Neural Activity Classification with Machine Learning Models Trained on Interspike Interval Time-Series Data

Ivan Lazarevich
Group for Neural Theory, Laboratoire de Neurosciences Cognitives,
École Normale Supérieure,
Paris, France
Lobachevsky State University,
Nizhny Novgorod, Russia
ivan@lazarevi.ch

Ilya Prokin
Aptadeep,
Paris, France

Boris Gutkin
Group for Neural Theory, Laboratoire de Neurosciences Cognitives,
École Normale Supérieure,
Paris, France
Center for Cognition and Decision Making,
Higher School of Economics,
Moscow, Russia

Victor Kazantsev
Lobachevsky State University,
Nizhny Novgorod, Russia

Modern well-performing approaches to neural decoding are based on machine learning models such as decision tree ensembles and deep neural networks. The wide range of algorithms that can be utilized to learn from neural spike trains, which are essentially time-series data, results in the need for diverse and challenging benchmarks for neural decoding, similar to the ones in the fields of computer vision and natural language processing. In this work, we propose a spike train classification benchmark, based on open-access neural activity datasets and consisting of several learning tasks such as stimulus type classification, animal's behavioral state prediction and neuron type identification. We demonstrate that an approach based on hand-crafted time-series feature engineering establishes a strong baseline performing on par with state-of-the-art deep learning based models for neural decoding. We release the code allowing to reproduce the reported results [1].

Introduction

The latest advances in multi-neuronal recording technologies such as two-photon calcium imaging [1], extracellular recordings with multi-electrode arrays [2], Neuropixels probes [3] allow producing large-scale single-neuron resolution brain activity data with remarkable magnitude and precision. Some of the neural spiking data recorded in animals has been released to the public in the scope of data repositories such as CRCNS.org [4]. In addition to increasing experimental data access, various neural data analysis tools have been developed, in particular for the task of neural decoding, which is often posed as a supervised learning problem [5]: given firing activity of a population of neurons at each time point, one has to predict the value of a certain quantity pertaining to animal's behaviour such as its velocity at a given point in time.

Such a formulation of the neural decoding task implies that it is a multivariate time-series regression or classification problem. An array of supervised learning methods focused specifically on general time-series data has been developed over the years, ranging from classical approaches [6] to deep neural networks for sequential data [7]. It is not fully clear, however, how useful these methods are for the specific tasks of learning from neural spiking data. In order to establish a sensible ranking of these algorithms for neural decoding, there is a need for a common spiking activity recognition benchmark. In this work, we propose a diverse and challenging spike train classification benchmark based on several open-access neuronal activity datasets. This benchmark incorporates firing activity from different brain regions of different animals (retina, prefrontal cortex, motor and visual cortices) and comprises distinct task types such as visual stimulus type classification, animal's behavioral state prediction from individual spike trains and interneuron subtype recognition from firing patterns. All of these tasks are formulated as univariate time-series classification problems, that is, one needs to predict the target category based on an individual spike train recorded from a single neuron. The formulation of the classification problems implies that the predicted category is stationary across the duration of the given spike train sample.

Our main contributions can be summarized as follows:

- We propose a diverse spike train classification benchmark based on open-access data.
- We show that global information such as the animal's behavioral state or stimulus type can be decoded (with high accuracy) from *single-neuron* spike trains

[1] https://github.com/vanyalzr/pyspikelib

containing several tens of interspike intervals.

- We establish a strong baseline for spike train classification based on hand-crafted time-series feature engineering that performs on par state-of-the-art with deep learning models.

Related works

Well-established machine learning techniques such as gradient boosted decision tree ensembles and recurrent neural networks have been successfully applied both to neural activity decoding (predicting stimuli/action from spiking activity) [5] as well as neural encoding (predicting neural activity from stimuli) [8]. Neural decoding tasks are often formulated as regression problems, whereas binned spiking count time-series of a single fixed neural population are used to predict the animal's position or velocity in time.

A number of previous studies on feature vector representations of spike trains also focused on defining a spike train distance metric [9] for identification of neuronal assemblies [10]. Several different definitions of the spike train distance exist such as van Rossum distance [11], Victor-Purpura distance [12], SPIKE- and ISI- synchronization distances [13] (for a thorough list of existing spike train distance metrics see [9]). These distance metrics were used to perform spike train clustering and classification based on the k-Nearest-Neighbors approach [14]. Jouty et al. [15] employed ISI and SPIKE distance measures to perform clustering of retinal ganglion cells based on their firing responses to a given stimulus.

In addition to characterization with spike train distance metrics, some previous works relied on certain statistics of spike trains to differentiate between cell types. Charlesworth et al. [16] calculated basic statistics of multi-neuronal activity from cortical and hippocampal cultures and were able to perform clustering and classification of activity between these culture types. Li et al. [17] used two general features of the interspike interval (ISI) distribution to perform clustering analysis to identify neuron subtypes. Such approaches represent neural activity (single or multi-neuron spiking patterns) in a low-dimensional feature space where the hand-crafted features are defined to address specific problems and might not provide an optimal feature representation of spiking activity data for a general decoding problem. Finally, not only spike timing information can be used to characterize neurons in a supervised classification task. Jia et al. [18] used waveform features of extracellularly recorded action potentials to classify them by brain region of origin.

The aforementioned works were aimed at, to some extent or another, trying to decode properties of neurons or stimuli given recorded spiking data. In some of the cases the datasets used were not released to be openly available, in some of the cases the predictive models used constituted quite simple baselines for the underlying decoding/cell identification tasks. In this work, we aim to propose a benchmark base on open-access datasets that is diverse and challenging enough to robustly demonstrate gains of advanced time-series machine learning approches as compared to some of the simple baselines used in previous works.

I. Methods

A. Overview of time series classification methods

We applied general time series feature representation methods [6] for classification of neuronal spike train data. Most approaches in time series classification are focused on transforming the raw time series data into an effective feature space representation before training and applying a machine learning classification model. Here we give a brief overview of state-of-the-art approaches one could utilize in order to transform time series data into a feature vector representation for efficient neural activity classification.

1) Neighbor-based models with time series distance measures: A strong baseline algorithm for time series classification is k-nearest-neighbors (kNN) with a suitable time series distance metric such as the Dynamic Time Warping (DTW) distance or the edit distance (ED) [6]. In this work, we evaluated performance of nearest-neighbor models for generic distance measures such as l_p and DTW distance, converting spike trains to the interspike-interval (ISI) time-series representation prior to calculating the spike-train distances. Some of the distance metrics we also used for evaluation are essentially distribution similarity measures (e.g. Kolmogorov-Smirnov distance, Earth mover's distance) which allow comparing ISI value distiributions within spike trains. Such a spike train distance definition would only use the information about the ISI distribution in the spike train, but not about its temporal structure. Alternatively, one can keep the original event-based representation of the spike train and compute the spike train similarity metrics such as van Rossum or Victor-Purpura distances or ISI/SPIKE distances [9].

The choice of the distance metric determines which features of the time series are considered as important. Instead of defining a complex distance metric, one can explicitly transform time series into a feature space by calculating various properties of the series that might be important (e.g. mean, variance). After assigning appropriate weights to each feature one can use kNN with any standard distance metric. Moreover, such a representation allows the application of any state-of-the-art machine learning classification algorithm beyond kNN to obtain better classification results. In the following, we discuss approaches using various feature space representations available for time series data.

2) Models using hand-crafted time series features: One of the useful and intuitive approaches in time series classification is focused on manually calculating a set of descriptive features for each time series (e.g. their basic statistics, spectral properties, other measures used in signal processing and so on) and using these feature

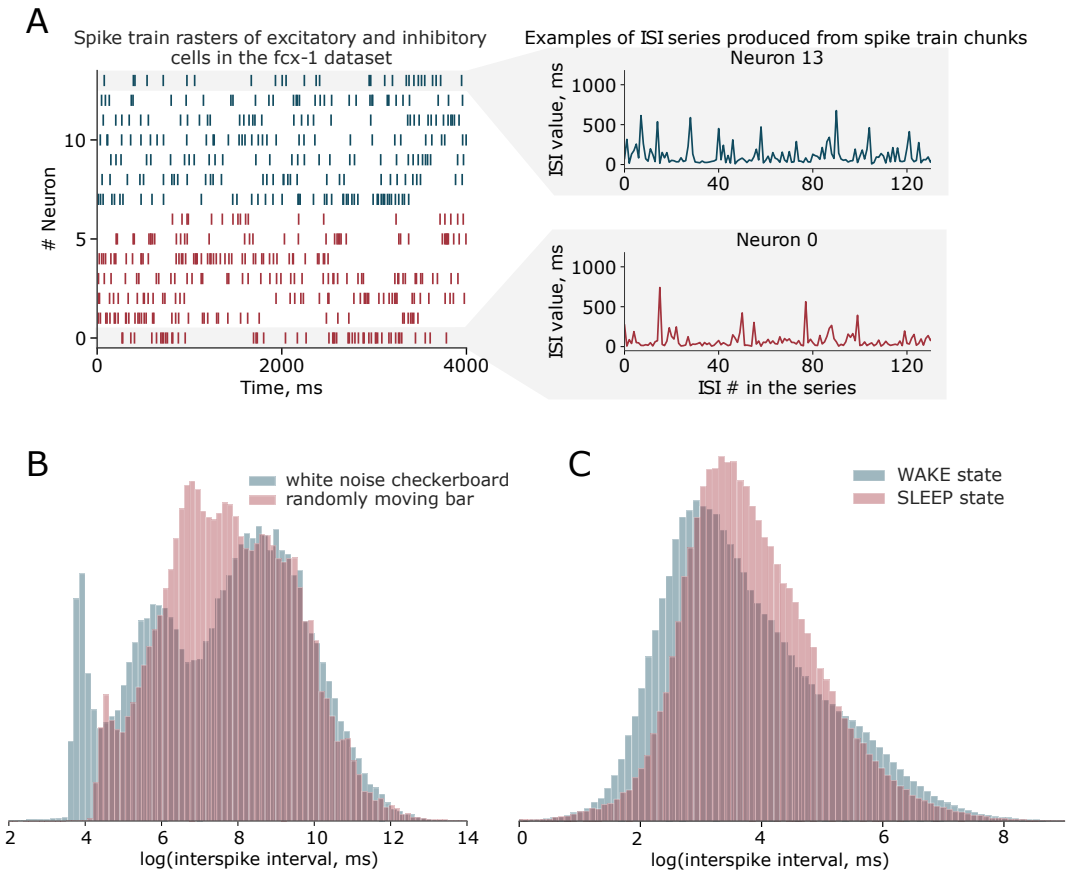

Fig. 1. (A) Examples of spiking activity recordings in the CRCNS fcx-1 dataset in the WAKE state. Left: spike train raster of a random subset of excitatory cells (red) and inhibitory cells (blue). Right: examples of ISI series produced from spike train chunks of inhibitory/excitatory cells in the fcx-1 dataset. (B) Interspike interval value distribution histograms generated from the aggregated spike trains of retinal ganglion cells in response to a "white noise checkerboard" visual stimulus (blue) and a "randomly moving bar" stimulus (red). (C) Interspike interval value distribution histograms generated from the aggregated PFC spike trains (fcx-1 dataset) corresponding to the WAKE (blue) or SLEEP (red) state of the rat.

sets as vectors describing each sample series. There exist approaches which enable automated calculation of a large number of time series features which may be typically considered in different application domains. Such approaches include automated time series phenotyping implemented in the *hctsa* MATLAB package [19] and automated feature extraction in the *tsfresh* Python package [20]. Here we utilize the *tsfresh* package which enables calculation of 779 descriptive time series features for each spike train, ranging from Fourier and wavelet expansion coefficients to coefficients of a fitted autoregressive process.

Once each time series (spike train) is represented as a feature vector, the spiking activity dataset has the standard form of a matrix with size $[n_{\text{samples}}, n_{\text{features}}]$ rather than the raw dataset with shape $[n_{\text{samples}}, n_{\text{timestamps}}]$. This standardized dataset can be then used as an input to any machine learning algorithm such as logistic regression or gradient boosted trees [21]. We found this approach to set a strong baseline for all of the classification tasks we considered.

3) Deep learning models: Lastly, there are deep learning based approaches working well for time-series [22] such as deep recurrent networks like Long Short-Term Memory (LSTM) models [23], one-dimensional convolutional neural networks (1D-CNNs) [24] that achieve good results on generic time series datasets [25]. Some deep learning architectures like InceptionTime [26] were developed specifically for time series data and represent state-of-the-art among deep learning models on time-series classification benchmarks such as the UCR time series archive [27].

B. The proposed spike train classification benchmark.

We propose a spike train classification benchmark comprising of several different open-access dataset and distinct

classification tasks. The datasets used for the benchmark are as follows:

- **Retinal ganglion cell stimulus type classification** based on the published dataset [28], [29]: Spike time data from multi-electrode array recordings of salamander retinal ganglion cells under four stimulus conditions: a white noise checkerboard, a repeated natural movie, a non-repeated natural movie, and a bar exhibiting random one-dimensional motion. We define the 4-class classification task to predict the stimulus type given the spike train chunk, also considering binary classification tasks for pairs of stimuli types (e.g. "white noise checkerboard" vs. "randomly moving bar").
- **WAKE/SLEEP classification** based on fcx-1 dataset [30], [31] from CRCNS.org [4]: Spiking activity and Local-Field Potential (LFP) signals recorded extracellularly from frontal cortices of male Long Evans rats during wake and sleep states without any particular behavior, task or stimulus. Around 1100 units (neurons) were recorded, 120 of which are putative inhibitory cells and the rest is putative excitatory cells. Figure 1 shows several examples of spiking activity recordings that can be extracted from the fcx-1 dataset. The authors classified cells into an inhibitory or excitatory class based on the action potential waveform (action potential width and peak time). Sleep states (SLEEP activity class) were labelled semi-automatically based on extracted LFP and electromyogram features, and the non-sleep state was labelled as the WAKE activity class. We define the binary classification task as the prediction of WAKE or SLEEP animal state given a spike train chunk recorded from a putative excitatory cell.
- **Interneuron subtype classification task** based on the Allen Cell Types dataset [32]: Whole cell patch clamp recordings of membrane potential in neurons of different types. We selected the PV, VIP and SST interneurons from the whole dataset, as these interneuron groups comprise the majority of inhibitory cells in the prefrontal cortex [33]. We selected the spike trains recorded under the naturalistic noise stimulation protocol (as a proxy for the *in vivo* spontaneous activity in these cells). The non-trivial prediction task is defined for VIP vs. SST spike train classification, since the PV interneuron spike trains can be easily distinguished from the other interneuron types. The latter is due to a significantly higher firing frequency in PV interneurons that we found in the Allen Cell Types dataset.

C. Cross-validation scheme and data preprocessing

Suppose we are given a dataset containing data from several mice each recorded multiple times with a large number of neurons captured in each recording. For each recorded neuron, we have a corresponding spike train captured over a certain period of time (assuming that the preprocessing steps like spike sorting or spiking time inference from fluorescence traces were performed beforehand). The number of spikes within each spike-train is going to be variable. A natural way to standardize the length of spike-train sequences would be dividing the full spike train into chunks of N_{size} spike times, where N_{size} is fixed for each chunk. The chunks can be can be produced by moving a sliding window across the spike-timing-vector. Thus, each neuron would contribute a different number of spike-timing-chunks depending on its average firing rate. This sliding window procedure can be applied to both the sequences of inter-spike intervals (ISIs) and time-binned spike count time series. We empirically investigate the advantages of using either ISI-encoding or spike-count encoding of spike trains further in this work.

The cross-validation strategy we use in thus work is based on group splits, whereby we determine the split into the training and the testing datasets based on neuron/animal identifiers available in the original data. The motivation is that, in cases recordings from several animals and corresponding animal identifiers are available, the set of animals used to construct the training dataset and the set of animals for the testing dataset should not overlap in order to test whether the trained decoding models could generalize across different animals. In case animal identifiers are not available, we split the dataset into training and testing based on non-overlapping neuron identifiers in train and test. We perform several random train/test splits for all datasets based on animal/neuron groups to evaluate the variance of classification metrics across splits.

The metrics we use to evaluate model performance in classification tasks is accuracy and AUC-ROC. The testing datasets are balanced by undersampling to mitigate the influence of class imbalance on the ranking properties of accuracy and AUC-ROC metrics. Since the class distribution in the testing datasets is balanced by construction, the values of these metrics reflect the real classification performance of the trained models relative to the chance level of 0.5.

The base task we consider for all benchmark datasets is classification given an individual spike train. However, prediction performance can be improved by aggregating predictions from several spike trains (in case such data is available) or from several sub-sequences of a single large spike train. If the final classification in such a setting is done by majority voting from all single spike-train predictions and we assume that recorded spike-trains (neurons) are randomly sampled from the whole ensemble, the optimistic estimate for accuracy improvement with the number of spike trains (neurons) N_{trains} would be

$$\mu = \sum_{i=m}^{N_{\text{trains}}} C^{i}_{N_{\text{trains}}} p^{i} (1-p)^{N_{\text{trains}}-i} \tag{1}$$

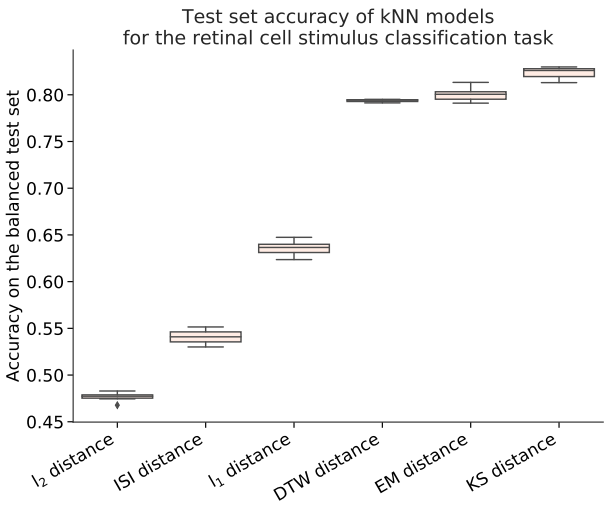

Fig. 2. Spike train classification accuracy values for the retinal neuron activity dataset for nearest-neighbor models with different distance metrics . The task is defined as binary classification of the stimulus type ("white noise checkerboard" or "randomly moving bar"), with the test set balanced in class distribution (that is, accuracy=0.5 corresponds to chance level).

where μ is the probability that the majority vote prediction is correct, p is the probability of a single classifier prediction being correct (single spike train prediction accuracy), N_{trains} is the number of predictions made, $m = [N_{trains}/2] + 1$ is the minimal majority of votes. We found that the empirical values of accuracy improvement are close to the optimistic analytical estimate (1) in both cases when the spike train chunks are sampled from different neurons and from a large spike train of a single neuron.

II. Results

A. Visual stimulus type classification from retinal spike trains

We first start with looking the at the retinal ganglion cell spike train classification task. Recorded spike trains in the dataset are associated with one of the four categories corresponding to different visual stimulus types, labelled with "white noise checkerboard", "randomly moving bar", "repeated natural movie" and "unique natural movie". The classification task is, given a chunk of the spike train recording, predict the corresponding stimulus type category. The number of neurons in the dataset belonging to each category is 155, 140, 178, 152, respectively. The number of interspike intervals is quite variable among individual cells (due to firing rate variability) ranging from 100 ISIs per recording to as much as 60000 ISIs per recording.

We focus on the binary classification task aimed at predicting one of the two types of stimuli: "randomly moving bar" or "white noise checkerboard". We select recorded spike trains corresponding to those stimuli types and split 50% of recorded neurons for the training part of the dataset and the remaining 50% for the testing dataset. We encode spike trains using the ISI representation and apply a rolling window of size equal to 200 ISIs with a stride of 100 ISIs to earch recorded neuron. We then perform undersampling to make the class distribution balanced and arrive at a dataset of 5188 training and 5272 testing examples (each containing 200 spikes), with an equal amount of spike-trains corresponding to both stimuli types in the training and testing datasets.

Nearest-neighbor models for spike train classification: We evaluated performance of nearest-neighbor models with different distance metrics on the retinal stimulus classification task, results are shown on Figure 3. We found that the nearest neighbor model with the DTW distance is amongst the best performing ones, but is still out performed by 1-NN models with Kolmogorov-Smirnov and Earth Mover's distances, suggensting that differences in ISI distributions contain discriminative information helpful for the classification task.

Hand-crafted feature extraction + classification models: The kNN results clearly suggest that characteristics of the interspike-interval distribution of the given spike train are predictive of the category label in our classification task. At the same time, one would expect the the temporal (sequential) information contained in the spike train also has certain predictive power. A straightforward way to incorporate both types of features in the model is to build a corresponding vector embedding of the spike train time-series. An efficient way to do so is to use a set of hand-crafted time-series features, like for example the set of 779 features provided in the *tsfresh* Python package. In order to compute vector embeddings for the spike trains in the training and testing datasets, one has to convert spike times into a time-series, which in principle could be done using either an interspike-interval encoding (the time-series is the sequence of ISIs) or a spike-count encoding (time is binned and spike counts in each time bin comprise the time series). The latter type of encoding depends on an additional hyperparameter which is the size of the time bin while ISI-encoding is parameter-free. We tested both types of spike-train encoding for our task and observed that on average models trained using the ISI-encoding of spikes perform better than the ones using binned spike counts. Furthermore, we found that combining features corresponding to both encoding types leads to better performance compared to using a single encoding scheme (see Figure 3).

For each spike-train encoding type, we computed the 779-dimensional *tsfresh* time-series embeddings independently for each sample in the training and testing datasets (no statistic aggregation across samples is performed). We then performed simple pre-processing steps by (i) removing low-variance features from the embedding (features f satisfying $\mathrm{std}(f)/(\mathrm{mean}(f) + \varepsilon) < \theta$ with $\theta = 0.2$ and $\varepsilon = 10^{-9}$ were removed) and (ii) performing standard scaling for each feature using mean and variance statistics

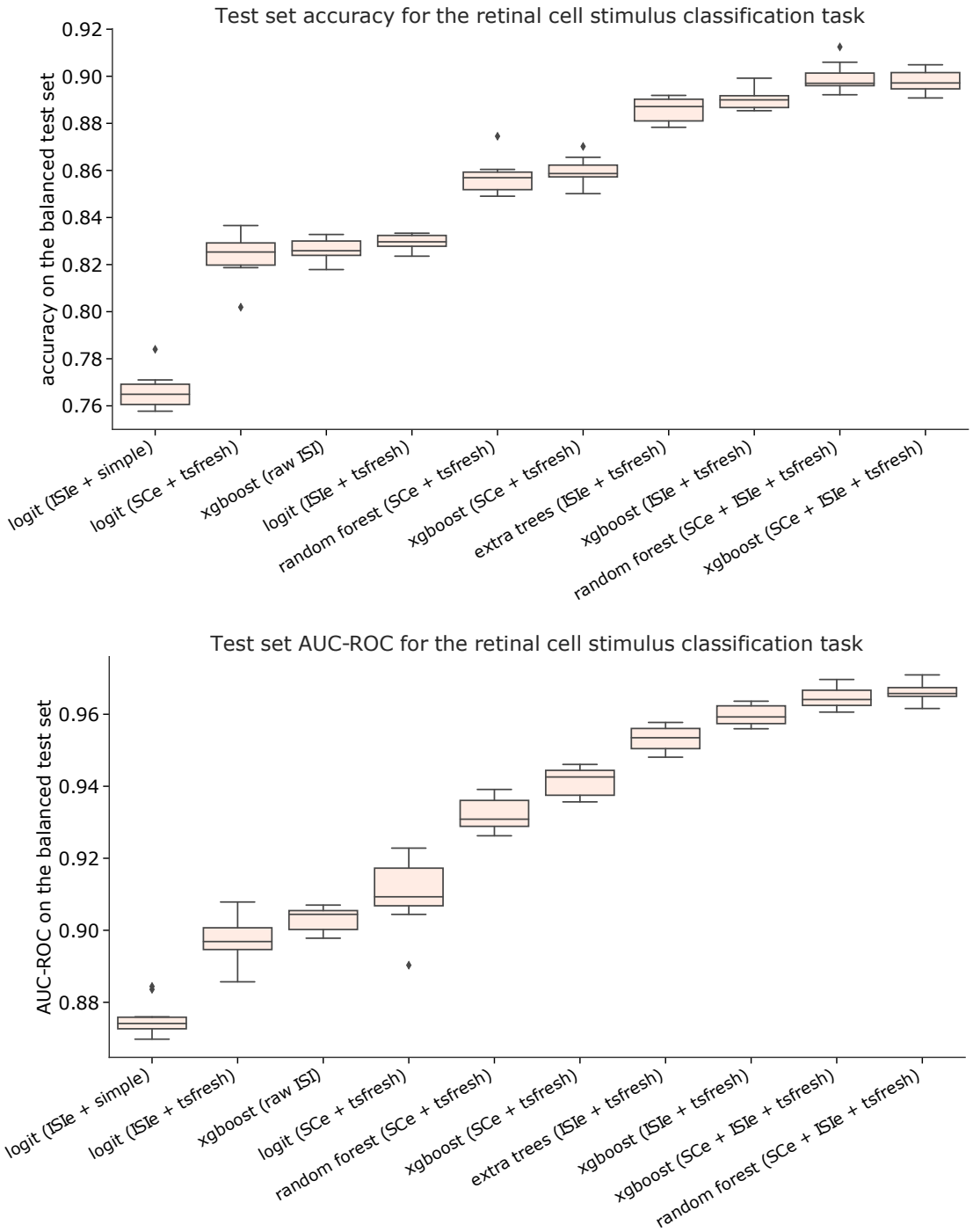

Fig. 3. Spike train classification metric values for the retinal neuron activity dataset on a range of models. The task is defined as binary classification of the stimulus type ("white noise checkerboard" or "randomly moving bar"), with the test set balanced in class distribution (that is, accuracy=0.5 corresponds to chance level). Accuracy is shown on the left and AUC-ROC on the right for the same set of models and train/test dataset splits. The "simple" model tag corresponds to spike trains encoded with 6 basic distribution statistics (representing a simple baseline), the "raw ISI" tag implies that the model has been directly trained on ISI time-series data without encoding. The "tsfresh" tag corresponds to encoding with the full set of time-series features. "ISIe" stands for interspike-interval encoding of the spike train, "SCe" stands for spike-count encoding. "ISIe + SPe" means that feature vectors corresponding to both types of encoding are concatenated.

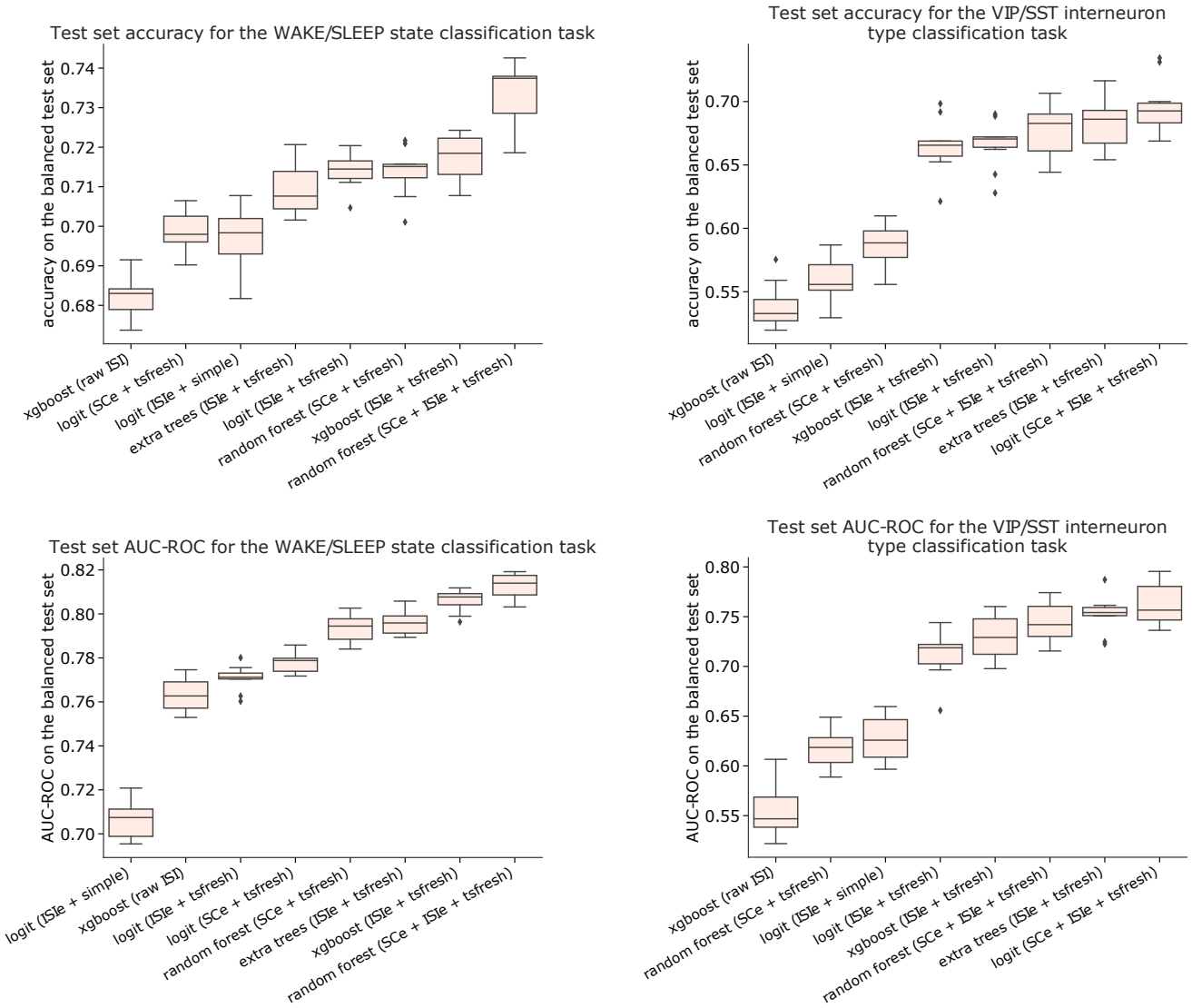

Fig. 4. Spike train classification metric values for the WAKE/SLEEP state prediction dataset (letf) and VIP/SST interneuron type prediction dataset (right) on a range of models. The task is defined as binary classification, with the test set balanced in class distribution for both datasets (that is, accuracy=0.5 corresponds to chance level). Accuracy is shown on the top panes and AUC-ROC on the bottom panes for the same set of models and train/test dataset splits. The "simple" model tag corresponds to spike trains encoded with 6 basic distribution statistics (representing a simple baseline), the "raw ISI" tag implies that the model has been directly trained on ISI time-series data without encoding. The "tsfresh" tag corresponds to encoding with the full set of time-series features. "ISIe" stands for interspike-interval encoding of the spike train, "SCe" stands for spike-count encoding. "ISIe + SPe" means that feature vectors corresponding to both types of encoding are concatenated.

	Retinal stimulus (white noise vs. moving bar)	fcx-1 WAKE/SLEEP state prediction	Allen Cell Types SST/VIP INs
Logistic regression on basic ISI statistics	76.49 ± 0.78	69.83 ± 0.82	55.57 ± 1.63
GBDT on unprocessed ("raw") ISI time-series	82.58 ± 0.47	68.30 ± 0.51	53.27 ± 1.74
Logistic regression on tsfresh-encoded ISI time-series	82.96 ± 0.32	71.44 ± 0.45	67.05 ± 1.89
Random forest on tsfresh-encoded ISI time-series	88.69 ± 0.48	72.31 ± 0.60	67.95 ± 2.21
GBDT on tsfresh-encoded ISI time-series	88.99 ± 0.47	71.84 ± 0.60	66.55 ± 2.11
Random forest on tsfresh-encoded ISI + spike count time-series	89.70 ± 0.62	73.74 ± 0.75	68.27 ± 2.29
1D-CNN on unprocessed ("raw") ISI time-series	61.57 ± 2.33	71.88 ± 1.58	57.13 ± 1.87
InceptionTime on unprocessed ("raw") ISI time-series	73.60 ± 1.34	72.35 ± 1.31	60.08 ± 2.10

TABLE I

Test set accuracy values for different spike train classification models over the three tasks proposed in the benchmark.

collected over the training dataset. Note that since the number of spikes in each data sample is fixed and inter-spike intervals are highly variable, the number of time bins also becomes variable from sample to sample. The *tsfresh* embeddings can nevertheless be computed since they are applicable to variable-length time series.

We then trained classification models on the resulting spike-train vector embeddings. We chose a representative set of models comprising (i) a linear model, namely logistic regression with an l_2 penalty and (ii) various types of tree-based ensembles: a random forest classifier, randomized decision trees (extra trees), a gradient boosted decision tree ensemble. Refer to the

Classification results obtained with the described approach are presented in Figure 3. We report classification accuracy and AUC-ROC values for each model type; note that the testing sets were constructed to be balanced such that the accuracy value of 0.5 corresponded to chance level. We generated 10 random balanced subsamples of the training and testing sets (80% of training/testing data randomly sampled) to estimate accuracy/AUC-ROC distributions presented via boxplots in Figure 3. We were able to reach significant performance levels (> 0.9 accuracy, > 0.96 AUC-ROC) with our best *tsfresh*-based models on the binary stimulus classification task ("randomly moving bar" vs. "white noise checkerboard"). To make better sense of these metric values, we compared our *tsfresh*-based models against two simple baselines: (a) a logistic regression model on ISI-encoded spike-trains represented by 6 basic statistical features – the mean, median, minimum and maximum ISI values, the standard deviation and the absolute energy of the ISI-sequence (the mean of squared ISI values) and (ii) a gradient-boosted decision tree ensemble trained directly on unprocessed ISI-encoded spike trains.

We found that the worst performing model was logistic regression trained on 6 basic statistical features of the ISI distribution, reaching the median test set accuracy of 0.7649 and AUC-ROC of 0.8741. We further demonstrated that one could significantly improve upon those results by training a strong model directly on the unprocessed ISI-encoded samples, with a GBDT model reaching 82.63 median test set accuracy and 0.9030 median AUC-ROC (see Figure 3 and Table I). We found that using *tsfresh* embeddings significantly improves upon that, with a GBDT model reaching lower metric values for spike-count-endoded spike trains (85.97 median accuracy; 94.13 median AUC-ROC) compared to ISI-encoded spike trains (89.05 median accuracy; 95.97 median AUC-ROC for the GBDT classifier). Furthermore, merging *tsfresh* features calculated for both encoding types and training a classifer on the extended set of features increased the test accuracy to 89.77 (with median AUC-ROC of 96.46) for the GBDT classifier.

We also evaluated the performance of state-of-the-art deep learning models in our retinal stimulus classification

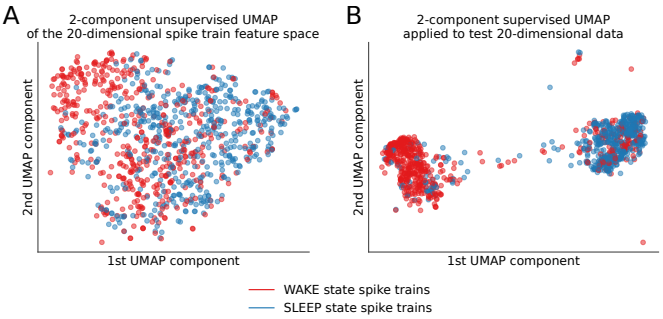

Fig. 5. Spike train feature embeddings for WAKE (points marked red) vs. SLEEP (points marked blue) activity states of the neural circuit. Two-dimensional embeddings of the (20-dimensional) selected-*tsfresh*-feature space using (A) unsupervised UMAP and (B) supervised UMAP embedding algorithms for spike trains corresponding to WAKE vs. SLEEP activity states.

tasks, using implementation from *sktime-dl* [34]. Accuracy results for 1D-CNN and InceptionTime models are shown in Table I. It is clear that models using *tsfresh*-encodings perform on par with deep learning models in terms of classification accuracy.

We performed the same preprocessing steps for the WAKE/SLEEP and VIP/SST datasets as for the retinal stimulus classification dataset. The rolling window of size equal to 200 ISIs and a stride of 100 ISIs together with undersampling for class balance produced a dataset of 19786 training examples and 5540 testing examples (according to a 70/30 train/test split) for the WAKE/SLEEP state dataset. A rolling window of size equal to 50 ISIs and a stride of 20 ISIs with undersampling was applied to the VIP/SST dataset, resulting in 1630 training and 872 testing examples.

We observed similar trends both for the WAKE/SLEEP state and VIP/SST interneuron classification tasks, shown in Figure 4 and Table I. The best performing models were found to be *tsfresh*-based ones using the combined ISI and spike count encodings of the undelying spike trains.

The set of most discriminative spike train features: Being able to estimate feature importance ranks from trained decision tree ensembles allows us to detect the most discriminating features of ISI time-series for a particular classification problem. In order to select the important groups of discriminative features, we applied the following procedure to the full set of *tsfresh* features: first, we trained 10 logistic regression models with l_1 regularization penalty (with different random seeds, $C = 0.01$) on the full feature set; we then selected the features which had non-zero values of corresponding weight coefficients in all trained models. After that, we identified highly correlated pairs of features ($|R| > 0.98$), which represent almost equivalent quantities, and removed one randomly selected feature out of each such pair. Further, we trained several random forest classifier models (with different random seeds) and calculated the aggregated feature importance ranks across models to select groups of features relevant

for the classification task. After merging important feature sets corresponding to all of the three classification tasks, the following groups of *tsfresh* features are selected:

- *median*, *kurtosis*, *quantile_q* – simple statistics of the ISI value distribution in the series like the median ISI value, q quantiles and kurtosis of the ISI value distribution
- *change_quantiles* – this feature is calculated by fixing a corridor of the time series values (defined by lower and higher quantile bounds, q_l and q_h, which are hyperparameters), then calculating a set of consecutive change values in the series (differencing) and then applying an aggregation function (mean or variance). Another boolean hyperparameter *is_abs* determines whether absolute change values should be taken or not.
- *fft_coefficient* – absolute values of the DFT expansion coefficients.
- *agg_linear_trend* – features from linear least-squares regression (standard error in particular) for the values of the time series that were aggregated over chunks of a certain size (with different aggregation functions like min, max, mean and variance). Chunk sizes vary from 5 to 50 points in the series.

To visualize class separation for the WAKE vs. SLEEP state spike trains from the fcx-1 dataset as point clouds in two dimensions, we took the top-20 importance *tsfresh* features identified during the above feature selection procedure. We then used dimensionality reduction techniques on this reduced 20-dimensional dataset to visualize the structure of the data with respect to excitatory/inhibitory labels of the series. Results are shown in Fig. 5 for two Uniform Manifold Approximation and Projection, UMAP [35] low-dimension embedding algorithms. We also applied other methods such as PCA and t-SNE (t-distributed Stochastic Neighbor Embedding) which gave essentially the same results (not shown). In all cases, classes cannot be linearly separated in two-dimensional embedding spaces, however, there is a separation of large fraction of the points of the excitatory-cell and inhibitory-cell classes.

We conclude that the hand-crafted feature engineering approach combined with strong tree-based learning models sets a strong baseline for spike train classification for all of the tree studied tasks.

Discussion

In this work, we have introduced a diverse neuronal spike train classification benchmark to evaluate neural decoding algorithms. The benchmark consists of several single-neuron spike train prediction tasks spanning stimulus type prediction, neuron type identification and animal behavioural state prediction. We have demonstrated that individual neuronal spike trains contain information related to the global state of the underlying neural circuit and this information can be decoded if appropriate time-series learning models are used. Extensive experiments on several datasets that we have conducted imply that not only ISI value distribution is important for global state identification but also the temporal information contained in the spike trains, that is, features related to the exact sequences of interspike intervals in neural firing. We have identified groups of features highly informative for neural decoding tasks and established that this feature encoding combined with strong supervised learning algorithms such as gradient boosted tree ensembles establishes a strong baseline on the proposed benchmark that performs on par with state-of-the-art deep learning approaches. It was shown that significantly large accuracy values can be obtained on all of the proposed tasks using the hand-crafted feature encoding approach on single-neuron spike train chunks containing as low as 50 interpike intervals. We suggest that accuracy values can further be improved by model ensembling and test-time data augmentation. We propose that neural decoding models be evaluated on diverse and challenging tasks including the proposed benchmark in order to establish a sensible model performance ranking similarly to what is done for computer vision and natural language understanding problems. We believe that this would drive further development of highly accurate neural decoding/neural activity mining approaches enabling their application in precision-critical tasks such as identifying pathological disease-related firing activity patterns in the brain.

References

[1] M. Pachitariu, C. Stringer, S. Schröder, M. Dipoppa, L. F. Rossi, M. Carandini, and K. D. Harris, "Suite2p: beyond 10,000 neurons with standard two-photon microscopy," *Biorxiv*, p. 061507, 2016.

[2] D. Tsai, E. John, T. Chari, R. Yuste, and K. Shepard, "High-channel-count, high-density microelectrode array for closed-loop investigation of neuronal networks," in *Engineering in Medicine and Biology Society (EMBC), 2015 37th Annual International Conference of the IEEE*. IEEE, 2015, pp. 7510–7513.

[3] N. A. Steinmetz, C. Koch, K. D. Harris, and M. Carandini, "Challenges and opportunities for large-scale electrophysiology with neuropixels probes," *Current opinion in neurobiology*, vol. 50, pp. 92–100, 2018.

[4] J. L. Teeters and F. T. Sommer, "Crcns. org: a repository of high-quality data sets and tools for computational neuroscience," *BMC Neuroscience*, vol. 10, no. S1, p. S6, 2009.

[5] J. I. Glaser, R. H. Chowdhury, M. G. Perich, L. E. Miller, and K. P. Kording, "Machine learning for neural decoding," *arXiv preprint arXiv:1708.00909*, 2017.

[6] A. Bagnall, J. Lines, A. Bostrom, J. Large, and E. Keogh, "The great time series classification bake off: a review and experimental evaluation of recent algorithmic advances," *Data Mining and Knowledge Discovery*, vol. 31, no. 3, pp. 606–660, 2017.

[7] H. I. Fawaz, G. Forestier, J. Weber, L. Idoumghar, and P.-A. Muller, "Deep learning for time series classification: a review," *Data Mining and Knowledge Discovery*, vol. 33, no. 4, pp. 917–963, 2019.

[8] A. S. Benjamin, H. L. Fernandes, T. Tomlinson, P. Ramkumar, C. VerSteeg, R. H. Chowdhury, L. E. Miller, and K. P. Kording, "Modern machine learning as a benchmark for fitting neural responses," *Frontiers in computational neuroscience*, vol. 12, 2018.

[9] T. Tezuka, "Multineuron spike train analysis with r-convolution linear combination kernel," *Neural Networks*, vol. 102, pp. 67–77, 2018.

[10] M. D. Humphries, "Spike-train communities: finding groups of similar spike trains," *Journal of Neuroscience*, vol. 31, no. 6, pp. 2321–2336, 2011.

[11] M. v. Rossum, "A novel spike distance," *Neural computation*, vol. 13, no. 4, pp. 751–763, 2001.

[12] J. D. Victor and K. P. Purpura, "Metric-space analysis of spike trains: theory, algorithms and application," *Network: computation in neural systems*, vol. 8, no. 2, pp. 127–164, 1997.

[13] M. Mulansky and T. Kreuz, "Pyspike—a python library for analyzing spike train synchrony," *SoftwareX*, vol. 5, pp. 183–189, 2016.

[14] T. Tezuka, "Spike train pattern discovery using interval structure alignment," in *International Conference on Neural Information Processing*. Springer, 2015, pp. 241–249.

[15] J. Jouty, G. Hilgen, E. Sernagor, and M. Hennig, "Non-parametric physiological classification of retinal ganglion cells," *bioRxiv*, p. 407635, 2018.

[16] P. Charlesworth, E. Cotterill, A. Morton, S. G. Grant, and S. J. Eglen, "Quantitative differences in developmental profiles of spontaneous activity in cortical and hippocampal cultures," *Neural development*, vol. 10, no. 1, p. 1, 2015.

[17] M. Li, F. Zhao, J. Lee, D. Wang, H. Kuang, and J. Z. Tsien, "Computational classification approach to profile neuron subtypes from brain activity mapping data," *Scientific reports*, vol. 5, p. 12474, 2015.

[18] X. Jia, J. Siegle, C. Bennett, S. Gale, D. Denman, C. Koch, and S. Olsen, "High-density extracellular probes reveal dendritic backpropagation and facilitate neuron classification," *bioRxiv*, p. 376863, 2018.

[19] B. D. Fulcher and N. S. Jones, "hctsa: A computational framework for automated time-series phenotyping using massive feature extraction," *Cell systems*, vol. 5, no. 5, pp. 527–531, 2017.

[20] M. Christ, N. Braun, J. Neuffer, and A. W. Kempa-Liehr, "Time series feature extraction on basis of scalable hypothesis tests (tsfresh–a python package)," *Neurocomputing*, 2018.

[21] J. H. Friedman, "Greedy function approximation: a gradient boosting machine," *Annals of statistics*, pp. 1189–1232, 2001.

[22] Y. LeCun, Y. Bengio, and G. Hinton, "Deep learning," *nature*, vol. 521, no. 7553, p. 436, 2015.

[23] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural computation*, vol. 9, no. 8, pp. 1735–1780, 1997.

[24] Y. LeCun, B. Boser, J. S. Denker, D. Henderson, R. E. Howard, W. Hubbard, and L. D. Jackel, "Backpropagation applied to handwritten zip code recognition," *Neural computation*, vol. 1, no. 4, pp. 541–551, 1989.

[25] F. Karim, S. Majumdar, H. Darabi, and S. Chen, "Lstm fully convolutional networks for time series classification," *IEEE Access*, vol. 6, pp. 1662–1669, 2018.

[26] H. I. Fawaz, B. Lucas, G. Forestier, C. Pelletier, D. F. Schmidt, J. Weber, G. I. Webb, L. Idoumghar, P.-A. Muller, and F. Petitjean, "Inceptiontime: Finding alexnet for time series classification," *Data Mining and Knowledge Discovery*, vol. 34, no. 6, pp. 1936–1962, 2020.

[27] H. A. Dau, A. Bagnall, K. Kamgar, C.-C. M. Yeh, Y. Zhu, S. Gharghabi, C. A. Ratanamahatana, and E. Keogh, "The ucr time series archive," *IEEE/CAA Journal of Automatica Sinica*, vol. 6, no. 6, pp. 1293–1305, 2019.

[28] J. S. Prentice, O. Marre, M. L. Ioffe, A. R. Loback, G. Tkačik, and M. J. Berry, "Error-robust modes of the retinal population code," *PLoS computational biology*, vol. 12, no. 11, p. e1005148, 2016.

[29] G. P. J. S. I. M. L. B. I. M. J. M. O. Loback, Adrianna R.; Tkačik, "Multi-electrode retinal ganglion cell population spiking data," *Dryad. Dataset.*, 2016.

[30] B. Watson, D. Levenstein, J. Greene, J. Gelinas, and G. Buzsaki, "Multi-unit spiking activity recorded from rat frontal cortex (brain regions mpfc, ofc, acc, and m2) during wake-sleep episode wherein at least 7 minutes of wake are followed by 20 minutes of sleep. crcns.org," 2016.

[31] B. O. Watson, D. Levenstein, J. P. Greene, J. N. Gelinas, and G. Buzsáki, "Network homeostasis and state dynamics of neocortical sleep," *Neuron*, vol. 90, no. 4, pp. 839–852, 2016.

[32] "Allen cell types dataset." [Online]. Available: https://celltypes.brain-map.org/

[33] B. Rudy, G. Fishell, S. Lee, and J. Hjerling-Leffler, "Three groups of interneurons account for nearly 100% of neocortical gabaergic neurons," *Developmental neurobiology*, vol. 71, no. 1, pp. 45–61, 2011.

[34] M. Löning, A. Bagnall, S. Ganesh, V. Kazakov, J. Lines, and F. J. Király, "sktime: A unified interface for machine learning with time series," *arXiv preprint arXiv:1909.07872*, 2019.

[35] L. McInnes, J. Healy, N. Saul, and L. Großberger, "Umap: Uniform manifold approximation and projection," *The Journal of Open Source Software*, vol. 3, no. 29, p. 861, 2018.