike belonging to a particular cluster will be observed. To exploit this type of information, a few additional frames were taken from the resting phase. We got the best results using two extra samples.

The results for data of different lengths and offsets are depicted in Figure 11.

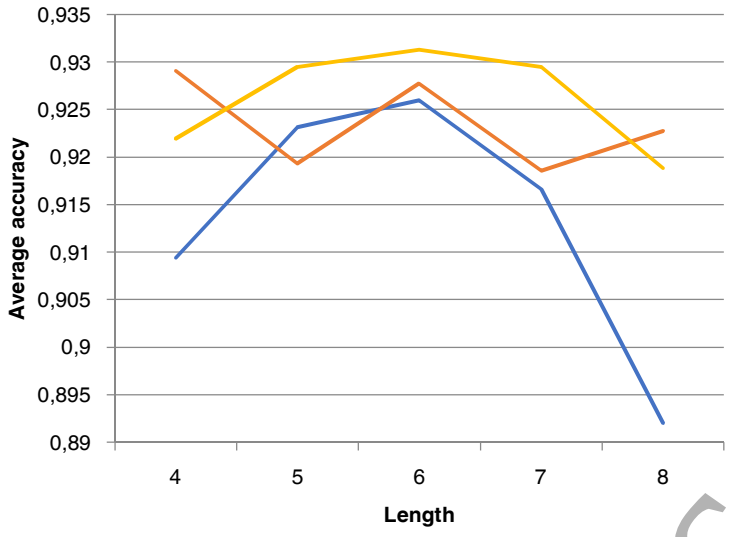

Figure 11. Accuracy for timeslots having different starting points with respect to their length.

We found that the greatest accuracy could be achieved with timeslots having the length of 6 and the starting point of 17; the best result (0.961) was got for a timeslot of this kind. An appropriate confusion matrix is displayed in Table 5.

Real \ Predicted	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22
0	2089	0	0	0	2	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
1	0	906	0	0	0	0	0	0	0	0	0	91	0	0	0	3	0	1	0	0	0	0	0
2	0	1	316	0	0	0	1	2	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0
3	0	0	0	512	0	0	0	1	0	0	0	0	0	0	0	1	0	0	0	18	1	0	0
4	6	1	0	0	1056	0	2	0	0	1	1	0	0	0	1	0	0	2	0	0	0	0	0
5	0	1	0	1	0	579	0	0	0	0	1	0	0	0	0	0	0	0	4	0	0	0	0
6	9	0	0	0	2	0	3261	5	2	0	0	0	0	0	3	3	0	2	0	0	0	0	0
7	17	1	1	0	4	0	8	1645	0	0	0	0	0	1	2	6	0	3	0	0	0	0	1
8	3	0	1	0	0	0	25	7	2311	1	1	5	2	0	1	5	0	4	3	0	0	0	0
9	7	1	0	0	3	0	4	1	0	469	0	0	0	1	0	0	0	2	0	0	0	0	1
10	2	0	0	0	1	0	5	2	0	0	502	1	0	0	0	4	0	1	0	0	0	1	0
11	66	0	0	0	1	0	7	1	2	0	1	1443	0	0	2	3	0	2	0	0	0	0	1
12	2	6	1	0	1	0	13	0	1	0	2	1	1398	0	2	8	0	11	1	0	0	8	0
13	0	4	0	0	0	0	1	0	0	0	1	0	2	343	0	0	0	0	0	1	0	0	1
14	1	0	0	0	1	0	3	1	0	1	0	0	0	0	547	0	1	0	1	0	1	0	0
15	17	0	0	0	6	0	27	15	1	1	1	4	5	2	5	3112	0	15	0	1	1	1	3
16	0	0	0	5	0	0	2	1	0	0	0	0	0	0	0	0	1173	1	0	0	231	0	0
17	8	0	0	0	4	0	8	1	0	0	0	0	0	0	2	2	0	978	1	0	0	0	0
18	6	0	0	0	2	0	8	1	5	0	2	3	1	1	0	7	0	5	525	1	0	0	0
19	0	0	0	27	1	0	0	0	0	0	0	0	1	2	0	0	0	1	2	345	0	0	0
20	0	47	2	1	1	0	0	1	1	0	0	1	8	9	0	1	0	4	0	11	1526	0	0
21	0	0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	2	1	0	0	109	0
22	3	0	0	0	2	0	4	0	0	0	0	2	0	0	0	3	0	1	0	0	0	0	112

Table 5. Confusion matrix displaying the results of the 3-D network (for the details on formatting, see the caption of Table 4.).

Comparing the confusion matrix to the previous one, it can be seen that the majority of misclassifications disappeared (only exceeding the 10 % of the total number of spikes in case

of one neuron). This happened because separate clusters exhibit different temporal characteristics that makes sorting possible despite their presumable spatial similarity.

We repeated the test for all datasets; in this case, timeslots with the starting point of 16 were applied. These timeslots contained the frames the best results in the 2-D case were got for, thus accuracies for frames and timeslots could be compared.

We found that significant improvement can be achieved using the 3-D approach: in this case, the worst average accuracy (0.879) was above the best we got for 2D. We experienced the greatest improvement (0.12) in case of the dataset having the worst results for frames.

Sorters were tested on entirely randomized data as well. In this case, data were shuffled before being divided into training/validation/test databases; the size of the databases did not change. The average accuracy of these sorters for each dataset is compared to the results have been got for data in chronological order in Figure 12.

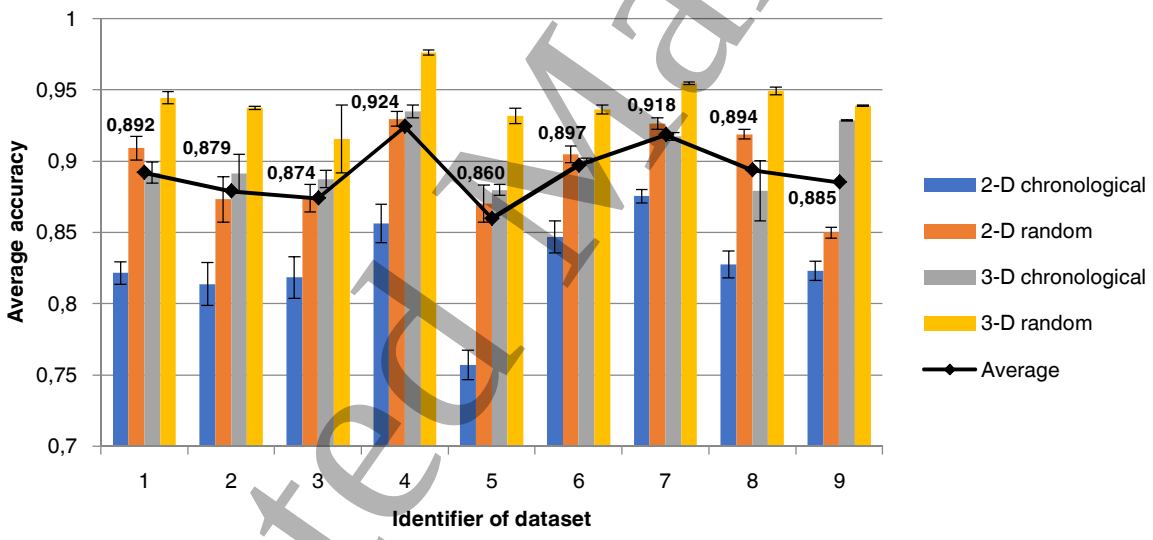

Figure 12. Average accuracy of 2-D and 3-D sorters using data in chronological and random order with respect to the datasets.

We found that sorters trained and tested on randomized data outperformed the ones that used data in chronological order: The 3-D random sorter yielded the best results (with the highest accuracy of 0.977), and the accuracy of 2-D random sorters were close to the 3-D chronological ones. The average accuracy across each dataset was 0.827 for 2-D chronological, 0.895 for 2-D random, 0.901 for 3-D chronological and 0.943 for 3-D random

data, yielding an overall accuracy of 0.891. A confusion matrix of the 3-D random sorter can be seen in Table 6.

Real \ Predicted	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22
0	2036	0	0	0	1	0	1	7	1	2	0	41	0	0	0	0	0	2	0	0	0	0	0
1	0	990	0	0	0	0	0	0	0	0	0	11	0	0	0	0	0	0	0	0	0	0	0
2	0	0	319	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0	0	0	1
3	0	0	0	489	0	1	0	0	0	0	0	0	1	0	0	0	0	0	0	39	3	0	0
4	3	1	0	0	1054	0	2	0	0	0	0	0	1	0	0	2	0	5	0	0	0	1	1
5	0	0	0	0	0	585	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0	0
6	7	0	0	0	3	0	3239	10	3	3	1	3	2	0	0	6	0	6	3	0	0	0	1
7	8	1	1	0	3	1	3	1664	2	1	1	0	1	0	0	2	1	0	0	0	0	0	0
8	1	0	0	0	2	0	6	3	2341	0	0	3	2	0	0	5	0	3	0	0	0	0	3
9	3	1	0	0	1	0	2	0	0	478	1	0	0	0	0	0	0	1	1	0	0	0	1
10	2	1	0	0	1	3	3	1	0	0	506	0	1	0	0	0	0	1	0	0	0	0	0
11	75	76	0	0	2	0	4	2	0	0	0	1363	0	0	0	3	0	1	2	0	0	0	1
12	0	1	0	0	0	1	1	0	2	0	0	0	1437	0	0	1	0	4	6	0	2	0	0
13	0	0	0	0	0	1	0	0	0	0	0	0	4	344	0	0	0	0	1	0	3	0	0
14	1	0	0	0	1	1	0	1	1	0	1	0	3	0	542	1	3	0	0	0	0	0	2
15	6	2	0	0	5	1	12	7	2	0	1	2	0	0	1	3173	0	3	1	0	0	0	1
16	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1410	0	0	0	2	0	0
17	1	2	0	1	4	0	1	6	0	0	0	1	1	0	0	1	0	985	1	0	0	0	0
18	1	1	0	0	0	3	1	1	0	0	0	0	3	0	0	0	0	0	557	0	0	0	0
19	0	0	0	1	1	5	0	0	0	0	0	0	0	0	0	0	0	2	0	370	0	0	0
20	0	0	0	3	0	0	0	1	0	0	1	0	4	3	1	2	11	1	1	4	1581	0	0
21	0	0	0	0	0	0	0	0	0	1	0	0	0	0	0	0	0	1	1	0	0	109	1
22	1	0	0	0	0	0	0	2	0	2	0	0	0	0	0	1	0	0	3	0	0	0	118

Table 6. Confusion matrix displaying the results of the 3-D sorter using random data.

As we can see, this confusion matrix approximates the ideal properly, having values almost only in the main diagonal. The average of misclassifications is 2.64 % (this value is exceeded in case of only five clusters).

3. Prediction

The distant goal of this project is to establish a stable algorithmic background for BCIs, and processing time proves to be a bottleneck for these systems. Given that detection, transmission and actuation times are approximately constant, we raised the question whether control signal could be generated faster by predicting the identity of each single unit activity right before the particular neuron fires. Theoretically, this could be achieved using the LFP originating from the surrounding structures that communicate with the identified cells and the local potential changes in the plasma membrane of the proximal neurons.

The architecture of the network applied was identical with the base structure of the 3-D classifier, exhibiting one convolutional layer, followed by a fully connected and an output layer. The number and size of the filters remained unchanged, as well. As activation function of the

first two modules, rectified linear unit (ReLU) was applied while the output was formed using softmax nonlinearity.

The data used were timeslots of 5 frames corresponding to a 1.05 ms long time window (preserving the decimation by 5), sweeping through several different starting points ranging from 10 (in this case, the central frame of the timeslot is the marked one) to 29. The characteristic image of these timeslots can be seen in Figure 13.

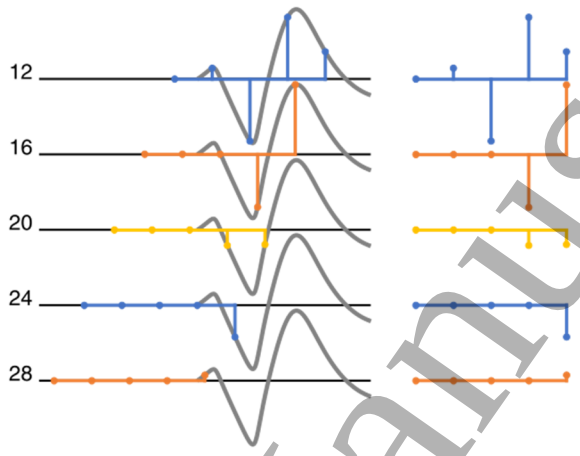

Figure 13. The characteristic shape of the timeslots used for prediction with respect to their starting point. On the left, the timeslots are superimposed on a generic spike waveform for better visibility; on the right, only the frames taken can be seen.

Measurement results are depicted in Figure 14.

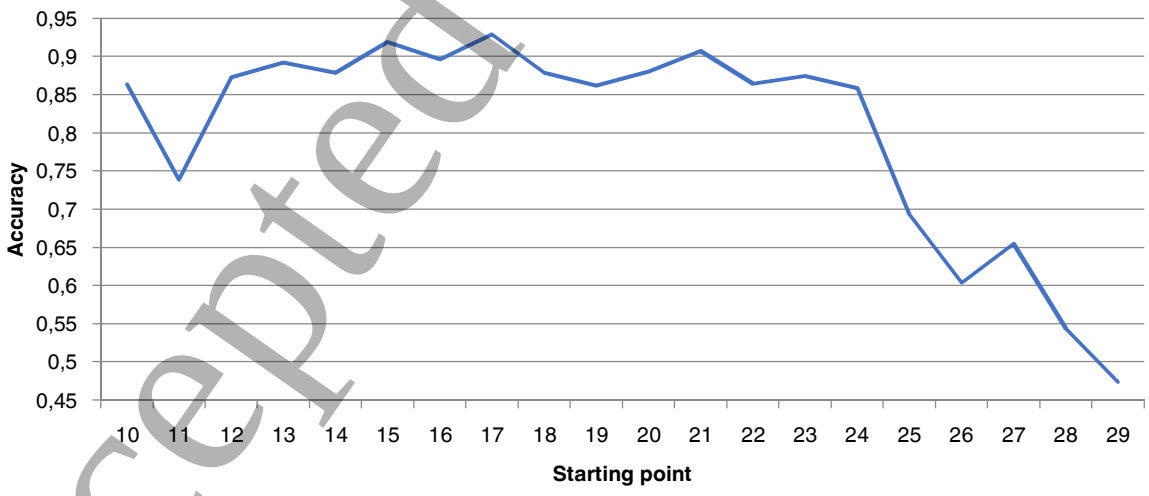

Figure 14. The accuracy of the network with respect to the starting point of the timeslots shifted backwards.

Our results show that as long as the spike peak is within the timeslot, the system performs with a constant accuracy (of about 80-90 %), independently from the location of the mark inside the timeslot. Note that for starting points greater than 21-22, the peak of the action potential is not included in the timeslot. As the shift approaches 24, the performance begins to drop rapidly. That indicates that the initial waveform (i.e. the 'falling edge') of the spike exhibits key information about the identity of the corresponding single unit, enough to let the system accurately label action potentials even in the absence of spike peaks. Having the starting point of the timeslots at 28, the accuracy falls to 50 %.

Since the number of single units being classified is above 20 and the classification is based solely on the potential changes in the surrounding tissue (i.e. the action potential is entirely excluded), this result can be considered an early validation of the predicting capabilities of the network. The system proves to be worthy of further development, carrying the potential to predicting upcoming spikes in the range of 0.1–0.2 ms.

Discussion

Although Brain-Computer Interface development has seen its renaissance in the last decade, current technology is far from suitable for everyday use. From data acquisition through control signal formulation to the design and coordination of the actuators there are several issues yet to be tackled.

We have successfully shown that CNNs combined with RNNs hold the potential for identifying the intervals corresponding to actual activities. Our capabilities for evaluating the results were limited by the quality of ground data, thus our research was not conclusive in regards to the exact accuracy of the detector. However, considering the positive prediction value an average of 19.07%, while in case of the recall an average of 69.63% could be reached given these datasets. We assume that a relatively high amount of false positive events cause little disturbance in a BCI system if it is equipped with a robust spike sorter.

For the task of data classification, two architectures were applied, using 2-D and 3-D input. We examined these systems tuning several different parameters with respect to data (e.g. offset, length) and the sorters themselves (e.g. activation function, use of pooling). The

experiments conducted granted us very promising results: the average accuracy of the system resided between 81.8 and 97.7 % (with an overall average of 89.1 %), even considering datasets for which the number of single units exceeded 40. In general, better performance was achieved using 3-D networks (grasping spatiotemporal variations of the input) but at the cost of definitely longer training time (in contrast with the 2-D sorter exploiting only spatial variations in data). Another aspect was the usage of 'chronological' and 'random' data: in the former case, data blocks used for training, validation and test were in chronological order, i.e. sequential data were split into three disjoint blocks, then the samples were shuffled block-wise; in the latter case, data were randomized first then split into parts. In our experience, sorters trained on random data proved to be more accurate. A possible cause of this behavior is the migration of the neurons: the cells are capable of movement and the insertion of a relatively large probe, accompanied by death, injury, or simple displacement involving multiple cells, can reinforce this effect. The result of the process is that the activities belonging to the same cluster might show up on different sets of channels at the start and at the end of the recording. This effect could be diminished in the cases discussed by using shuffled samples, or increasing the size of the training database. In practice, nevertheless, solutions that are more robust are needed to track these migrating clusters, enabling the network to extend its training phase beyond actual functioning.

The last idea we intended to pursue was whether our classifier could be utilized for predicting the identity of upcoming action potentials. We shifted timeslot sampling backwards to see how it affects the classification accuracy. The performance of the system was very sensitive to the presence of the initial waveform of the action potential in the timeslot (regardless of whether the frame corresponding to the activity peak was actually included). The network was able to predict the identity of the spike based solely on the initial waveform (about 0.1 ms before the actual peak) but the performance dropped drastically as it was getting shifted out of the timeslot. In spite of this, the accuracy remained moderately high considering the vast number of single units and that only the LFP and activities of nearby neurons could be used. These findings indicate the possibility of successful spike prediction using a more complex system in the future.

Conclusion

In this study we proposed an alternative to current spike detection and spike sorting algorithms exploiting the potential convolutional and recurrent neural networks have for these purposes.

Although these algorithms could provide a good basis for creating functional, real-time BCIs, there still are several obstacles to overcome. One of the main issues we had to face was the lack of adequately labeled data. We aim to create an appropriate ground truth dataset in order to tune these algorithms further.

In spite of these difficulties, our detector and sorter proved usable, yielding a satisfactory performance in each case examined. We assume that from an ensemble algorithm composed from the parts implemented, even better performance could be expected.

Acknowledgements

The research leading to these results has received funding from the Hungarian Brain Research Program Grant (Grant No. 2017-1.2.1-NKP-2017-00002). The research within project No. VEKOP-2.3.2-16-2017-00013 by István Ulbert was supported by the European Union and the State of Hungary, co-financed by the European Regional Development Fund. Gergely Márton and Richárd Fiáth are thankful for the OTKA PD121015 (G. M.) and PD124175 (R. F.) grants supported by the National Research, Development and Innovation Office.

References

[1] Schwartz A B, Cui X T, Weber D J J and Moran D W 2006 Brain-Controlled Interfaces: Movement Restoration with Neural Prosthetics *Neuron* **52** 205–20

[2] Hochberg L R, Bacher D, Jarosiewicz B, Masse N Y, Simeral J D, Vogel J, Haddadin S, Liu J, Cash S S, Van Der Smagt P and Donoghue J P 2012 Reach and grasp by people with tetraplegia using a neurally controlled robotic arm *Nature* **485** 372–5

[3] Wodlinger B, Downey J E, Tyler-Kabara E C, Schwartz A B, Boninger M L and

Collinger J L 2015 Ten-dimensional anthropomorphic arm control in a human brain-machine interface: Difficulties, solutions, and limitations *J. Neural Eng.* **12**

[4] Márton G, Orbán G, Kiss M, Fiáth R, Pongrácz A and Ulbert I 2015 A multimodal, SU-8 - Platinum - Polyimide microelectrode array for chronic in vivo neurophysiology *PLoS One* **10**

[5] Xu H, Hirschberg A W, Scholten K, Berger T W, Song D and Meng E 2018 Acute in vivo testing of a conformal polymer microelectrode array for multi-region hippocampal recordings *J. Neural Eng.* **15**

[6] Waldert S, Pistohl T, Braun C, Ball T, Aertsen A and Mehring C 2009 A review on directional information in neural signals for brain-machine interfaces *J. Physiol. Paris* **103** 244–54

[7] Schwartz A B, Kettner R E and Georgopoulos A P 1988 Primate motor cortex and free arm movements to visual targets in three-dimensional space. I. Relations between single cell discharge and direction of movement. *J. Neurosci.* **8** 2913–27

[8] Georgopoulos A P, Caminiti R, Kalaska J F and Massey J T 1983 *Spatial Coding of Movement: A Hypothesis Concerning the Coding of Movement Direction by Motor Cortical Populations* vol 49

[9] Georgopoulos A P, Schwartz A B and Kettner R E 1986 Neuronal population coding of movement direction *Science (80-.).* **233** 1416–9

[10] Rey H G, Pedreira C and Quian Quiroga R 2015 Past, present and future of spike sorting techniques *Brain Res. Bull.* **119** 106–17

[11] Lewicki M S 1998 A review of methods for spike sorting: The detection and classification of neural action potentials *Netw. Comput. Neural Syst.* **9**

[12] Vargas-Irwin C and Donoghue J P 2007 Automated spike sorting using density grid contour clustering and subtractive waveform decomposition *J. Neurosci. Methods* **164** 1–18

[13] Fee M S, Mitra P P and Kleinfeld D 1996 Automatic sorting of multiple unit neuronal

signals in the presence of anisotropic and non-Gaussian variability *J. Neurosci. Methods* **69** 175–88

[14] Chah E, Hok V, Della-Chiesa A, Miller J J H, O'Mara S M and Reilly R B 2011 Automated spike sorting algorithm based on Laplacian eigenmaps and k-means clustering *J. Neural Eng.* **8**

[15] Kim K H and Kim S J 2000 Neural spike sorting under nearly 0-dB signal-to-noise ratio using nonlinear energy operator and artificial neural-network classifier *IEEE Trans. Biomed. Eng.* **47** 1406–11

[16] Choi J H, Jung H K and Kim T 2006 A new action potential detector using the MTEO and its effects on spike sorting systems at low signal-to-noise ratios *IEEE Trans. Biomed. Eng.* **53** 738–46

[17] Quiroga R Q, Nadasdy Z and Ben-Shaul Y 2004 Unsupervised spike detection and sorting with wavelets and superparamagnetic clustering *Neural Comput.* **16** 1661–87

[18] Abeles M and Goldstein M H 1977 Multispike Train Analysis *Proc. IEEE* **65** 762–73

[19] Wood E, Fellows M, Donoghue J R and Black M J 2005 Automatic spike sorting for neural decoding pp 4009–12

[20] Wang G L, Zhou Y, Chen A H, Zhang P M and Liang P J 2006 A robust method for spike sorting with automatic overlap decomposition *IEEE Trans. Biomed. Eng.* **53** 1195–8

[21] Biffi E, Ghezzi D, Pedrocchi A and Ferrigno G 2008 Spike detection algorithm improvement, spike waveforms projections with PCA and hierarchical classification pp 122–122

[22] Chen Y Y, Tsai Y M and Chen L G 2011 Algorithm and implementation of multi-channel spike sorting using GPU in a home-care surveillance system *Proceedings - IEEE International Conference on Multimedia and Expo*

[23] Oweiss K G and Anderson D J 2002 Spike sorting: A novel shift and amplitude invariant technique *Neurocomputing* **44–46** 1133–9

[24] Hulata E, Segev R and Ben-Jacob E 2002 A method for spike sorting and detection based on wavelet packets and Shannon's mutual information *J. Neurosci. Methods* **117** 1–12

[25] Takahashi S, Anzai Y and Sakurai Y 2003 A new approach to spike sorting for multi-neuronal activities recorded with a tetrode - How ICA can be practical *Neurosci. Res.* **46** 265–72

[26] Vogelstein R J, Murari K, Thakur P H, Diehl C, Chakrabartty S and Cauwenberghs G 2005 Spike sorting with support vector machines pp 546–9

[27] Yang K, Wu H and Zeng Y 2017 A Simple Deep Learning Method for Neuronal Spike Sorting *Journal of Physics: Conference Series* vol 910

[28] Stevenson I H and Kording K P 2011 How advances in neural recording affect data analysis *Nature Neuroscience* vol 14 pp 139–42

[29] Chaure F J, Rey H G and Quian Quiroga R 2018 A novel and fully automatic spike-sorting implementation with variable number of features *J. Neurophysiol.* **120** 1859–71

[30] Li J, Chen X and Li Z 2019 Spike detection and spike sorting with a hidden Markov model improves offline decoding of motor cortical recordings *J. Neural Eng.* **16**

[31] Arora A, Lin J J, Gasperian A, Maldjian J, Stein J, Kahana M and Lega B 2018 Comparison of logistic regression, support vector machines, and deep learning classifiers for predicting memory encoding success using human intracranial EEG recordings *J. Neural Eng.* **15**

[32] Schirrmeister R T, Springenberg J T, Fiederer L D J, Glasstetter M, Eggensperger K, Tangermann M, Hutter F, Burgard W and Ball T 2017 Deep learning with convolutional neural networks for EEG decoding and visualization *Hum. Brain Mapp.* **38** 5391–420

[33] Craik A, He Y and Contreras-Vidal J L 2019 Deep learning for electroencephalogram (EEG) classification tasks: A review *J. Neural Eng.* **16** 31001

[34] Houston B, Thompson M, Ko A and Chizeck H 2019 A machine-learning approach to volitional control of a closed-loop deep brain stimulation system *J. Neural Eng.* **16**

[35] Saif-ur-Rehman M, Lienkämper R, Parpaley Y, Wellmer J, Liu C, Lee B, Kellis S, Andersen R A, Iossifidis I, Glasmachers T and Klaes C 2019 SpikeDeeptector: A deep-learning based method for detection of neural spiking activity *J. Neural Eng.*

[36] Fiáth R, Márton A L, Mátyás F, Pinke D, Márton G, Tóth K and Ulbert I 2019 Slow insertion of silicon probes improves the quality of acute neuronal recordings *Sci. Rep.* **9** 111

[37] Fiáth R, Raducanu B C, Musa S, Andrei A, Lopez C M, van Hoof C, Ruther P, Aarts A, Horváth D and Ulbert I 2018 A silicon-based neural probe with densely-packed low-impedance titanium nitride microelectrodes for ultrahigh-resolution in vivo recordings *Biosens. Bioelectron.* **106** 86–92

[38] Pachitariu M, Steinmetz N, Kadir S, Carandini M and Harris K 2016 Fast and accurate spike sorting of high-channel count probes with KiloSort *Advances in Neural Information Processing Systems* vol 2016 pp 4448–56

[39] Fukushima K 1980 Neocognitron: A self-organizing neural network model for a mechanism of pattern recognition unaffected by shift in position *Biol. Cybern.* **36** 193–202

[40] Hochreiter S and Schmidhuber J 1997 Long Short-Term Memory *Neural Comput.* **9** 1735–80

[41] Werbos P J 1988 Generalization of backpropagation with application to a recurrent gas market model *Neural Networks* **1** 339–56