Brain Informatics

RESEARCH

Open Access

SANTIA: a Matlab-based open-source toolbox for artifact detection and removal from extracellular neuronal signals

Marcos Fabietti[1], Mufti Mahmud[1,2,3*], Ahmad Lotfi[1], M. Shamim Kaiser [4], Alberto Averna[5], David J. Guggenmos[6], Randolph J. Nudo[6], Michela Chiappalone[7] and Jianhui Chen[8,9]

Abstract

Neuronal signals generally represent activation of the neuronal networks and give insights into brain functionalities. They are considered as fingerprints of actions and their processing across different structures of the brain. These recordings generate a large volume of data that are susceptible to noise and artifacts. Therefore, the review of these data to ensure high quality by automatically detecting and removing the artifacts is imperative. Toward this aim, this work proposes a custom-developed automatic artifact removal toolbox named, SANTIA (SigMate Advanced: a Novel Tool for Identification of Artifacts in Neuronal Signals). Developed in Matlab, SANTIA is an open-source toolbox that applies neural network-based machine learning techniques to label and train models to detect artifacts from the invasive neuronal signals known as local field potentials.

Keywords: Local field potential, Artifacts, Neural networks, Machine learning, Neuronal signals

1 Introduction

Neural recordings give insight into the brain's structures and functions. The recording systems aim to capture the electrical activity of the biological structures; however, these are not isolated systems and activities from other sources are also recorded. Besides, faulty equipment handling, electrical stimulation, or movements of electrodes can cause distortions in the recordings. As part of the recording process, the recordings must be reviewed to identify corrupted segments and address them, as they are detrimental for any posterior analysis. This includes artifact removal (e.g., filtering, template subtraction, or advanced computational techniques) or discarding the segment.

Each neural recording session produces a huge volume of data, especially if it is obtained over a long period of time and the experiment requires repetition. The amount of data gets multiplied by the number of recording sites. The post-experimental reviewing process consisting of annotating long recordings for evoked responses or unusual activities, which may happen in a much smaller time scale (e.g., 0.1 s in an hour), is a tedious and tiresome task. By automating this task, the researcher can focus on the interpretation task for diagnosis or an application. Employing machine learning (ML) algorithms, which have the ability to learn from patterns to predict unseen data, has been successful in the literature. However, a computational background is required to apply them successfully as there are intricacies such as defining hyper-parameters.

Research groups in the neuroscience community have developed and shared toolboxes for analyzing neural recordings [1–3]. Given the wide arrange of neuronal signals, data formats, analysis techniques, and purposes, each one has advocated their efforts into specific elements. Table 1 lists the available open toolboxes and their functions in regard to aiding noise detection and removal

*Correspondence: muftimahmud@gmail.com; mufti.mahmud@ntu.ac.uk
[1] Department of Computer Science, Nottingham Trent University, Clifton Lane, Nottingham NG11 8NS, UK
Full list of author information is available at the end of the article

Table 1 Open-source toolboxes and noise detection and removal functionalities

	Artifact detection	Digital filtering	Data visual.	Spectral analysis	Stim. art. removal	File oper.	Multiple formats
Brainstorm [5]	✓	✓	✓	✓	X	X	✓
BSMART [6]	X	X	✓	✓	X	X	✓
Chronux [7]	X	X	✓	✓	X	X	X
Elephant [8]	X	X	✓	✓	X	X	✓
Fieldtrip [9]	X	✓	✓	✓	X	X	✓
Klusters, NeuroScope, NDManager [10]	X	✓	✓	✓	X	✓	✓
Neo [11]	X	X	✓	X	X	✓	✓
NeuroChaT [12]	X	X	✓	✓	X	X	✓
Spycode [13]	X	✓	✓	✓	X	X	✓
SANTIA	✓	✓	✓	✓	✓	✓	✓

Data visualization, stimulation artifact removal and file operations (i.e., file splitting, concatenation, column rearranging)

in local field potential signals (LFP). An in-depth analysis of these toolboxes is reported in [4]. Hence, the description below will be dedicated to elaborate on the reported toolboxes.

Brainstorm [5] is an open-source application dedicated to neuronal data visualization and processing, with an emphasis on cortical source estimation techniques and their integration with anatomical magnetic resonance imaging data. It offers an intuitive interface, powerful visualization tools, and the structure of its database allows the user to work at a higher level. BSMART [6] is a toolbox intended for spectral analysis of continuous neural time series data recorded simultaneously from multiple sensors. It is composed mainly of tools for autoregressive model estimation, spectral quantity analysis, and network analysis. All functionality has been integrated into a graphical user interface (GUI) environment designed for easy accessibility.

Chronux [7] is an open-source Matlab software project for the analysis of neural signals via signal specialized modules for spectral analysis, spike sorting, local regression, audio segmentation, and other tasks. Similarly, Elephant [8] is a Python library for the analysis of electrophysiological data, such as LFP or intracellular voltages. It offers a broad range of functions for analyzing multiscale data of brain dynamics from experiments and brain simulations, such as signal-based analysis, spike-based analysis, and methods combining both signal types.

FieldTrip [9] is an open-source software package developed for the analysis of electrophysiological data. It supports reading data from a large number of different file formats and includes algorithms for data preprocessing, event-related field/response analysis, parametric and non-parametric spectral analysis, forward and inverse source modeling, connectivity analysis, classification, real-time data processing, and statistical inference. Klusters, NeuroScope, and NDManager [10] are a free software suite for neurophysiological data processing and visualization. NeuroScope is an advanced viewer for electrophysiological and behavioral data with limited editing capabilities, Klusters a graphical cluster cutting application for manual and semi-automatic spike sorting, and NDManager an experimental parameter and data processing manager.

Neo [11] is a tool whose purpose is to handle electrophysiological data in multiple formats. Due to its unique property of being able to read or write the data from or to a variety of commonly used file formats, it is included in the list. NeuroChaT [12] is an Python open-source toolbox created to standardized open-source analysis tools available for the analysis of neuronal signals recorded in vivo in the freely behaving animals.

Spycode [13] is a smart tool for multi-channel data processing which possesses a vast compendium of algorithms for extracting information both at a single channel in addition to at the whole network level, and the capability of autonomously repeating the same set of computational operations to multiple recording streams, all without manual intervention.

Out of the aforementioned toolboxes, the only one that allows for artifact detection is Brainstorm. It allows for manual inspection and automatic detection of artifacts, mainly of muscular and movement origin, by filtering the signals in frequency bands (ocular 1.5–15 Hz; for ECG: 10–40 Hz; for muscle noise and some sensor artifacts: 40–240 Hz and subject movement, eye movements, and dental work 1–7 Hz) and classifying the absolute value of signal with a standard deviation threshold. However, artifacts can span a large bandwidth and studies show that they can overlap with those of the neural signals [14]. As

Table 2 Advancements of SANTIA over SigMate

Toolbox	SAD	UNoC	SE	Up	DF	DV	SA	SAR	FO	MF
SigMate [15]	X	X	X	X	✓	✓	✓	✓	✓	✓
SANTIA	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓

SAD state-of-the-art artifact detection, *UNoC* unlimited number of channels, *SE* supported environment, *Up* updates, *DF* digital filtering, *DV* data visualization, *SA* spectral analysis, *SAR* stimulation artifact removal, *FO* file operations, *MF* multiple formats

an example, the alpha band (8–12 Hz) can have oscillations of high amplitude and be falsely detected as an artifact.

There is one other toolbox that deals with LFP artifact detection. This is SigMate [15–17], a Matlab-based tool that incorporates standard methods to analyze spikes and electroencephalography (EEG) signals, and in-house solutions for LFP analysis. The functionality provided by SigMate include: artifact removal, both fast [18] and slow [14], angular tuning detection [19], noise characterization [20], cortical layer activation order detection, and network decoding [21–24], sorting of single trial LFP [25–28], etc. It deals with slow stimulus artifact removal through an algorithm that subtracts an estimation of the signal by averaging the peaks and valleys detected in it, eliminating the offset. In addition, it allows for visualization of the spectrogram using short-time Fourier transform of the recording to allocate artifactual frequency bands and allow their filtering, among many other analysis functionalities.

To offer a more competitive toolbox, it has been expanded with new functionalities, reported in Table 2. These include state-of-the-art modules for artifact detection, or the analysis of any number of channels unlike SigMate which is limited to 5. Thus, in this paper, we present the SANTIA toolbox (SigMate Advanced: a Novel Tool for Identification of Artifacts in Neuronal Signals), a friendly user interface that aids the offline identification of artifacts process by simplifying the steps to train powerful computational algorithms with the minimum input of the user. For a wider adoption by the community, the toolbox is freely available online at https://github.com/IgnacioFabietti/SANTIAtoolbox.

The recording of neuronal data, especially when using multi-electrode arrays, can lead to electronic files of notable size. Figure 1 illustrates a conducted survey of the formats of invasive neural recordings in open datasets [29]. The data show that '.mat' is the preferred extension for storage by a substantial margin. This emphasizes the necessity to develop tools which address the datasets available in '.mat' format. Therefore, SANTIA was implemented in Matlab and works with single files containing multi-channel data files in a variety of formats. The toolbox only depends on the Deep Learning Toolbox and the basic version of Matlab 2020a and above, therefore can function in any operating system. SANTIA has been developed with the latest app development environment of Matlab, which allows it to be supported for longer and be improved with new modules, such as GUI improvements which are planned for the next update.

The remainder of the paper is composed of 5 sections: Sect. 2 describes the local field potentials; Sect. 3 describes the methods followed by the testing results presented in Sect. 4. Finally, in Sect. 5, discussion and conclusion are presented.

2 Local field potentials

Local field potentials are invasive neuronal recordings, which are equal to the sum of the activity of a neuronal population, that has been low-pass-filtered under 300 Hz, and whose amplitude ranges from a few micro-volts to hundreds of micro-volts or more depending on the studied structure [30]. They can be recorded by single or multi-channel micro-electrodes (glass micro-pipettes, metal, or silicon electrodes), during in vitro or in vivo

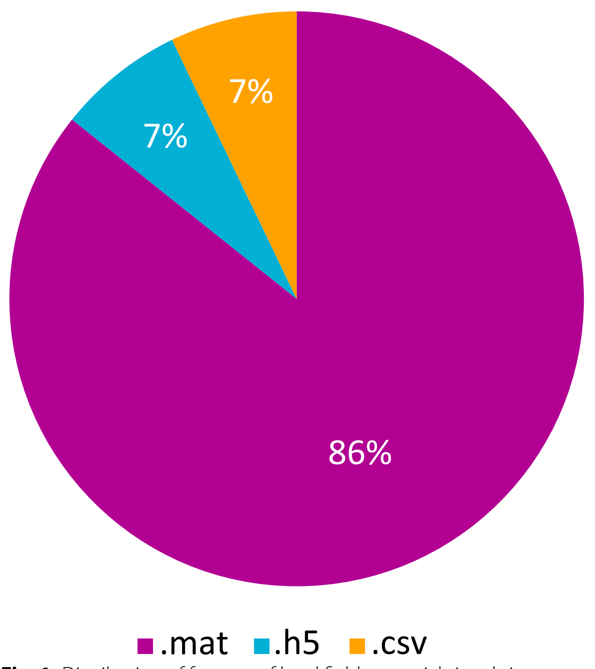


Fig. 1 Distribution of formats of local field potential signals in open datasets, extracted from [29]

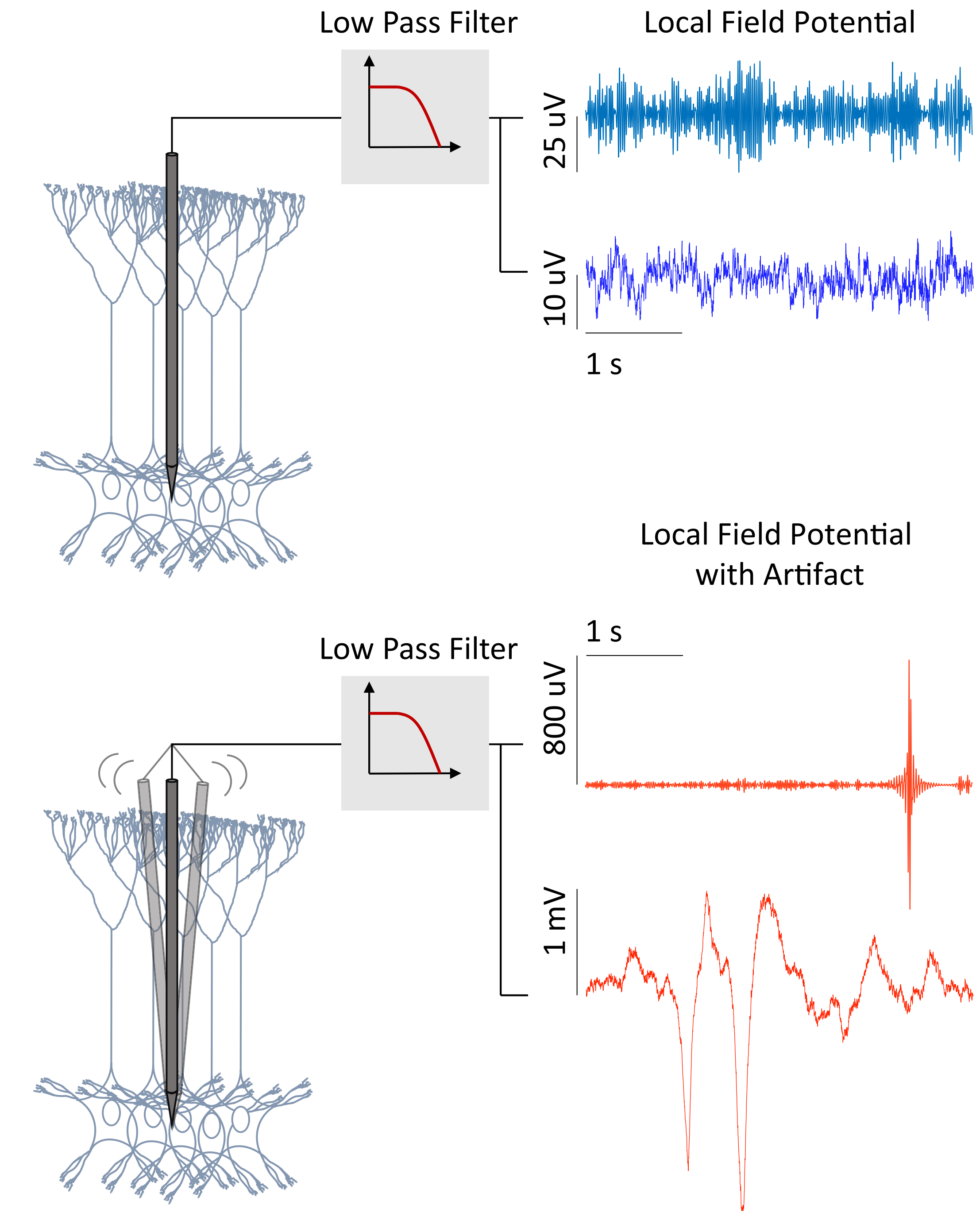


Fig. 2 Recording of extracellular neuronal signals from behaving rodents using linear implantable neural probe (shown in gray). Representative local field potential signals with and without movement artifacts are shown from two datasets. The blue traces denote signals without artifacts and the red traces show examples of movement artifacts present in the signals

experiments to gain insight into the behavior of brain structures, and diagnosis, and are used in application such as brain–machine interfaces. Figure 2 illustrates the concept.

As with all neuronal signals, their recording process can be influenced by internal and external factors, causing artifacts. Within an organism, electric potentials are also generated mainly from ocular, muscle, or heart activity, i.e., electrooculogram, electromyogram, and electrocardiogram, respectively. Examples of external sources include transmission lines, cellphone signals, and faulty experimental setup. Local field potentials in particular can be affected by spike bleed-through [31], light stimulation [32], respiration-coupled oscillations [33], and deep brain stimulation artifacts [34].

The consequences of the presence of artifacts can be detrimental, such as misdiagnosis, disturbance of the study of the brain activity, or causing a brain–machine interface device to be mistakenly operated. Looking at the case of another neuronal signal, EEG signals, the presence of abnormalities raised the median review time from 8.3 to 20.7 min [35]. To make use of these recording successfully, these artifacts must be first identified and then dealt with. The use of computational techniques which are able to learn from complex data patterns has yielded promising results in the field. In the next section, they will be described.

3 Methods

3.1 Artifact detection

While there are many contributions on artifact detection in neuronal signals, specially non-invasive ones like EEG, the same cannot be said about LFP. For the latter, the main approach has been the application of ML algorithms in the form of artificial neural networks.

Artificial intelligence has been used for analysis of patterns and classification in diverse fields such as, anomaly detection [29, 36–44], biological data mining [45, 46], disease detection [47–58], monitoring of human [59–62], financial forecasting [63], image analysis [64, 65], and natural language processing [66–68]. Most of the time, these algorithms are composed of multiple layers of neurons for processing of non-linear information and were inspired by how the human brain works. Each neuron calculates an inner product of its inputs (x_i) and their respective weights (w_i), and then, the bias (b) is added and, finally, the non-linear activation function is applied, which in most cases is a sigmoid function, *tan* hyperbolic, or rectified linear unit. Thus, the output of a neuron (z_i) can be expressed as detailed in Eq. 1

$$z_i = f\left(\sum_{i=1}^{n} x_i w_i + b\right). \tag{1}$$

To propagate the information and train the network, the output of a layer is fed as input to the subsequent unit in the next layer. The result of the final output layer is used as the solution for the problem.

There are many variations of the neural network architecture based on their principles in determining their rules. For example, authors in [69] trained a multi-layered perceptron (MLP) to identify slow-waves in LFP. An MLP is composed of three sections: an input layer, a hidden layer, and an output layer, where the units of the latter two use the non-linear activation defined in Eq. 1. The modeling complex of non-linear relations improves when it contains multiple numbers of hidden layers, compared to a shallow architecture [70].

In our earlier publications [37], an MLP is employed to identify artifacts in LFP along with two other architectures: long short-term memory (LSTM) networks and one dimensional convolutional neural network (1D-CNN) [71, 72]. The diagrams of the main components of these architectures are depicted in Fig. 3. The LSTM architecture is a type of recurrent network spanning adjacent time steps in a manner that at every point the neurons take the current data input as well as the values of the hidden neurons that collect the information of the previous time steps. On the other hand, convolutional networks are a specific form of neural network that is well suited to computer vision applications due to their capacity to hierarchically abstract representations of spatial operations. A variation of it, designed for problems where the input is a time sequence, is named 1D-CNN.

A comparison of the results obtained can be seen in Table 3. Unlike other machine learning techniques where expertise is required to extract significant features from the signals and which may cause bias in itself, these results indicate that neural networks have the capacity to do it automatically. In addition, it is done in a computationally efficient way: 1-min LFP sampled at 1017 Hz analyzed in 2.27 s equal 26,881 data points analyzed per second. As a negative, the training of the neural network is the step where most time and computational power are consumed.

Having described the classification algorithm that will be used in the toolbox, we proceed to detail its use in the next section.

3.2 Operation

The toolbox can be directly downloaded from the Github repository (https://github.com/IgnacioFabietti/SANTIAtoolbox). Once the toolbox is launched, the GUI provides easy access to all modules. It is important to highlight that SANTIA is a generic environment structured around one single interface in which specific functions were implemented, not a library of functions on top of which a GUI has been added to simplify access.

It is structured in three main modules, designed to perform various processing and analysis on the neuronal signal files. The main functionalities of the first one include: data loading, scaling, reshaping, channel selection, labeling, saving, and 2D display. The second module is composed of: data loading and splitting, hyper-parameter

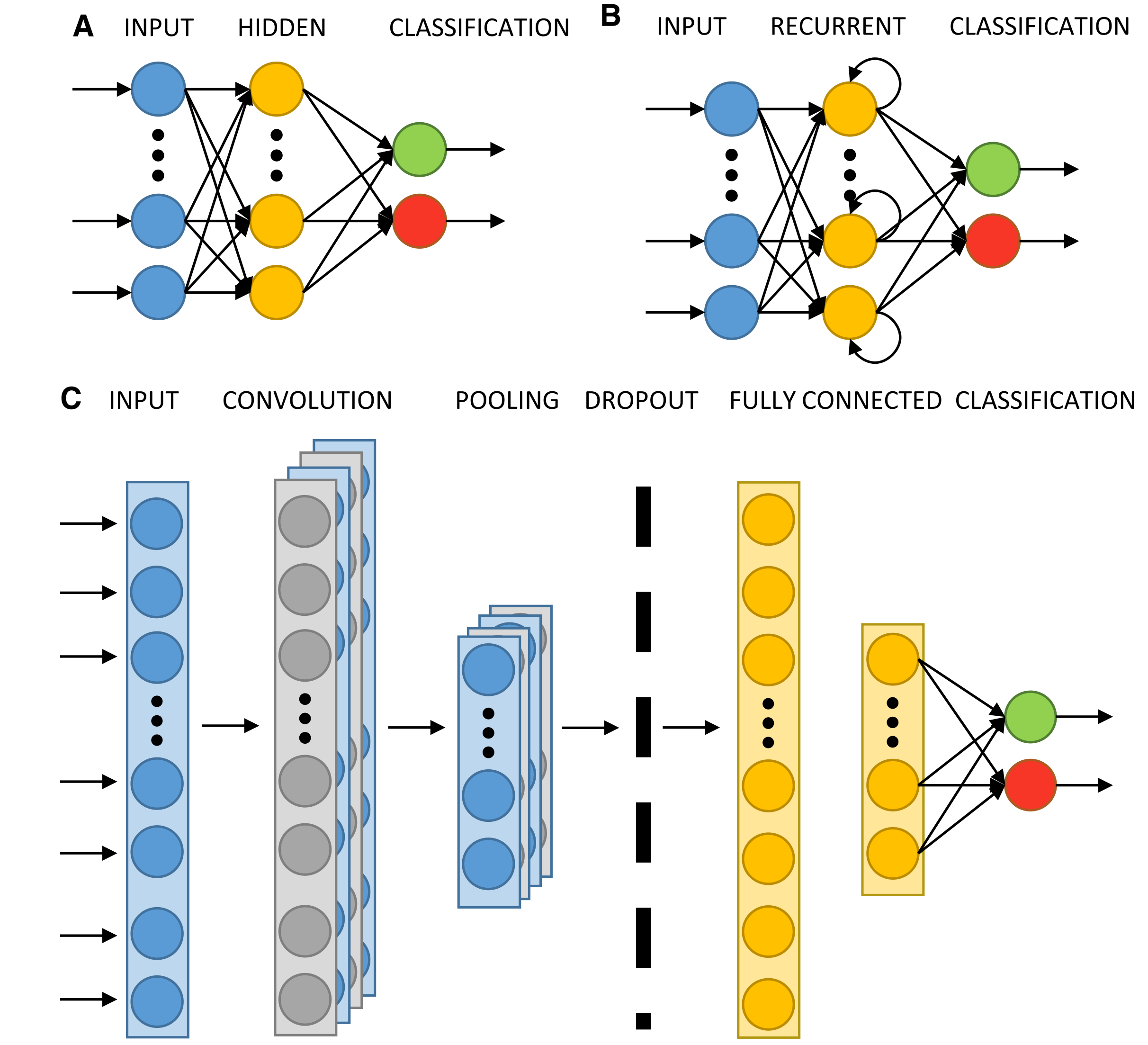


Fig. 3 Architectures of different neural network models: multi-layer perceptron (**A**), long short-term memory (**B**), and one-dimension convolutional neural network (**C**). Each circle represents a neuron, multiple rectangles a layer's depth, and the arrows how the information is propagated throughout each network

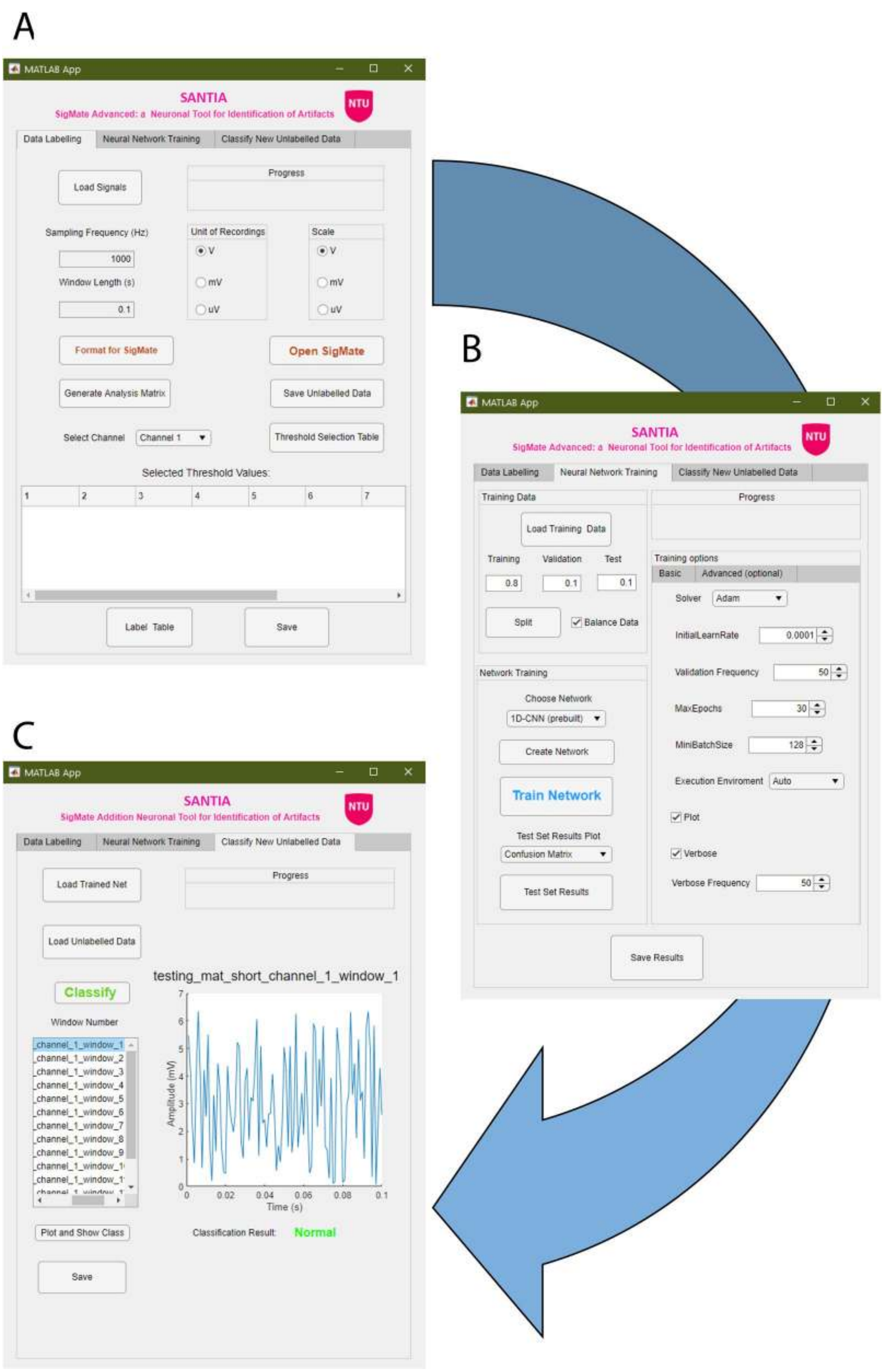
Table 3 Performance comparison, extracted from [72]

Network	Accuracy	Parameters	Computational time (s)
1D-CNN [72]	95.1	561218	2.27 ± 0.13
MLP [37]	93.2	1532	2.57 ± 0.06
LSTM [71]	87.1	4418	3.47 ± 0.04

setting, network load or design, network train, test set classification, and threshold setting and saving. Finally, the third one comprehends: data and network loading, classification, and 2D data display and saving.

The GUI allows for user interaction via the selection of functions, parameters, and keyboard inputs, which are processed in the back end. A verification routine executes before running any function to ensure the user has not skipped a step or has not completed the necessary inputs or parameter selection. This minimizes the possible human errors and time expenditure. In case of doubt of the purpose of an element of the GUI, tool tips appear when hovering the cursor over it with a brief explanation.

The functions to display warning messages, generate figures, and compute the labeling, training, or classification are allocated in the back end. These developed features were tested with a dataset recorded from a 4-shank,



Sampling Frequency (Hz) 1000
Window Length (s) 0.1
Unit of Recordings: [✓ V] mV uV
Scale: [✓ V] mV uV
Select Channel: Channel 1

Training 0.8 Validation 0.1 Test 0.1
✓ Balance Data
Solver: Adam
InitialLearnRate 0.0001
Validation Frequency 50
MaxEpochs 30
MiniBatchSize 128
Execution Enviroment: Auto
✓ Plot
✓ Verbose
Verbose Frequency 50
Choose Network: 1D-CNN (prebuilt)
Test Set Results Plot: Confusion Matrix

Classification Result: Normal

Fig. 4 Screenshots of the SANTIA toolbox graphical user interface: Data Labeling (**A**), Neural Network Training (**B**), and Classify New Unlabeled Data (**C**)

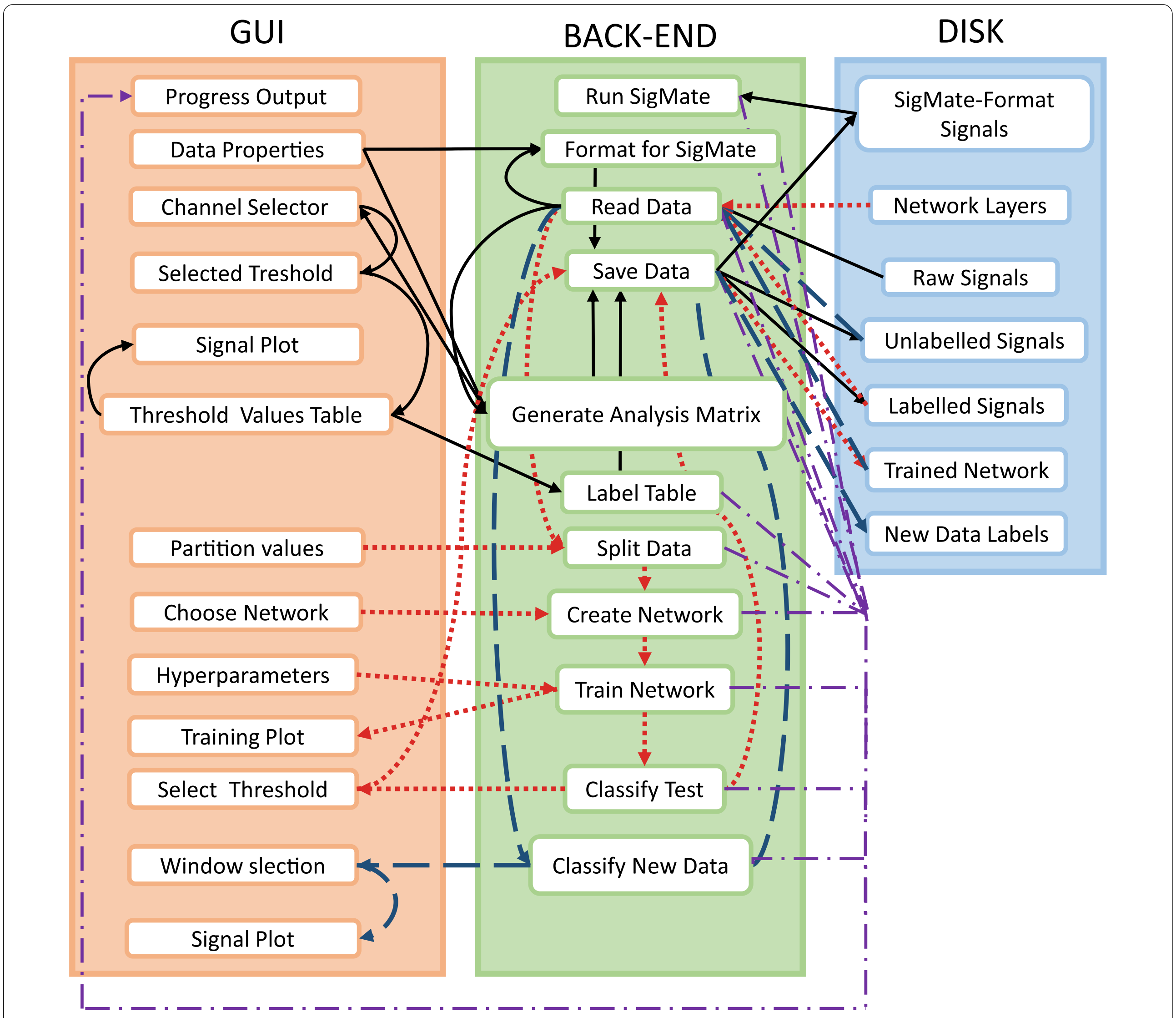


Fig. 5 Functional block diagram of the Toolbox.Arrows in black correspond to the "Data Labeling" module , in red to the "Neural Network Training" module, in dark blue to the "Classify New Unlabeled Data" module, and the purple arrows indicate the progress output

16-contact site electrode from anesthetized rats. At the end of each module, the respective outputs can be exported to a '.mat' file, which can easily be utilized in other applications due to the accessibility of the format.

The following sections describe the individual modules in greater detail. As a visual aid, Fig. 4 shows the screenshots of the software package, Fig. 5 illustrates the function block diagram, and finally, Fig. 6 shows the workflow diagram.

3.2.1 Data labeling

In the first module, the process begins with the 'Load Signals' button, which opens the import wizard to load the neural recordings as an $m \times n$ matrix, where m is the number of channels and n are the data points of each channel signal. The compatible formats include ASCII-based text (.txt, .dat, .out, .csv), spreadsheets files (.xls, .xlsx, .xlsm), and Matab files (.set, .mat), which correspond to 93% of the surveyed data in Fig. 1. The user is required to input the sampling frequency in Hz and the window length in seconds that they wish to analyze. In addition, the unit of the recording and the opportunity to scale is presented, as lots of errors happen due to incorrect annotations of magnitudes.

Once all of these parameters have been filled, 'Generate Analysis Matrix' will structure the data for posterior analysis. This means that given a window length w, and sampling frequency f, the $m \times n$ matrix becomes a new

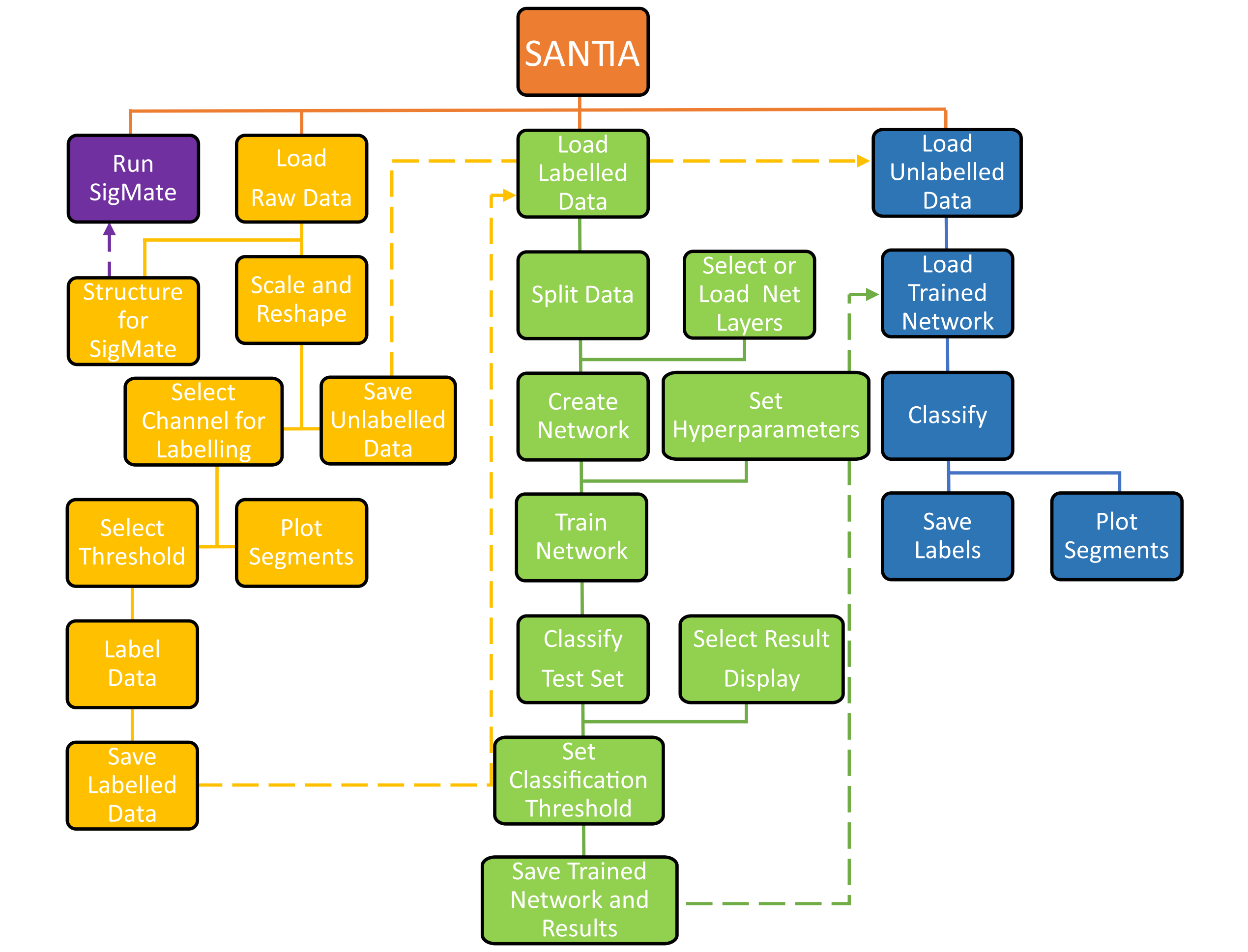


Fig. 6 Workflow of the SANTIA toolbox, where the "Data Labeling" modules are colored yellow, the "Neural Network Training" modules in green, and "Classify New Unlabeled Data" modules in blue

$p \times q$ one, where $p = \frac{m \times n}{w \times f}$ and $q = w \times f$. This is incorporated into a table that has row names that follow the format 'file_id+_channel_+i+_window_j' where file_id is the name of the LFP data file, i the number of channels where $i = 1, ..., m$ and j the corresponding window. In addition, its columns are named: first "window_power" followed by the values of the signal t_k where $k = 1, ..., q$. As this process involves the creation of p amount of row names and window's power, a memory check is done to read available memory and alert if the usage of more than 80% of the available memory would be needed.

The option to save these data for posterior classification is presented as 'Save Unlabeled Data.' Otherwise, the user continues by selecting a channel in the drop-down menu or clicking on a table cell and the 'Threshold Selection Table' process. This opens a new window with the structured data table, and by clicking on a row, the options to plot the selected window or to define its power as a threshold value appear. As a visual aid, windows with same or higher power are colored red and those with less green, i.e., artifactual and normal, respectively.

In another manner, the user can manually input threshold values in the main app's table, and once he has completed it for all channels, the data can be labeled and saved as a standardized struct, which contains the original filename, the structured data with its labels, the sampling frequency, window length, the scale, and the threshold values. This information allows researchers to quickly identify different matrices they create and wish to compare. An aid in form of text in the 'Progress' banner allows the users to know when each step has been completed, and it is replicated throughout each module.

The user can also structure the data for SigMate analysis. The toolbox expects a datapoints(n) × channels(m) format, with the first column as timestamp and each of the channel's signal in the following columns. In addition,

Table 4 Guide to determine best channels and epochs to use of baseline walk and rest recordings in medial prefrontal cortex (mPFC) and the mediodorsal (MD) thalamus, as mentioned in the file named "Coherence Phase Plot Guide"

Rat	mPFC chan1	mPFC chan2	MD chan1	MD chan2	Walk epoch	Rest epoch
KF9	5	6	3	7	960–1160	3780–3820
KF10	3	4	3	8	670–860	1260–1390
KF14	2	6	5	7	740–940	3350–3550
KF15	3	4	5	7	450–640	1600–1700
KF25	2	6	2	5	1480–1680	1700–1800
KF26	1	6	1	6	1180–1380	1050–1150
KF27	2	4	5	8	480–680	2160–2250

The first column is the rat identification, column 2 and 3 the selected two best channels of the mPFC recordings, and 4 and 5 of the MD recordings. Finally, column 6 shows the range of artifact-free epochs during walking and column 7 during resting, respectively [74]

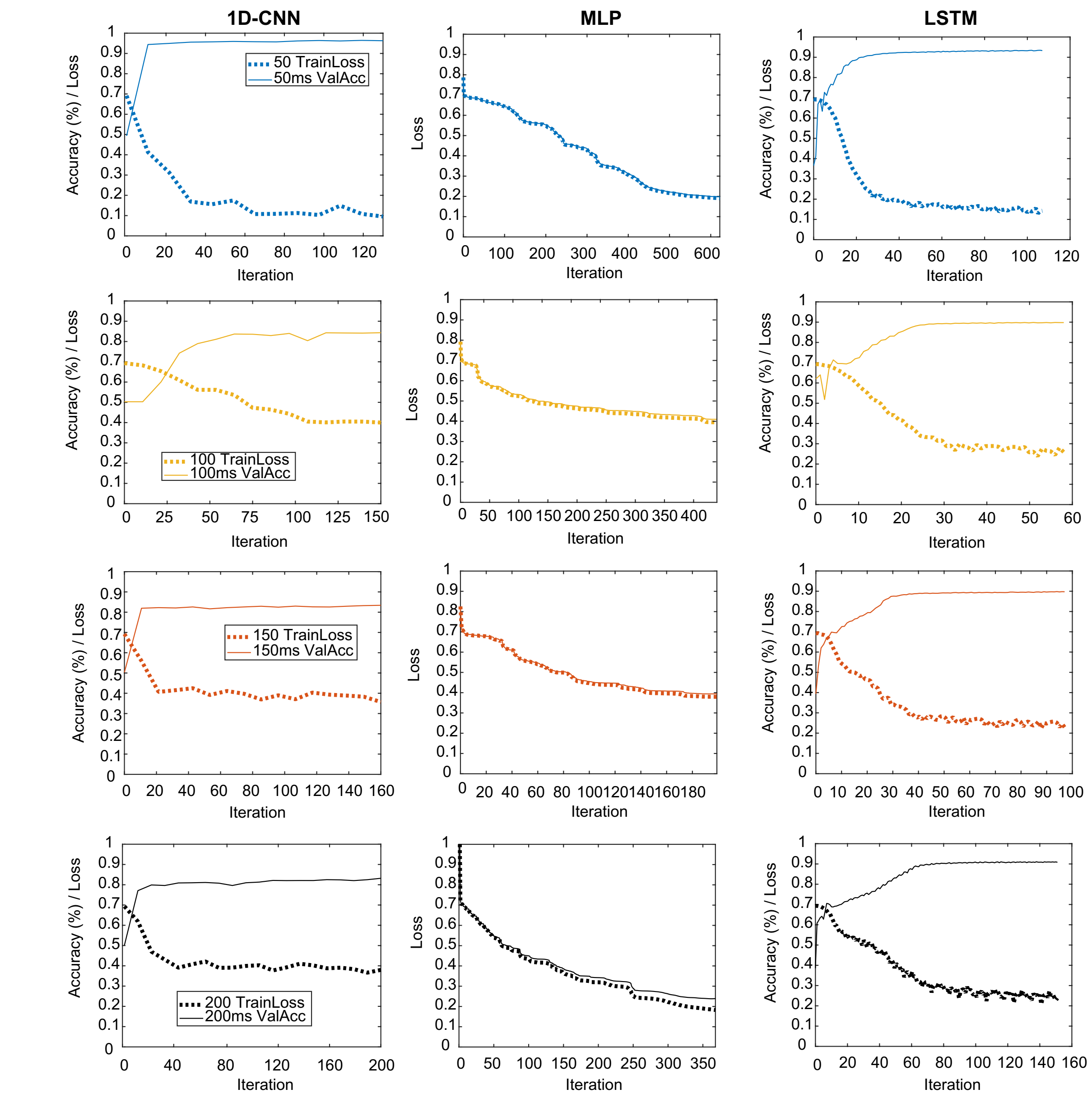


Fig. 7 Training plots for models trained with the first dataset

as it only handles five channels at a time, $m/5$ files have to be generated. Thus, SANTIA transposes the input matrix, generates the timestamp based on the declared sampling frequency, and generates the files. Afterward, it asks the user to select a directory to save them.

3.2.2 Neural network training

The second module starts with loading structured data from the previous module. The user is asked to set the values for training, validation, and test splitting. This is common practice to avoid over- and under-fitting results. As artifacts are rare events, the datasets usually present strong imbalance which can cause bias in the training; a tick box for balancing the data is present next to the 'Split' button. Clicking it generates three datasets with non-repetitive randomized elements from the original matrix.

This is followed by choosing the network, where the options are MLP, LSTM, 1D-CNN, or for the user to load his/her custom set of layers. This is done by choosing a Matlab file which has a Layer-type variable, i.e., layers that define the architecture of neural networks for deep learning, without the pre-trained weights. These can be modified via console or the Deep Network Designer Toolbox, and for more information, we direct the reader to the mathworks page[1]. While employing different architectures might yield better results, it is also possible that they might not be structured properly and lead to under-fitting, over-fitting, or fail to learn at all. Therefore, a limitation of employing custom networks is the time consumption that takes getting the correct combination of layers, as well as setting parameters such as filter size or activation function. Optionally, the user can customize the training hyper-parameters such as the solver, initial learning rate, and execution environment, among others. These intentionally mirror the ones included in the Deep Network Designer to facilitate its usage to those familiarized with it. These are removed for the MLP option, as it uses a different toolbox (i.e., patternnet of Deep Learning Toolbox [73]), which thus does not allow the same configurations. Clicking the 'Create Network" button loads the training options and sets the input layer to match the window size.

The 'Train Network' button runs the train network function, which inherits the training options and network previously defined. For the 1D-CNN, as the deep learning toolbox is intended for images, the 2D matrices are resized to a 4D vector: $1 \times$ window length $\times 1 \times$ number of windows, originally intended to be: width $\times$ height $\times$ channels $\times$ number of examples. A display of the training process automatically appears, unless the user decides not to, which enables monitoring the process and early stopping.

Having completed the training, the user can select whether the 'Classify Test Set' displays the confusion matrix, the area under the receiver-operating characteristic (AUROC) curve, or opens up a new window where the accuracy, F1 score, and confusion matrix appear along with the possibility to modify the classification threshold (set at 0.5 by default). Finally, 'Save Results' creates a struct with data's filename, the trained network, the training information, the test set's classification threshold, AUROC, accuracy, F1 score, and confusion matrix.

3.2.3 Classify new unlabeled data

The last module begins with loading a trained net along with its classification threshold and unlabeled structured data. After its classification, the options to plot each of the windows with the corresponding color-coded label appear. Finally, users can save the labels as a table with the corresponding window name. Having described the toolbox's methods, components, and its functions, we proceed to a test case with real recorded LFP.

4 Results

In this section, we describe the datasets used to test the app, and the results obtained from them. The artifact detection task carried out by SANTIA toolbox was tested on a daily usage grade Acer TravelMate P278-MGlaptop consisting of 8 gigabyte of RAM and Intel$^{®}$Core$^{™}$i7-6500U CPU @ 2.50 GHz processor.

Table 5 First dataset's results for different architectures and sequence length: training loss, validation accuracy, testing accuracy, and testing AUROC

Network	Sequence length (ms)	Training loss	Val. Acc.	Test Acc.	Test AUROC
MLP	50	0.20	0.92	0.92	0.98
	100	0.41	0.82	0.81	0.90
	150	0.39	0.83	0.83	0.90
	200	0.24	0.91	0.91	0.97
1D-CNN	**50**	**0.10**	**0.96**	**0.97**	**0.99**
	100	0.39	0.84	0.84	0.89
	150	0.37	0.83	0.83	0.91
	200	0.36	0.83	0.83	0.91
LSTM	50	0.16	0.93	0.94	0.99
	100	0.26	0.90	0.91	0.97
	150	0.25	0.89	0.90	0.97
	200	0.25	0.91	0.90	0.97

Values pertaining to model's best performance are highlighted in bold

[1] https://uk.mathworks.com/help/deeplearning/ref/nnet.cnn.layer.layer.html

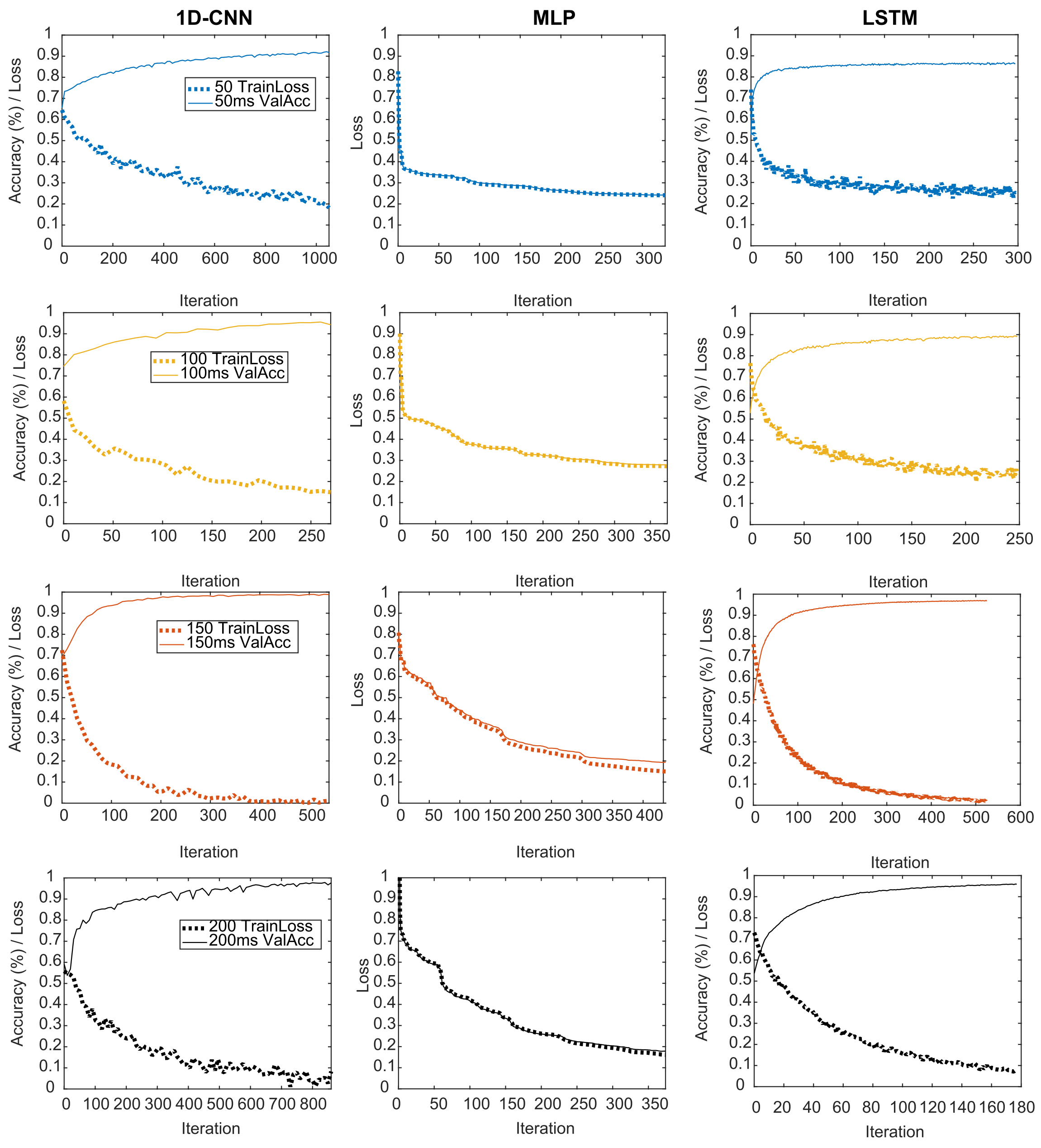


Fig. 8 Training plots for models trained with the second dataset

4.1 Dataset 1

A publicly available dataset [74] was used to test the toolbox. Thorough details of the recording and experiment are explained in the article linked to the dataset [75]. Male Long Evans rats (Charles River, Frederick, MD, USA) weighing from 280 to 300 g were trained to walk on a circular treadmill. The recorded LFP were sampled at 2 kHz, and after low-pass filtering, they were amplified times a thousand and band-pass filtered (0.7–150 Hz).

For the purpose of testing the toolbox, only the baseline recordings (prior to ketamine injection) were used. Baseline recordings were composed of at least two 5-min counter-clockwise walking cycles on a slow-moving treadmill and two 40-s rest periods without artifacts. Visual evaluation and videotaped motor activity were used to classify artifact-free periods of 100 s in treadmill-on epochs and 40 to 100 second periods in treadmill-off epochs, which are detailed in Table 4. These labeled artifact-free epochs were used to extract the threshold power

Table 6 Second dataset's results for different architectures and sequence length: training loss, validation accuracy, testing accuracy, and testing AUROC

Network	Sequence length (ms)	Training loss	Val. Acc.	Test Acc.	Test AUROC
MLP	50	0.24	0.78	0.78	0.857
	100	0.27	0.89	0.86	0.94
	150	0.15	0.94	0.95	0.99
	200	0.16	0.94	0.96	0.98
1D-CNN	50	0.18	0.92	0.91	0.97
	100	0.15	0.94	0.96	0.97
	150	**0.01**	**0.99**	**0.99**	**0.99**
	200	0.08	0.98	0.97	0.99
LSTM	50	0.25	0.86	0.86	0.94
	100	0.26	0.89	0.89	0.96
	150	0.02	0.97	0.97	0.99
	200	0.07	0.96	0.96	0.99

Values pertaining to model's best performance are highlighted in bold

value for each channel. It was chosen as the maximum power of the windows in those intervals, for each respective window size.

To understand the effect of window size on the artifact detection process, different windows of 0.05, 0.1, 0.15, and 0.2 s were taken and fed to the model. The number of examples obtained after downsampling to balance the classes was on average 275, 687 per window size. For the 1D-CNN and LSTM, the optimization algorithm used was Adam, with an initial learning rate of 0.001, the momentum of 0.9, and a batch size of 1280. On the other hand, the MLP was optimized via a scaled conjugate gradient function. The performance of the models during training is shown in Fig. 7. As they originate from different toolboxes, the MLP does not generate the accuracy throughout the training, and thus, it is not shown.

These results are consistent with previously obtained ones. They indicate that since the filters from the 1D-CNN learn from regions of the signal, instead of the individual values, they are able to learn more robust features of the signals and lead to better classification. Performance on the test sets is similar to that obtained in the validation set, as shown in Table 5. The best test set classification results were achieved by the 50 ms 1D-CNN, an accuracy of 96.5%, and an AUROC of 0.993, indicating that the network has been able to learn successfully.

4.2 Dataset 2

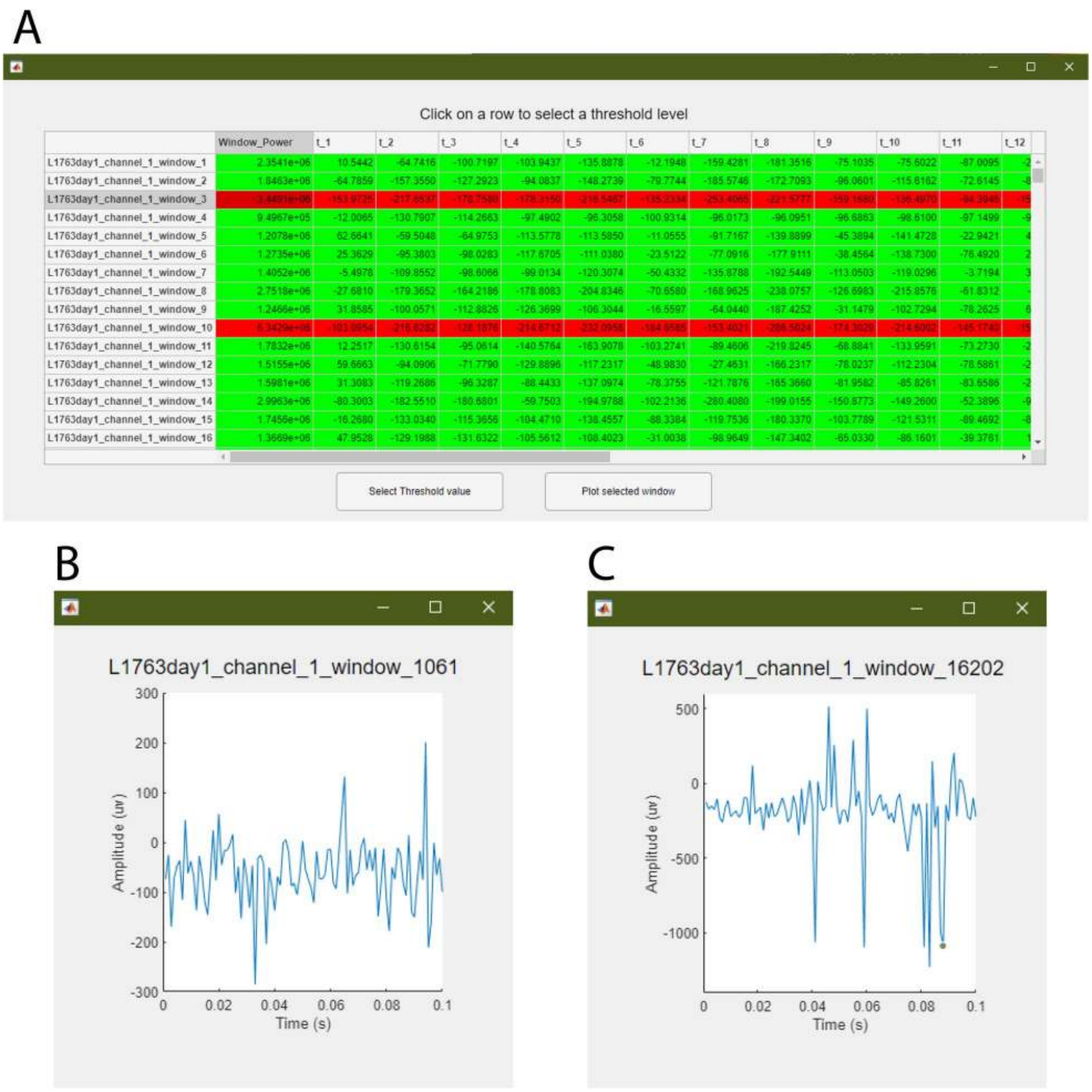
The toolbox was tested using LFP recorded from rats as previously described in [37, 76]. The LFP were downsampled to 1017.3 Hz and low-pass filtered (with a 0–500 Hz cut-off frequency). 294, 592 zero-mean examples were used in this task which were divided into training (80%), validation (10%), and testing (10%) sets, and used to train the models with the same hyper-parameter configuration used with the previous dataset.

Figure 8 displays the performance of the training and validation set of the different sequence lengths for the two architectures, while the results are compiled in Table 6. Overall, the 1D-CNN outperforms the MLP and LSTM across window sizes. Models with input size of 150 ms have the lowest losses and highest accuracies, meaning that it is the best trade-off between information fed the model and its performance, among the chosen window sizes for this dataset. As shown, different datasets are probable to have different optimal trade-offs between window size and accuracy, due to factors such as sampling rate and artifact frequency.

The results are on par with the previous dataset, indicating that the method is robust and possesses generalizability. The 1D-CNN model has shown to obtain the best scores in both cases, establishing it as the better architecture for this type of data.

4.3 Outputs

Figures 9, 10, 11 and 12 show output windows of the toolbox generated during its operation. Figure 9 displays output windows generated after the data file is loaded. They include the selection of threshold, as shown in Fig. 9A, where green lines show windows representing data above the threshold and red lines show below it, and two representative figures of normal (in Fig. 9B) and artifactual windows (in Fig. 9C). Figure 10 shows the output windows for the neural network training process which currently support MLP, LSTM, and 1D-CNN. As the networks come from different Matlab-toolboxes, their individual configurations require separate processes which are represented in Fig. 10A, B for MLP and 1D-CNN/LSTM, respectively. After having completed the training, the different plots of the test set results of the first dataset for the 50 ms window that were generated are shown in Fig. 11. As a part of allowing the user to evaluate the performance of the models, these figures show the confusion matrix (see Fig. 11A), AUROC curve (see Fig. 11B), and accuracy and F1 score for given classification thresholds (see Fig. 11C, D). Finally, Fig. 12 illustrates the contents of output files generated in each module. These files are saved in Matlab format (.mat) and contain key values for the user to quickly access them, as well as the processed variables needed for any posterior predictions.



Click on a row to select a threshold level

	Window_Power	t_1	t_2	t_3	t_4	t_5	t_6	t_7	t_8	t_9	t_10	t_11	t_12
L1763day1_channel_1_window_1	2.3541e+06	10.5442	-64.7416	-100.7197	-103.9437	-135.8878	-12.1948	-159.4281	-181.3516	-75.1035	-75.6022	-87.0095	-2
L1763day1_channel_1_window_2	1.8463e+06	-64.7859	-157.3550	-127.2923	-94.0837	-148.2739	-79.7744	-185.5746	-172.7093	-96.0601	-115.6162	-72.6145	-8
L1763day1_channel_1_window_3	3.4491e+06	-153.9725	-217.8537	-176.7580	-178.3150	-216.5467	-135.2334	-253.4065	-221.5777	-159.1580	-136.4970	-94.3946	-15
L1763day1_channel_1_window_4	9.4967e+05	-12.0065	-130.7907	-114.2663	-97.4902	-96.3058	-100.9314	-96.0173	-96.0951	-96.6863	-98.6100	-97.1499	-9
L1763day1_channel_1_window_5	1.2078e+06	62.6641	-59.5048	-64.9753	-113.5778	-113.5850	-11.0555	-91.7167	-139.8899	-45.3894	-141.4728	-22.9421	4
L1763day1_channel_1_window_6	1.2735e+06	25.3629	-95.3803	-98.0283	-117.6705	-111.0380	-23.5122	-77.0916	-177.9111	-38.4564	-138.7300	-76.4920	2
L1763day1_channel_1_window_7	1.4052e+06	-5.4978	-109.8552	-98.6066	-99.0134	-120.3074	-50.4332	-135.8788	-192.5449	-113.0503	-119.0296	-3.7194	3
L1763day1_channel_1_window_8	2.7518e+06	-27.6810	-179.3652	-164.2186	-178.8083	-204.8346	-70.6580	-168.9625	-238.0757	-126.6983	-215.8576	-61.8312	-
L1763day1_channel_1_window_9	1.2466e+06	31.8585	-100.0571	-112.8826	-126.3699	-106.3044	-16.5597	-64.0440	-187.4252	-31.1479	-102.7294	-78.2625	6
L1763day1_channel_1_window_10	6.3429e+06	-103.0954	-216.6282	-128.1876	-214.8712	-232.0956	-184.9585	-153.4021	-286.5024	-174.3029	-214.6092	-185.1740	-15
L1763day1_channel_1_window_11	1.7832e+06	12.2517	-130.6154	-95.0614	-140.5764	-163.9078	-103.2741	-89.4606	-219.8245	-68.8841	-133.9591	-73.2730	-2
L1763day1_channel_1_window_12	1.5155e+06	59.6663	-94.0906	-71.7790	-129.8896	-117.2317	-48.9830	-27.4631	-166.2317	-78.0237	-112.2304	-78.5861	-2
L1763day1_channel_1_window_13	1.5981e+06	31.3083	-119.2686	-96.3287	-88.4433	-137.0974	-78.3755	-121.7876	-165.3660	-81.9582	-85.8261	-83.6586	-2
L1763day1_channel_1_window_14	2.9963e+06	-80.3003	-182.5510	-180.6801	-59.7503	-194.9788	-102.2136	-280.4080	-199.0155	-150.8773	-149.2600	-52.3896	-9
L1763day1_channel_1_window_15	1.7456e+06	-16.2680	-133.0340	-115.3656	-104.4710	-138.4557	-88.3384	-119.7536	-180.3370	-103.7789	-121.5311	-89.4692	-8
L1763day1_channel_1_window_16	1.3669e+06	47.9528	-129.1988	-131.6322	-105.5612	-108.4023	-31.0038	-98.9649	-147.3402	-65.0330	-86.1601	-39.3761	1

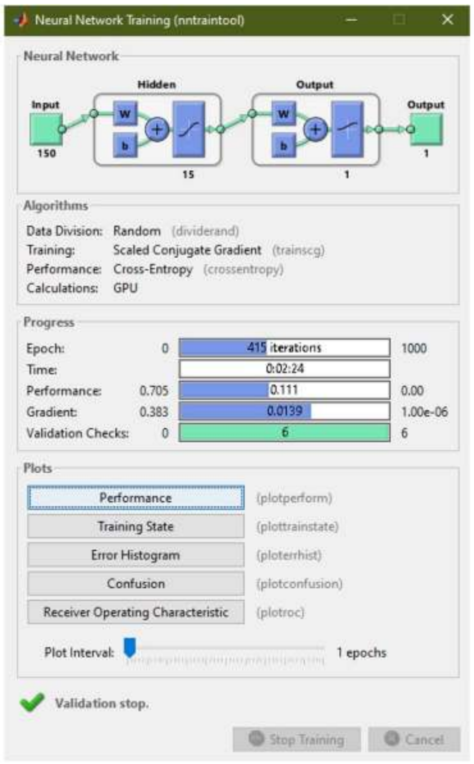
Fig. 9 Screenshots of the toolbox's threshold selection outputs: threshold selection table (**A**), a window of a non-artifactual signal (**B**), and a window of an artifactual signal (**C**)

5 Discussion and conclusion

We developed the SANTIA toolbox to facilitate and standardize the labeling of artifacts in recorded LFP. The simple three-module GUI is designed for researchers without a programming background, and the built-in methodology will allow them to quickly scan and detect the artifacts in their data. It is a project under constant development, and the current version provides an environment where new features can quickly be implemented and adapted to the toolbox. Examples of future developments include:

Online processing The tool currently allows for offline labeling, but we wish to expand it a allow the analysis

A


Neural Network Training (nntraintool)
Neural Network
Hidden
Output
Input
150
15
1
Output
1
Plots
Performance (plotperform)
Training State (plottrainstate)
Error Histogram (ploterrhist)
Confusion (plotconfusion)
Receiver Operating Characteristic (plotroc)
Stop Training
Cancel

Algorithms

Data Division: Random (dividerand)
Training: Scaled Conjugate Gradient (trainscg)
Performance: Cross-Entropy (crossentropy)
Calculations: GPU

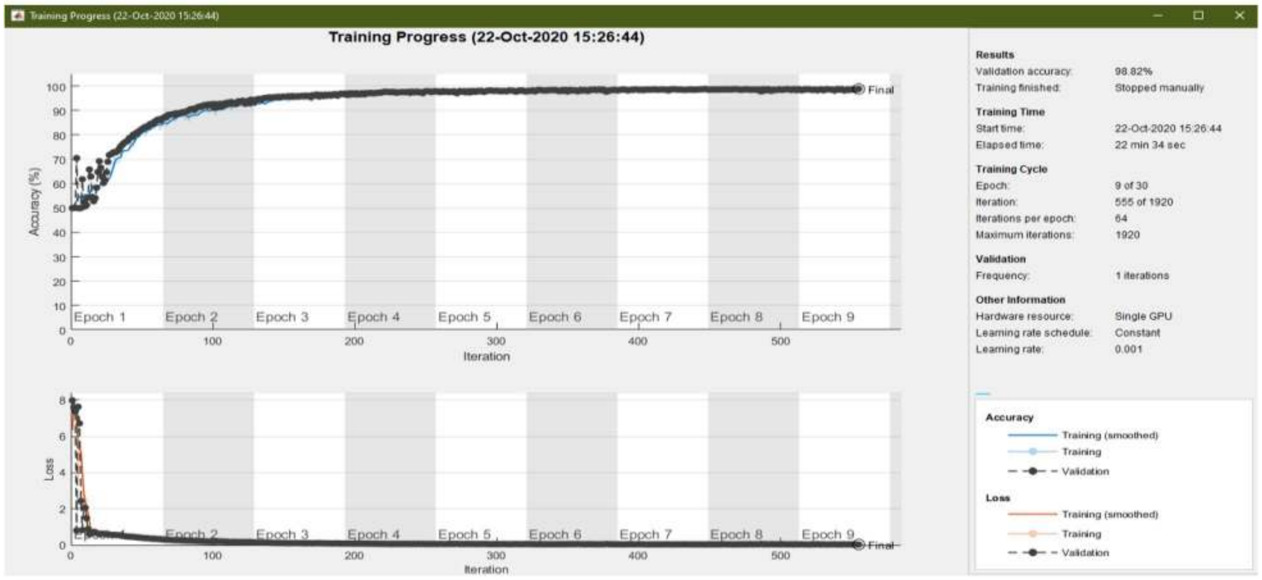
Progress

Epoch:	0	415 iterations	1000
Time:		0:02:24	
Performance:	0.705	0.111	0.00
Gradient:	0.383	0.0139	1.00e-06
Validation Checks:	0	6	6

Plot Interval: 1 epochs

Validation stop.

B



Results
Validation accuracy: 98.82%
Training finished: Stopped manually

Training Time
Start time: 22-Oct-2020 15:26:44
Elapsed time: 22 min 34 sec

Training Cycle
Epoch: 9 of 30
Iteration: 555 of 1920
Iterations per epoch: 64
Maximum iterations: 1920

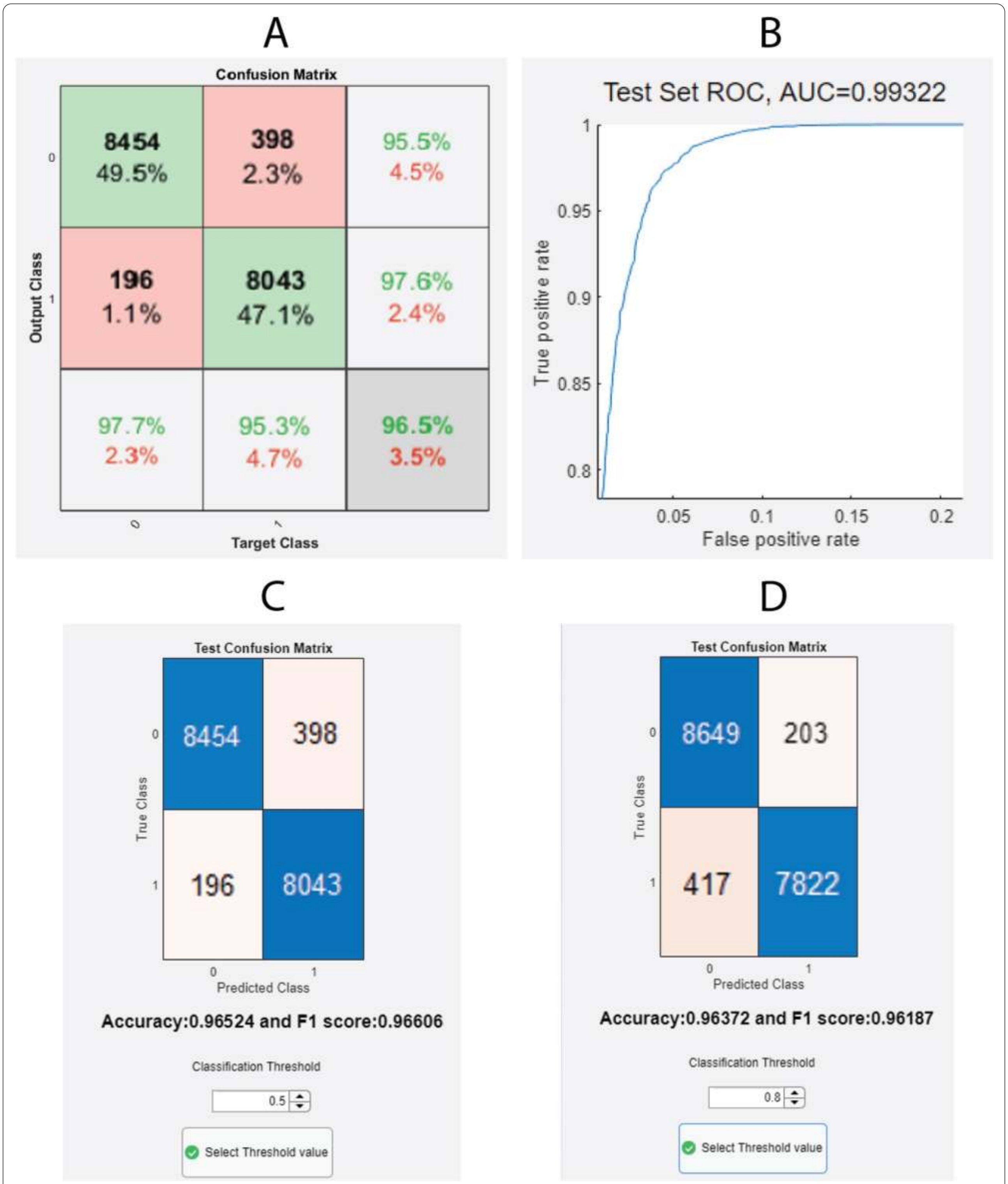
Validation
Frequency: 1 iterations

Other Information
Hardware resource: Single GPU
Learning rate schedule: Constant
Learning rate: 0.001

Fig. 10 Screenshots of the toolbox's network training outputs: multi-layer perceptron training process (**A**) and one dimensional convolutional neural network training process (**B**)

of signals as they are being recorded, to optimize the process.

Expand format compatibility There are different libraries for deep learning such as the TensorFlow-Keras, Caffe, and the ONNX (Open Neural Network Exchange) model formats for neural network layers [46]. We wish to add the possibility to read those formats, and in addition the options to import from and save to HDF5 files for the neuronal data under the epHDF standard [77].



	0	1	
0	8454 49.5%	398 2.3%	95.5% 4.5%
1	196 1.1%	8043 47.1%	97.6% 2.4%
	97.7% 2.3%	95.3% 4.7%	96.5% 3.5%

	0	1
0	8454	398
1	196	8043

Accuracy:0.96524 and F1 score:0.96606

Classification Threshold 0.5

Select Threshold value

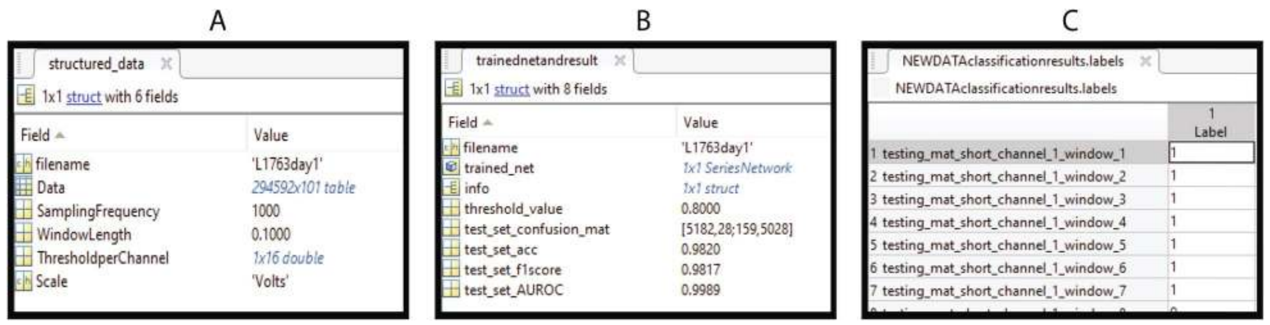
	0	1
0	8649	203
1	417	7822

Accuracy:0.96372 and F1 score:0.96187

Classification Threshold 0.8

Select Threshold value

Fig. 11 Screenshots of the toolbox's network test set results outputs: confusion matrix (**A**), AUROC curve (**B**), threshold selection window with default (**C**), and custom values (**D**)



structured_data

1x1 struct with 6 fields

Field	Value
filename	'L1763day1'
Data	294592x101 table
SamplingFrequency	1000
WindowLength	0.1000
ThresholdperChannel	1x16 double
Scale	'Volts'

trainednetandresult

1x1 struct with 8 fields

Field	Value
filename	'L1763day1'
trained_net	1x1 SeriesNetwork
info	1x1 struct
threshold_value	0.8000
test_set_confusion_mat	[5182,28;159,5028]
test_set_acc	0.9820
test_set_f1score	0.9817
test_set_AUROC	0.9989

NEWDATAclassificationresults.labels

NEWDATAclassificationresults.labels

	1 Label
1 testing_mat_short_channel_1_window_1	1
2 testing_mat_short_channel_1_window_2	1
3 testing_mat_short_channel_1_window_3	1
4 testing_mat_short_channel_1_window_4	1
5 testing_mat_short_channel_1_window_5	1
6 testing_mat_short_channel_1_window_6	1
7 testing_mat_short_channel_1_window_7	1

Fig. 12 Screenshots of the toolbox's saved files: labeled data (**A**), trained network and results (**B**), and new data labels (**C**)

User experience As this app is adopted by the community, with the feedback, we will improve its shortcomings. The inclusion of testing data, a video tutorial and upgrades of the threshold selection to facilitate its use via graphic elements is also planned. The optimization of some routines via parallelism is also a feature we wish to include, due to the possible large sizes of data files.

Multi-modality The incorporation of another source of information (e.g., sensor signal or video) can facilitate and improve the detection of artifacts [78]. A new module would allow the incorporation of such data to facilitate the labeling process or as part of a classification model's input.

Artifact removal Future work will pursue this aspect of artifact analysis as well, with state-of-the-art techniques such as denoising autoencoders [79, 80].

Portability As a long-term goal, we consider the implementation in a portable device, e.g., FPGA or Arduino board, to expand the practicality of its usage.

To conclude, SANTIA now represents an option for researchers looking to label artifacts in LFP recordings automatically. This is a work in progress, and some features are yet to be developed; however, the tests with a public and custom dataset have shown promising results. We hope that the neuroscience community adopts this tool, and with their feedback together with our future plans, an improved toolbox is achieved.

Acknowledgements
The authors would like to express their heartfelt gratitude to the scientists who had kindly released the data from their experiments.

Authors' contributions
This work was done in close collaboration among the authors. MF and MM conceived the idea and designed the initial prototype of the app, MF, MM, AL, MSK and JC refined the app, performed the analysis and wrote the paper. AA, DJG, RJN and MC collected the one of the datasets and edited the paper. All authors have contributed to, seen the paper. All authors read and approved the final manuscript.

Funding
This work was supported by the Nottingham Trent University PhD studentship 2019 which was awarded to Marcos I. Fabietti. Also, by the Science and Technology Project of Beijing Municipal Commission of Education (No. KM201710005026), the National Basic Research Program of China (No. 2014CB744600), and the Beijing Natural Science Foundation (No. 4182005).

Availability of data and materials
The source-code of the toolbox is available at https://github.com/IgnacioFabietti/SANTIAtoolbox.

Declarations

Ethics approval and consent to participate
Not applicable.

Consent for publication
Not applicable.

Competing interests
The authors declare that they have no competing interests.

Author details
[1] Department of Computer Science, Nottingham Trent University, Clifton Lane, Nottingham NG11 8NS, UK. [2] Medical Technologies Innovation Facility, Nottingham Trent University, Clifton Lane, Nottingham NG11 8NS, UK. [3] Computing and Informatics Research Centre, Nottingham Trent University, Clifton Lane, Nottingham NG11 8NS, UK. [4] Institute of Information Technology, Jahangirnagar University, Savar, Dhaka 1342, Bangladesh. [5] Department of Health Sciences, University of Milan, Via di Rudinì, 8, 20142 Milan, Italy. [6] Department of Physical Medicine and Rehabilitation, University of Kansas Medical Center, 3901 Rainbow Blvd, Kansas City 66160, USA. [7] Department of informatics, Bioengineering, Robotics and System Engineering-DIBRIS, University of Genova, Via All'Opera Pia, 13, 16145 Genoa, Italy. [8] Faculty of Information Technology, International WIC Institute, Beijing University of Technology, Beijing 100124, China. [9] Beijing International Collaboration Base on Brain Informatics and Wisdom Services, Beijing 100124, China.

Received: 2 May 2021 Accepted: 29 June 2021
Published online: 20 July 2021

References
1. Delorme A, Makeig S (2004) EEGLAB: an open source toolbox for analysis of single-trial EEG dynamics including independent component analysis. J Neurosci Methods 134(1):9–21
2. Egert U, Knott T, Schwarz C, Nawrot M, Brandt A, Rotter S, Diesmann M (2002) MEA-tools: an open source toolbox for the analysis of multi-electrode data with MATLAB. J Neurosci Methods 117(1):33–42
3. Yger P, Spampinato GL, Esposito E, Lefebvre B, Deny S, Gardella C, Stimberg M, Jetter F, Zeck G, Picaud S et al (2018) A spike sorting toolbox for up to thousands of electrodes validated with ground truth recordings in vitro and in vivo. Elife 7:34518

4. Unakafova VA, Gail A (2019) Comparing open-source toolboxes for processing and analysis of spike and local field potentials data. Front Neuroinform 13:57
5. Tadel F, Baillet S, Mosher JC, Pantazis D, Leahy RM (2011) Brainstorm: a user-friendly application for MEG/EEG analysis. Comput Intell Neurosci. https://doi.org/10.1155/2011/879716
6. Cui J, Xu L, Bressler SL, Ding M, Liang H (2008) BSMART: a Matlab/C toolbox for analysis of multichannel neural time series. Neural Netw 21(8):1094–1104
7. Bokil H, Andrews P, Kulkarni JE, Mehta S, Mitra PP (2010) Chronux: a platform for analyzing neural signals. J Neurosci Methods 192(1):146–151
8. Yegenoglu A et al (2015) Elephant—open-source tool for the analysis of electrophysiological data sets. In: Proc. Bernstein conference, pp 134–135
9. Oostenveld R, Fries P, Maris E, Schoffelen J-M (2011) FieldTrip: open source software for advanced analysis of MEG, EEG, and invasive electrophysiological data. Comput Intell Neurosci. https://doi.org/10.1155/2011/156869
10. Hazan L, Zugaro M, Buzsáki G (2006) Klusters, NeuroScope, NDManager: a free software suite for neurophysiological data processing and visualization. J Neurosci Methods 155(2):207–216
11. Garcia S, Guarino D, Jaillet F, Jennings TR, Pröpper R, Rautenberg PL, Rodgers C, Sobolev A, Wachtler T, Yger P et al (2014) Neo: an object model for handling electrophysiology data in multiple formats. Front Neuroinform 8:10
12. Islam MN, Martin SK, Aggleton JP, O'Mara SM (2019) NeuroChaT: a toolbox to analyse the dynamics of neuronal encoding in freely-behaving rodents in vivo. Wellcome Open Res 4:196
13. Bologna LL et al (2010) Investigating neuronal activity by SPYCODE multichannel data analyzer. Neural Netw 23(6):685–697
14. Mahmud M, Girardi S, Maschietto M, Rahman MM, Bertoldo A, Vassanelli S (2009) Slow stimulus artifact removal through peak-valley detection of neuronal signals recorded from somatosensory cortex by high resolution brain–chip interface. IFMBE Proc 25(4):2062–2065
15. Mahmud M, Bertoldo A, Girardi S, Maschietto M, Vassanelli S (2010) Sigmate: a MATLAB-based neuronal signal processing tool. In: Proc. IEEE EMBC, pp 1352–1355
16. Mahmud M, Bertoldo A, Girardi S, Maschietto M, Pasqualotto E, Vassanelli S (2011) SigMate: a comprehensive software package for extracellular neuronal signal processing and analysis. In: Proc. NER, pp 88–91. https://doi.org/10.1109/NER.2011.5910495
17. Mahmud M, Bertoldo A, Girardi S, Maschietto M, Vassanelli S (2012) SigMate: a Matlab-based automated tool for extracellular neuronal signal processing and analysis. J Neurosci Methods 207(1):97–112. https://doi.org/10.1016/j.jneumeth.2012.03.009
18. Mahmud M, Girardi S, Maschietto M, Vassanelli S (2012) An automated method to remove artifacts induced by microstimulation in local field potentials recorded from rat somatosensory cortex. In: Proc. BRC, pp 1–4. https://doi.org/10.1109/BRC.2012.6222169
19. Mahmud M, Girardi S, Maschietto M, Pasqualotto E, Vassanelli S (2011) An automated method to determine angular preferentiality using LFPs recorded from rat barrel cortex by brain–chip interface under mechanical whisker stimulation. In: Proc. EMBC, pp 2307–2310. https://doi.org/10.1109/IEMBS.2011.6090580
20. Mahmud M, Girardi S, Maschietto M, Rahman MM, Vassanelli S (2009) Noise characterization of electrophysiological signals recorded from high resolution brain–chip interface. In: Proc. ISBB, pp 84–87
21. Mahmud M, Bertoldo A, Maschietto M, Girardi S, Vassanelli S (2010) Automatic detection of layer activation order in information processing pathways of rat barrel cortex under mechanical whisker stimulation. In: Proc. EMBC, pp 6095–6098. https://doi.org/10.1109/IEMBS.2010.5627639
22. Mahmud M, Maschietto M, Girardi S, Vassanelli S (2012) A Matlab based tool for cortical layer activation order detection through latency calculation in local field potentials recorded from rat barrel cortex by brain–chip interface. In: Proc. BRC, pp 1–4. https://doi.org/10.1109/BRC.2012.6222170
23. Mahmud M, Pasqualotto E, Bertoldo A, Girardi S, Maschietto M, Vassanelli S (2011) An automated method for detection of layer activation order in information processing pathway of rat barrel cortex under mechanical whisker stimulation. J Neurosci Methods 196(1):141–150. https://doi.org/10.1016/j.jneumeth.2010.11.024
24. Mahmud M et al (2016) An automated method for characterization of evoked single-trial local field potentials recorded from rat barrel cortex under mechanical whisker stimulation. Cogn Comput 8(5):935–945. https://doi.org/10.1007/s12559-016-9399-3
25. Mahmud M, Travalin D, Bertoldo A, Girardi S, Maschietto M, Vassanelli S (2010) A contour based automatic method to classify local field potentials recorded from rat barrel cortex. In: Proc. CIBEC, pp 163–166. https://doi.org/10.1109/CIBEC.2010.5716087
26. Mahmud M, Travalin D, Hussain A, Girardi S, Maschietto M, Felderer F, Vassanelli S (2012) Single LFP sorting for high-resolution brain-chip interfacing. In Proc. BICS, 7366 LNAI, pp 329–337. https://doi.org/10.1007/978-3-642-31561-9_37
27. Mahmud M, Travalin D, Hussain A (2012) Decoding network activity from LFPS: a computational approach. In: Proc. ICONIP, 7663 LNCS, pp 584–591. https://doi.org/10.1007/978-3-642-34475-6_70
28. Mahmud M, Travalin D, Bertoldo A, Girardi S, Maschietto M, Vassanelli S (2012) An automated classification method for single sweep local field potentials recorded from rat barrel cortex under mechanical whisker stimulation. J Med Biol Eng. https://doi.org/10.5405/jmbe.923
29. Fabietti M, Mahmud M, Lotfi A (2020) Machine learning in analysing invasively recorded neuronal signals: available open access data sources. In: Proc. brain informatics, pp 151–162
30. Destexhe A, Goldberg JA (2013) LFP analysis: overview. In: Jaeger D, Jung R (eds) Encyclopedia of computational neuroscience. Springer, New York, NY, pp 52–55. https://doi.org/10.1007/978-1-4614-6675-8_782
31. Boroujeni KB, Tiesinga P, Womelsdorf T (2020) Adaptive spike-artifact removal from local field potentials uncovers prominent beta and gamma band neuronal synchronization. J Neurosci Methods 330:108485
32. Mikulovic S, Pupe S, Peixoto HM, Do Nascimento GC, Kullander K, Tort AB, Leão RN (2016) On the photovoltaic effect in local field potential recordings. Neurophotonics 3(1):015002
33. Tort AB, Ponsel S, Jessberger J, Yanovsky Y, Brankačk J, Draguhn A (2018) Parallel detection of theta and respiration-coupled oscillations throughout the mouse brain. Sci Rep 8(1):1–14
34. Qian X, Chen Y, Feng Y, Ma B, Hao H, Li L (2016) A method for removal of deep brain stimulation artifact from local field potentials. IEEE Trans Neural Syst Rehabilitat Eng 25(12):2217–2226
35. Brogger J, Eichele T, Aanestad E, Olberg H, Hjelland I, Aurlien H (2018) Visual EEG reviewing times with score EEG. Clin Neurophysiol Pract 3:59–64
36. Yahaya SW, Lotfi A, Mahmud M (2019) A consensus novelty detection ensemble approach for anomaly detection in activities of daily living. Appl Soft Comput 83:105613
37. Fabietti M, Mahmud M, Lotfi A, Averna A, Guggenmo D, Nudo R, Chiappalone M (2020) Neural network-based artifact detection in local field potentials recorded from chronically implanted neural probes. In: Proc. IJCNN, pp 1–8
38. Fabietti M, Mahmud M, Lotfi A, Averna A, Guggenmos D, Nudo R, Chiappalone M (2020) Adaptation of convolutional neural networks for multi-channel artifact detection in chronically recorded local field potentials. In: 2020 IEEE symposium series on computational intelligence (SSCI). IEEE, pp 1607–1613
39. Fabietti M, Mahmud M, Lotfi A (2020) Effectiveness of employing multimodal signals in removing artifacts from neuronal signals: an empirical analysis. In: Proc. brain informatics. Springer, pp 183–193
40. Yahaya SW, Lotfi A, Mahmud M (2021) Towards a data-driven adaptive anomaly detection system for human activity. Pattern Recogn Lett 145:200–207
41. Farhin F, Sultana I, Islam N, Kaiser MS, Rahman MS, Mahmud M (2020) Attack detection in internet of things using software defined network and fuzzy neural network. In: 2020 joint 9th international conference on informatics, electronics & vision (ICIEV) and 2020 4th international conference on imaging, vision & pattern recognition (icIVPR). IEEE, pp 1–6
42. Zaman S, Alhazmi K, Aseeri M, Ahmed MR, Khan RT, Kaiser MS, Mahmud M (2021) Security threats and artificial intelligence based countermeasures for internet of things networks: a comprehensive survey. IEEE Access
43. Fabietti M, Mahmud M, Lotfi A, Averna A, Guggenmos D, Nudo R, Chiappalone M (2021) Signal power affects artefact detection accuracy in chronically recorded local field potentials: preliminary results. In: 2021 10th international IEEE/EMBS conference on neural engineering (NER). IEEE, pp 166–169

44. Tahura S, Samiul SH, Kaiser MS, Mahmud M (2021) Anomaly detection in electroencephalography signal using deep learning model. In: Proceedings of international conference on trends in computational and cognitive engineering. Springer, pp 205–217
45. Mahmud M, Kaiser MS, Hussain A, Vassanelli S (2018) Applications of deep learning and reinforcement learning to biological data. IEEE Trans Neural Netw Learn Syst 29(6):2063–2079. https://doi.org/10.1109/TNNLS.2018.2790388
46. Mahmud M, Kaiser MS, McGinnity MT, Hussain A (2021) Deep learning in mining biological data. Cogn Comput 13(1):1–33. https://doi.org/10.1007/s12559-020-09773-x
47. Noor MBT, Zenia NZ, Kaiser MS, Mahmud M, Al Mamun S (2019) Detecting neurodegenerative disease from MRI: a brief review on a deep learning perspective. In: Proc. brain informatics, pp 115–125
48. Miah Y, Prima CNE, Seema SJ, Mahmud M, Kaiser MS (2021) Performance comparison of machine learning techniques in identifying dementia from open access clinical datasets. In: Proc. ICACIn, pp 79–89
49. Zohora MF, Tania MH, Kaiser MS, Mahmud M (2020) Forecasting the risk of type II diabetes using reinforcement learning. In: Proc. ICIEV. IEEE, pp 1–6
50. Sharpe R, Mahmud M (2020) Effect of the gamma entrainment frequency in pertinence to mood, memory and cognition. In: Proc. brain informatics, pp 50–61
51. Satu MS, Rahman S, Khan MI, Abedin MZ, Kaiser MS, Mahmud M (2020) Towards improved detection of cognitive performance using bidirectional multilayer long-short term memory neural network. In: Proc. brain informatics, pp 297–306
52. Rahman S, Sharma T, Mahmud M (2020) Improving alcoholism diagnosis: comparing instance-based classifiers against neural networks for classifying EEG signal. In: Proc. brain informatics, pp 239–250
53. Noor MBT, Zenia NZ, Kaiser MS, Al Mamun S, Mahmud M (2020) Application of deep learning in detecting neurological disorders from magnetic resonance images: a survey on the detection of Alzheimer's disease, Parkinson's disease and schizophrenia. Brain Inform 7:1–21
54. Aradhya VNM, Mahmud M, Guru DS, Agarwal B, Kaiser MS (2021) One shot cluster based approach for the detection of COVID-19 from chest X-ray images. Cogn Comput. https://doi.org/10.1007/s12559-020-09774-w
55. Dey N, Rajinikanth V, Fong SJ, Kaiser MS, Mahmud M (2020) Social group optimization-assisted Kapur's entropy and morphological segmentation for automated detection of covid-19 infection from computed tomography images. Cogn Comput 12(5):1011–1023
56. Al Banna MH, Ghosh T, Taher KA, Kaiser MS, Mahmud M (2020) A monitoring system for patients of autism spectrum disorder using artificial intelligence. In: Proc. brain informatics, pp 251–262
57. Sumi AI, Zohora MF, Mahjabeen M, Faria TJ, Mahmud M, Kaiser MS (2018) fASSERT: a fuzzy assistive system for children with autism using internet of things. In: Proc. brain informatics, pp 403–412
58. Tonni SI, Aka TA, Antik MM, Taher KA, Mahmud M, Kaiser MS (2021) Artificial intelligence based driver vigilance system for accident prevention. In: 2021 international conference on information and communication technology for sustainable development (ICICT4SD). IEEE, pp 412–416
59. Al Nahian MJ, Ghosh T, Uddin MN, Islam MM, Mahmud M, Kaiser M (2020) Towards artificial intelligence driven emotion aware fall monitoring framework suitable for elderly people with neurological disorder. In: Proc. brain informatics, pp 275–286
60. Jesmin S, Kaiser MS, Mahmud M (2020) Artificial and internet of healthcare things based Alzheimer care during COVID 19. In: Proc. brain informatics, pp 263–274
61. Nahiduzzaman M, Tasnim M, Newaz NT, Kaiser MS, Mahmud M (2020) Machine learning based early fall detection for elderly people with neurological disorder using multimodal data fusion. In: Proc. brain informatics, pp 204–214
62. Kaiser MS, Mahmud M, Noor MBT, Zenia NZ, Al Mamun S, Mahmud KA, Azad S, Aradhya VM, Stephan P, Stephan T et al (2021) iWorksafe: towards healthy workplaces during COVID-19 with an intelligent phealth app for industrial settings. IEEE Access 9:13814–13828
63. Orojo O, Tepper J, McGinnity TM, Mahmud M (2019) A multi-recurrent network for crude oil price prediction. In: Proc. IEEE SSCI, pp 2953–2958
64. Ali HM, Kaiser MS, Mahmud M (2019) Application of convolutional neural network in segmenting brain regions from MRI data. In: International conference on brain informatics, pp. 136–146
65. Ruiz J, Mahmud M, Modasshir M, Shamim Kaiser M (2020) Alzheimer's disease neuroimaging initiative, f.t.: 3D DenseNet ensemble in 4-way classification of Alzheimer's disease. In: Proc. brain informatics, pp 85–96
66. Rabby G, Azad S, Mahmud M, Zamli KZ, Rahman MM (2020) Teket: a tree-based unsupervised keyphrase extraction technique. Cogn Comput 12:811–833
67. Watkins J, Fabietti M, Mahmud M (2020) Sense: a student performance quantifier using sentiment analysis. In: Proc. IJCNN, pp 1–6
68. Chen L, Yan J, Chen J, Sheng Y, Xu Z, Mahmud M (2020) An event based topic learning pipeline for neuroimaging literature mining. Brain Inform 7(1):1–14
69. Bukhtiyarova O, Soltani S, Chauvette S, Timofeev I (2016) Supervised semi-automatic detection of slow waves in non-anaesthetized mice with the use of neural network approach. Transl Brain Rhythmicity 1(1):14–18
70. Kanal LN (2001) Perceptrons. In: Smelser NJ, Baltes PB (eds) International encyclopedia of the social & behavioral sciences. Pergamon, Oxford, pp 11218–11221. https://doi.org/10.1016/B0-08-043076-7/00572-6
71. Fabietti M, Mahmud M, Lotfi A, Averna A, Guggenmo D, Nudo R, Chiappalone M (2020) Artifact detection in chronically recorded local field potentials using long-short term memory neural network. In: Proc. AICT, pp 1–6
72. Fabietti M, Mahmud M, Lotfi A, Averna A, Guggenmo D, Nudo R, Chiappalone M (2020) Adaptation of convolutional neural networks for multi-channel artifact detection in chronically recorded local field potentials. In: Proc. SSCI, pp 1–7
73. Arulmozhi V (2011) Classification task by using matlab neural network tool box—a beginner's view. Int J Wisdom Based Comput 1(2):59–60
74. Furth K (2017) Replication Data for: neuronal correlates of ketamine and walking induced gamma oscillations in the medial prefrontal cortex and mediodorsal thalamus. Harvard Dataverse. https://doi.org/10.7910/DVN/MIBZLZ
75. Furth KE, McCoy AJ, Dodge C, Walters JR, Buonanno A, Delaville C (2017) Neuronal correlates of ketamine and walking induced gamma oscillations in the medial prefrontal cortex and mediodorsal thalamus. PLoS ONE 12(11):0186732
76. Averna A et al (2020) Differential effects of open- and closed-loop intracortical microstimulation on firing patterns of neurons in distant cortical areas. Cereb Cortex 30(5):2879–2896. https://doi.org/10.1093/cercor/bhz281
77. Teeters J, Sommer FT (2013) epHDF-a standard for storing electrophysiology data in HDF5. F1000Research. https://doi.org/10.3389/conf.fninf.2013.09.00068
78. Fabietti M, Mahmud M, Lotfi A (2020) Effectiveness of employing multimodal signals in removing artifacts from neuronal signals: an empirical analysis. In: Proc. brain informatics, pp 183–193
79. Ghosh R, Sinha N, Biswas SK (2018) Automated eye blink artefact removal from EEG using support vector machine and autoencoder. IET Signal Process 13(2):141–148
80. Leite NMN, Pereira ET, Gurjao EC, Veloso LR (2018) Deep convolutional autoencoder for EEG noise filtering. In: Proc. BIBM, pp 2605–2612

Publisher's Note