HIGH SCHOOL EDITION
Journal of Student Research

Volume 12 Issue 4 (2023)

Efficient Detection of Ion Channel Switching through Rapid Random Forest Models

Derek Cho

Tenafly High School

ABSTRACT

The proper function of ion channels is fundamental to many of our physiological processes, such as neural signal transmission, muscle contraction, and insulin secretion. As a result of genetic mutations and environmental factors, however, the dysfunction of ion channels can impede crucial cellular processes, resulting in channelopathies, such as cystic fibrosis and hypoglycemia, which may lead to highly impaired respiratory, cardiovascular, and muscular function. To detect ion channel dysfunction, a GPU-accelerated random forest model with an F1-score of 0.935 was developed to efficiently analyze electrophysiological time series data for the improper opening and closing of ion channels. The random forest model can run 200x faster than real-time 10 kHz electrophysiological data collection, holding immense potential for the development of reactive clinical measures for ion channel dysfunction and the development of novel therapies for various ion channel-based musculoskeletal disorders.

Introduction

Ion channels are pore-forming proteins found in cell membranes that allow for the transport of ions across hydrophobic cell membranes. The opening and closing of ion channels, known as ion channel events, can occur as a response to stimuli such as biomolecule binding, voltage changes, or mechanical stresses. The opening of ion channels, allowing the flow of ions, creates an electric current that serves crucial roles in neural signal transmission, muscle contraction, insulin secretion, and further physiological processes. However, ion channel dysfunction, caused by genetic mutations and environmental influences, that disrupt regular ion flow can impede crucial cellular processes and result in disorders known as channelopathies, including cystic fibrosis and hypoglycemia (Kasianowicz, 2012).

This paper aims to demonstrate the efficacy of ensemble machine learning algorithms, specifically random forest models, for the automatic detection of ion channel events and accurately determine the number of open ion channels based on electrophysiological time series data, which involves the measurement of the electrical activity of biological cells. Developing a machine learning model that can automatically detect the improper opening and closing of ion channels can provide critical, submicroscopic insights on the mechanism of channelopathies, enabling the development of novel treatment strategies and prompting rapid reactive measures, as well as provide a further understanding of the cellular mechanisms that govern human and animal biomechanics, facilitating the development of therapies for various musculoskeletal disorders.

Methods

Data Preprocessing

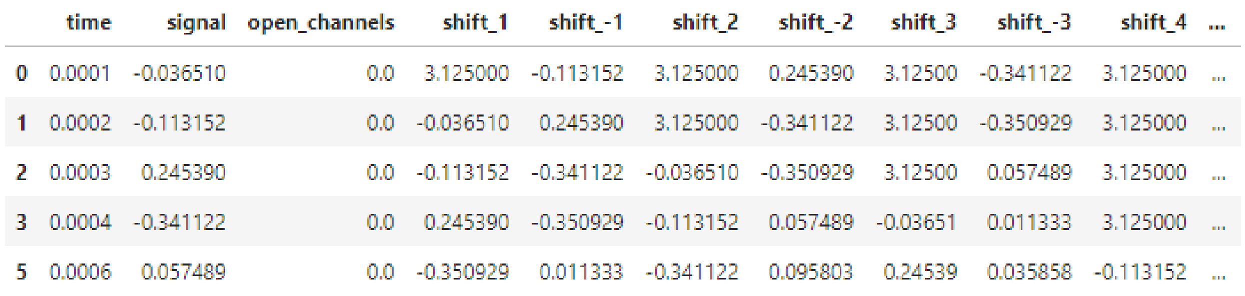
The electrophysiological time series data used to train and test the machine learning model was originally taken from a Kaggle competition, “University of Liverpool - Ion Switching.” The data was collected in 50-second long batches of 10 kHz signal measurements, which were combined to create a training dataset ranging from 0 to 500 seconds and a testing dataset ranging from 500 to 700 seconds, leading to an approximately 70/30 train-test split. Since the data was recorded in 10 kHz frequency, the training dataset contained 5,000,000 rows each measuring in 0.0001-second intervals while the testing dataset contained 2,000,000 rows. The training dataset contained three columns: time, signal, and open_channels, where signal was described by the electrical current in picoamps and open_channels was described by discrete integer values known as "classes," describing the number of open ion channels, ranging from 0 to 11. The testing dataset simply contained two columns (time and signal) as it was the model's task to predict open_channels for this dataset (*University of Liverpool - Ion Switching*, 2020).

Through Kaggle's collaborative notebooks, a cleaned dataset was obtained for this electrophysiological data. This cleaned dataset removed data drifts in both the train and test data, scaled the signal data by 1.25x to align with the open_channels data, and discarded noisy outliers that were indicated by anomalous spikes in signal data (*Remove Trends Giba - Explained*, 2020).

Feature Engineering

Using this cleaned dataset, further features were generated by taking the signal data and mapping it to new columns while shifting the indexes by 1 to 15 and -1 to -15, meaning that the original signal data at 0.0001 seconds would match the data in shift_1, a new column, at 0.0002 seconds, while the original signal data at 0.0002s would match the data in shift_-1 at 0.0001 seconds. This would create 30 new columns in the training and testing dataset, ranging from shift_-1 to shift_-15 and shift_1 to shift_15. Signal data for a column created with a shift would have data gaps at the beginning or end of the column, depending on whether it was a negative or positive shift. To address this missing data from shifting, the missing values were replaced with 3.125 picoamps, the mean of signal values throughout the 500 seconds of data.

Creating new data based on shifts of the original signal data would allow the machine learning model to concurrently analyze and compare signal data at ± 0.0015 seconds while training, looking for sudden yet sustained disparities in the signal data between certain time periods and the next 15 data points, which would indicate that an ion channel had just opened or closed.



	time	signal	open_channels	shift_1	shift_-1	shift_2	shift_-2	shift_3	shift_-3	shift_4	...
0	0.0001	-0.036510	0.0	3.125000	-0.113152	3.125000	0.245390	3.12500	-0.341122	3.125000	...
1	0.0002	-0.113152	0.0	-0.036510	0.245390	3.125000	-0.341122	3.12500	-0.350929	3.125000	...
2	0.0003	0.245390	0.0	-0.113152	-0.341122	-0.036510	-0.350929	3.12500	0.057489	3.125000	...
3	0.0004	-0.341122	0.0	0.245390	-0.350929	-0.113152	0.057489	-0.03651	0.011333	3.125000	...
5	0.0006	0.057489	0.0	-0.350929	0.011333	-0.341122	0.095803	0.24539	0.035858	-0.113152	...

Figure 1. Preview of the first 5 rows and 10 columns of the training dataset after data preprocessing and feature engineering.

Random Forest Model

The code for this project was written entirely through Kaggle notebooks. Random forest models make predictions on test data through the aggregation of predictions made by decision trees, which are individually trained on random subsets of data. When inputted with data, these individual decision trees are guided by nodes, indicated by circles in Figure 2, to undergo a decision-making process and determine the final output of the tree. The final outputs of all the decision trees are then ensembled by the random forest model into a final prediction, which is the "class" that was predicted by the majority of the decision trees (*What is Random Forest?*, n.d.).

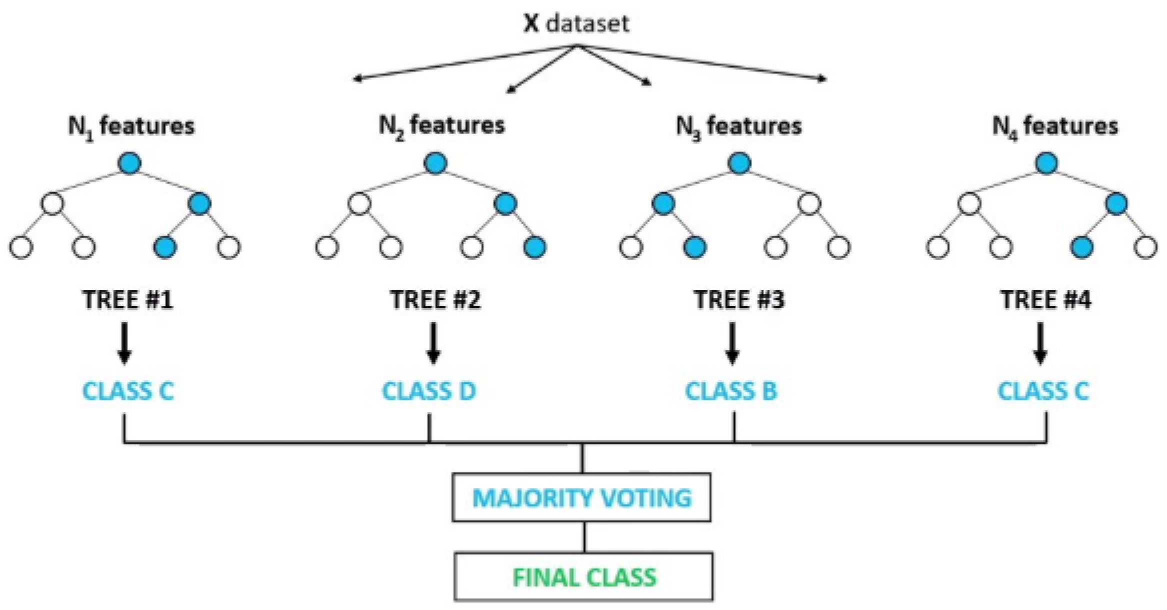


Figure 2. Visualization of sample Random Forest Model prediction logic (David, 2020).

Random forest models were chosen as the approach to this classification problem since they predict "classes" based on the input of multiple decision trees, which makes the model more robust to noisy data, which is often prevalent in the practical measurement of electrophysiological data.

Originally, sklearn's random forest models were employed on this classification problem; however, since sklearn uses CPU processing, minimal training progress was made against the immense size of the training dataset (5,000,000 rows x 33 columns). Training on just 10,000 rows of data took over 1 minute and training on 100,000 rows of data took over 10 minutes, evidently unfavorable to the objective of rapid real-time detection of ion channel events.

Thus, to achieve faster training times, RAPIDS cuML was implemented into the Kaggle environment, which allows machine learning algorithms to run on GPUs rather than solely CPUs, immensely accelerating machine learning tasks on large datasets, especially for models that involve many operations such as random forest models, which have to consider many nodes and trees (*Accelerating Random Forests*, 2019).

Two models with minimal hyperparameter tuning were implemented for this classification problem, cuML's RandomForestClassifier model and cuML's RandomForestRegressor model, both with the following parameters: n_estimators = 35, split_criterion = "mse," bootstrap = True, and max_depth = 18, while the other parameters were set to default. The parameter n_estimators determines the total number of decision trees, split_criterion sets the goal of the nodes to minimize the mean squared error of the decision tree's prediction, bootstrap as True allows each decision tree to look at a different, random subsection of the data, introducing diversity into the learning process, and max_depth determines the maximum number of node layers that are explored before making a final prediction within its decision tree.

The training process on both models was performed with five sklearn StratifiedKFolds, which splits the training dataset into 5 sections with similar class distributions. The model then looks at different combinations of four training sections and one validation section at every fold, essentially training and validating on the entire dataset. Through this technique, the model is exposed to diverse data while training and validating, which consequently avoids model overfitting as the validation data changes with every fold. Since there was a significant class imbalance in the dataset, the number of k-folds was capped at five as having too many would lead to sparse minority classes in each of the folds, leaving the model weak in predicting the minority classes (*Stratified K Fold*, 2023).

Results

Model Validation Metric

In both models, RandomForestRegressor and RandomForestClassifier, the model accuracy was determined using its macro F1-score, which was calculated using the built-in F1_score function in the sklearn library, taking the unweighted average of F1-scores for each "class" of open-channels data, delineated by the following equation:

$$F_1 = 2 \cdot \frac{precision \cdot recall}{precision + recall} = \frac{2TP}{2TP + FP + FN}$$

where precision is the proportion of positive identifications that were correct, recall is the proportion of actual positives that were identified correctly, TP (true positive) is the number of positive identifications that were correct, FP (false positive) is the number of positive identifications that were incorrect, and FN (false negative) is the number of negative identifications that were incorrect. An F1-score of 1.0 would be a perfectly accurate model (Hicks et al., 2022).

Interestingly, implementing the RandomForestClassifier model, intended to be used when predicting discrete "class" values, which would align with the nature of this problem as open_channels are discrete integer values, resulted in poor performance with an F1-score of 0.6157, most likely due to the lack of hyperparameter tuning. However, implementing the RandomForestRegressor model, which outputs decimal values that had to be rounded to the nearest integer to match the discrete nature of open_channels, resulted in significantly improved performance, with an F1-score of 0.935 and completing training in under 13 minutes.

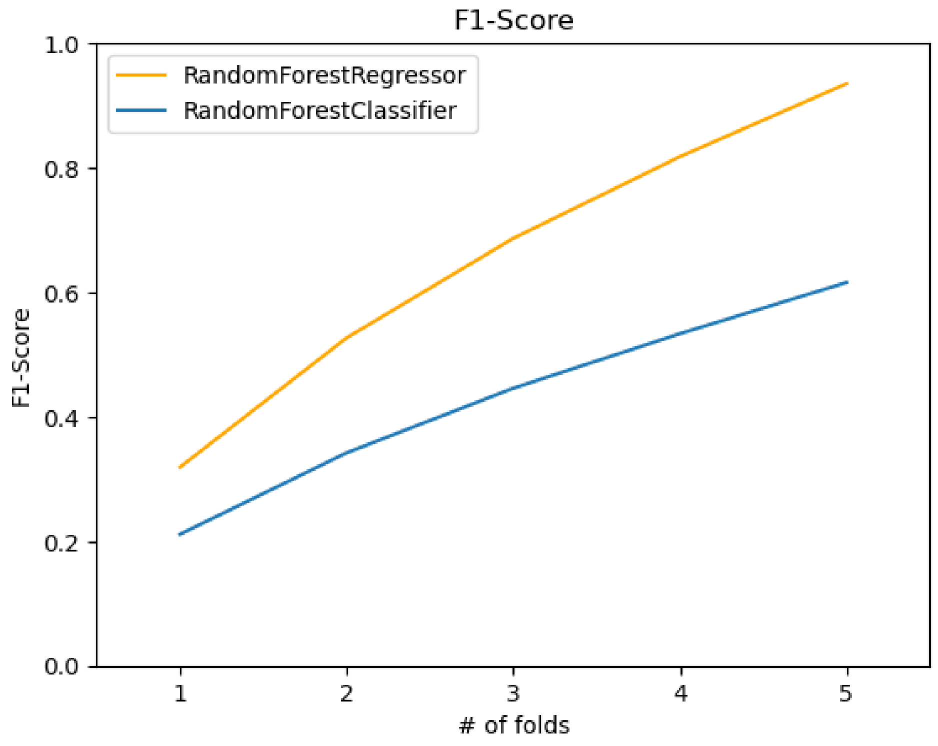


Figure 3. Progress of the RandomForest model's F1-Score as the models underwent n number of k-folds.

Upon successful training of the RandomForestRegressor model, the model was then tested on the testing dataset (2,000,000 rows x 32 columns), which contained 200 seconds of 10 kHz electrophysiological data. Making open_channel predictions for the entire testing dataset took under a second while achieving a testing F1-score of 0.93777.

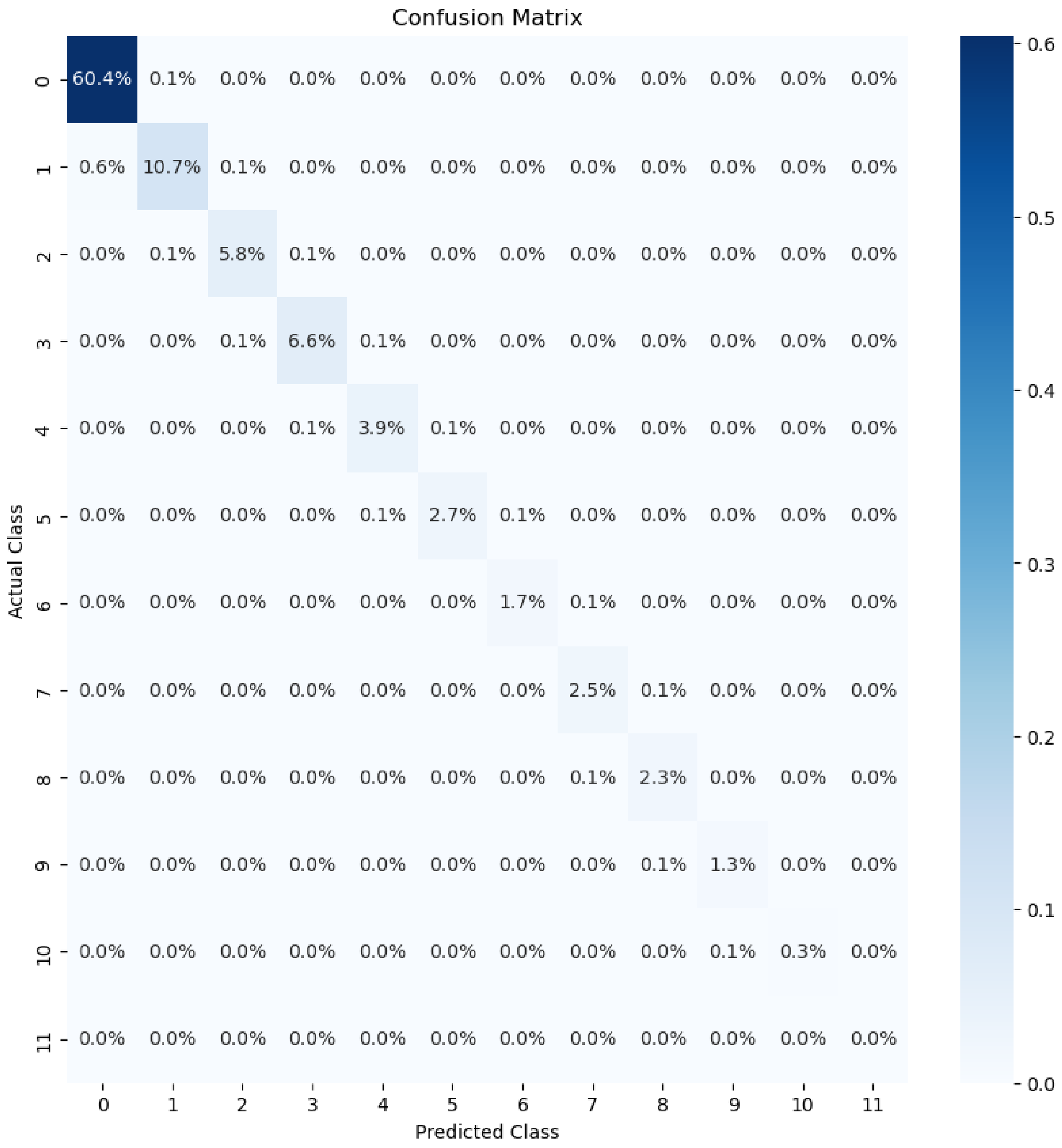


Figure 4. Confusion matrix of the RandomForestRegressor model predictions on the test dataset where "class" refers to open_channels. Despite the evident class imbalance in the dataset, the model performs well in predicting all classes, with its predictions rarely, if ever, deviating from the actual values of open_channel by more than one integer.

Discussion

This study introduced an ensemble learning approach for the automatic detection of ion channel events within noisy electrophysiological time series data. The principal model implemented in this study, the RandomForestRegressor model, was able to accurately determine the number of open ion channels based on 10 kHz electrophysiological data at 200x faster than real-time data collection.

These rapid and accurate results demonstrate the viability of this machine learning detection algorithm in real-time clinical applications. The ability to promptly identify improper ion channel events allows for the implementation of rapid reactive measures, such as delivering targeted bio-inhibitors or applying voltage shocks to, in real-time, close a specific ion channel and prevent channelopathies, and its adverse effects, from developing. Furthermore, this machine learning model extends to the further study of the cellular mechanisms that govern human and animal biomechanics. Understanding the submicroscopic processes that dictate our biomechanics can facilitate the development of novel therapies for various ion channel-based musculoskeletal disorders by identifying vital targets for repair.

Acknowledgments

I would like to thank the University of Liverpool for providing this public dataset, as well as other Kaggle users who have contributed ideas on potential approach strategies and techniques to this machine learning problem on the discussion forums.

References

Accelerating Random Forests up to 45x using cuML. (2019). Retrieved August 12, 2023, from https://medium.com/rapids-ai/accelerating-random-forests-up-to-45x-using-cuml-dfb782a31bea

David, D. (2020, August 6). *Random Forest Classifier Tutorial: How to Use Tree-Based Algorithms for Machine Learning*. freeCodeCamp. Retrieved August 12, 2023, from https://www.freecodecamp.org/news/how-to-use-the-tree-based-algorithm-for-machine-learning/

Hicks, S. A., Strümke, I., Thambawita, V., Hammou, M., Riegler, M. A., Halvorsen, P., & Parasa, S. (2022). On evaluation metrics for medical applications of artificial intelligence. *Scientific Reports*, 12(1), 5979. https://doi.org/10.1038/s41598-022-09954-8

Kasianowicz, J. (2012). Introduction to Ion Channels and Disease. *Chemical Reviews, 112(12)*, 6215-6452. https://doi.org/10.1021/cr300444k

Remove Trends Giba - Explained. (2020). Retrieved from https://www.kaggle.com/code/titericz/remove-trends-giba-explained/notebook

Stratified K Fold Cross Validation. (2023, January 10). Geeksforgeeks. Retrieved August 15, 2023, from https://www.geeksforgeeks.org/stratified-k-fold-cross-validation/

University of Liverpool - Ion Switching. (2020, February 24). Retrieved from https://www.kaggle.com/competitions/liverpool-ion-switching

What is random forest? (n.d.). IBM Corp. Retrieved August 12, 2023, from https://www.ibm.com/topics/random-forest