ON THE UTILITY OF MACHINE LEARNING APPROACHES TOWARDS AUTOMATING MICROSATELLITE ALLELE SCORING

Demavivas IF[1], Abrasaldo GA[2], Fabilloren SR, and Kim KE[3]

[1] Department of Physical Sciences and Mathematics, University of the Philippines Manila, Manila 1000

[2] Division of Natural Sciences and Mathematics, University of the Philippines Visayas Tacloban College, Tacloban City 6500

[3] Philippine Genome Center, University of the Philippines System, Quezon City 1101

ABSTRACT

Microsatellite fragment analysis is the first step in genetic research due to its low cost and accuracy in studies such as population diversity, mutation detection, linkage, and quantitative trait loci (QTL) mapping. The analysis is difficult to automate since it requires human intervention to effectively sift and select peaks through the noise. Some commercial and free software are deployed for the problem; however, most lack the accuracy good enough for automation and quick high-throughput applicability. *AuSOME-ML* is an open source Python script designed as a proof-of-concept for automatic detection and scoring of DNA fragment lengths using machine learning. It is a freely-available platform-independent software written in Python. Sandfish (*Holothuria scabra*) genotypes based on microsatellite markers is used to highlight the capabilities of *AuSOME-ML*. Such data is generated by Applied Biosystems® (ABI) sequencing machines exported as FSA file formats. In this study, 411 FSA files containing microsatellite data from Hsc-40 loci of *Holothuria scabra* was used to obtain approximately 3,000 regions containing peaks as the training data. The number of peaks in these regions, their total area of region, and the corresponding length (in bp) were used as the features to label the data as either noise (0) or true peak (1). The data were then trained with a number of classifiers, such as Random Forest using Scikit-learn packages in Python. *AuSOME-ML* is an open-source tool for genetic analysis. The package produces comparable results to other popular software with the added capacity of automation for high-throughput experiments aided by machine learning.

Introduction

Microsatellites also known as Single Sequence Repeats (SSR) are non-coding, repeating regions unique in the DNA of every organism (Brooker 1999). This uniqueness, due to replication and recombination errors, along with its high accuracy and low relative cost makes fragment analysis an important step in these studies (Inghelandt *et al.,* 2010). This method is widely utilized in fields such as forensics, population studies and conservation genetics. Fragment analysis includes PCR-amplified fragments with fluorescent dyes. These fragments are then separated with capillary electrophoresis where a high voltage charge induces the migrations of the fragments along a capillary tube. The fragments, with smaller fragments migrating faster, move along the tube and pass through a scanner where their fluorescence intensity is measured. The fluorescence intensity, along with the time difference in between scans allows the measurement of the fragment lengths. Using ABI instrumentation, these data points are recorded as .FSA files (Applied Biosystems 2006). Individual FSA files are analysed manually and meticulously using licensed and commercial software. Although some of these commercial software help enhance the analysis by suggesting possible correct peaks, it does not allow for full automation. Here we present a proof-of-concept demonstration of applying current machine learning models in automating the peak prediction in FSA files.

Scope and Limitations: This has only been tested with a single loci (Hsc 40) with a single fluorescent dye (6FAM) and GeneScan™ LIZ-500 ladder dye on Sandfish (*Holothuria scabra)* microsatellite data obtained from the population study on Philippine sandfish (Ravago-Gotanco and Kim 2019). This has been conducted using the versions Python 3.6.8, Biopython 1.74, Scikit-learn 0.21.3, Numpy 1.17.2, Pandas 0.25.1, matplotlib 3.1.1.

Review of Literature

Microsatellites

Microsatellites are non-coding, repeating bases in both eukaryotic and prokaryotic DNA. Also known as simple sequence repeats (SSR) or short tandem repeats (STR), It has a repeat unit size of about 2-6 base pairs, repeating many times (Oliveira et al 2006). They are unique to every organism due errors and mismatches in DNA replication and recombination (Schlötterer, C., & Pemberton 1998). In humans, an average mutation frequency of 0.01% is observed and is observed to be generally constant in most species (Brinkmann *et al.,* 1998). This generally predictable mutation rate is used in studies in forensics, population studies and conservation genetics (Remya et al., 2010).

Current population studies involving microsatellite fragment analysis generally follow a similar methodology (Gauffre et al 2007; Lefèvre et al 2012; Minter et al 2014; Ravago-Gotanco and Kim 2019). Microsatellites are genotyped using pre-developed markers as reference. They are then amplified through Polymerase Chain Reaction (PCR). Forward primers are then labelled using fluorescent dyes. These amplicons are pooled along with a fluorescent size standard and are then usually sequenced in ABI Sequencing Machines (Applied Biosystems 2016).

Covarrubias-Pazaran et al (2016) did a comparison on their R package, *Fragman* on other fragment analysis software and found out that most rely on commercial and licensed software such as GeneMarker®, Peak Studio, Genomatic, Biostrings, MsatAllele. Most of these fragment analysis software are semi-automatic and still need manual intervention.

Machine Learning

The exponential increase in the amount of biological data shows the need for more powerful data analysis tools to extract useful information.

Methodology

Data collection

Microsatellite data was obtained from the population genetic structure study of *Holuthuria scabra* in the Philippine Archipelago (Ravago-Gotanco and Kim 2019). ABI 3730xl DNA Analyzer was used to sequence the fragments and export in '.FSA' format. Out of all 477 FSA files, only the microsatellite data from one loci, Hsc 40, were used in this study. Data from the channels were extracted using the SeqIO interface from the open-source library Biopython (Cock *et al.,* 2009). Figure 1 shows the data channels in one FSA file which includes Channels 1, 2, 3, 4, and 105 corresponding to 6-FAM, VIC, NED, PET, and 500 LIZ dyes respectively with the channel 1 (6-FAM dye) containing the data for the Hsc 40 loci and channel 105 (500 LIZ dye) for the size standard.

Ladder sizing

The ladder or size standard present in each FSA file is used to approximate the corresponding length in base pairs (bp) of a peak call. Each peak in the size standard corresponds to an allele of known length. For the 500 LIZ dye, from left to right, these peaks have known lengths of 35, 50, 75, 100, 139, 150, 160, 200, 250, 300, 340, 350, 400, 450, 490, and 500 bp respectively. From this information, when a peak call exists between two peaks in the size standard, the corresponding length of the peak call can be approximated by interpolation.

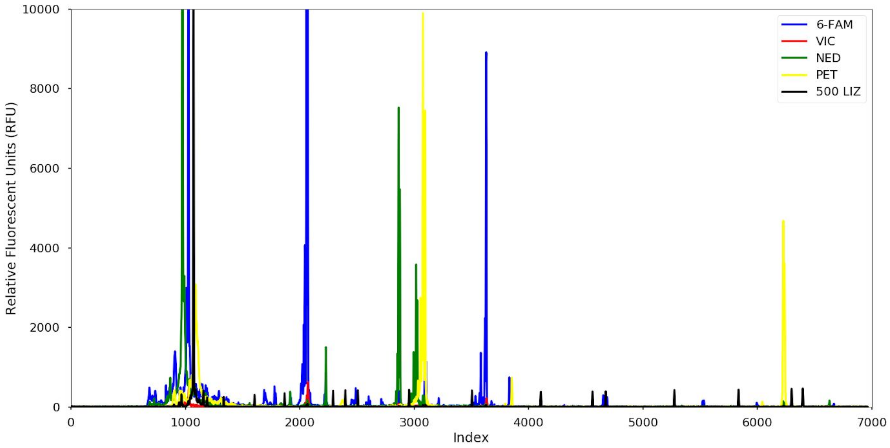

Figure 1. Contents of an FSA file. Aside from the metadata generated during sequencing, the FSA file also contains the raw sequencing data and peaks in Relative Fluorescence Units (RFU).

Data pre-processing and feature extraction

For each FSA file, after ladder sizing, the peaks in channel 1 (6-FAM dye) were detected using the *findpeaks* Python package from Janko Slavic (Slavič 2015). Only peaks that exists between the 35 bp and 500 bp allele of the size standard were selected to eliminate unnecessary peaks outside the range of the ladder. For each of the selected peaks, a size range was created that contains the peak from which 4 features were extracted; area of the region, number of peaks in the region, corresponding length in bp of the selected peak, and the height (in relative fluorescence units) of the peak as shown in Figure 2. Each of these regions were then labelled as either a true peak corresponding to an allele call in Hsc 40 loci (1) or noise (0) in reference to the data obtained from the *H. scabra* population study.

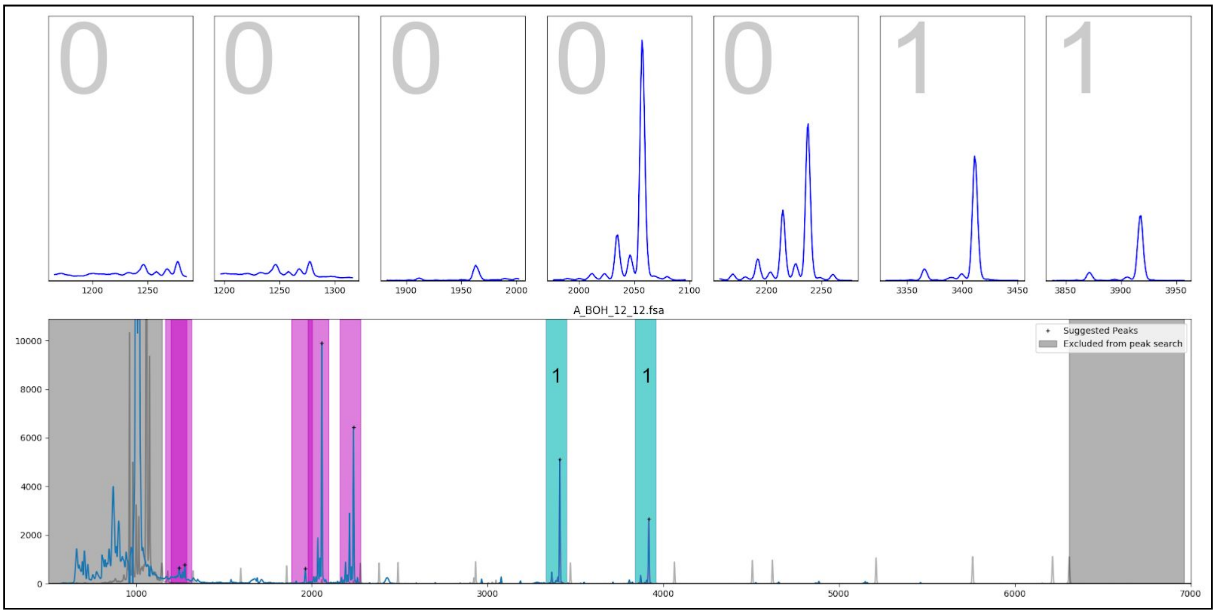

Figure 2. Peak detection in AuSOME-ML. Peaks in areas outside the matched ladder (shaded in gray) are excluded from the peak search. Detected peaks are displayed above the graph, while their position is highlighted below in the fragment data. Correct peaks are labelled as 1 (highlighted blue), and the false peaks are labelled as 0 (highlighted magenta).

After extracting and labelling all the peaks for each of the FSA files, the data was exported into a csv file which contain the label and features of the peaks as represented in Table 1.

Region (~3000)	Label (0, 1)	Area of the region	No. of peaks in region	Length of selected peak (bp)	Height of selected peak (RFU)
A_BOH_12_12.FSA					
1	0	3,057	6	41	557
2	0	3,183.5	7	44	730
3	0	7,437	6	111	553
4	0	3,920.5	7	118	9,146
5	0	15,616	8	135	6,168

6	1	33,573	5	244	5,122
7	1	18,153	4	288	2,672
...	...	...	...	...	...

Table 1. CSV file containing the features as dataset for machine learning.

Machine Learning model

Using the open source machine learning library Scikit-learn, the data was fitted into a Random Forest Classifier with 5-fold cross-validation using GridSearchCV (Pedregosa *et al.,* 2011). The features included the area of the peak, the number of peaks in the region, length and height of selected peak in region. A 75-25 split was done respectively for the training and testing of the model using sklearn function *train_test_split*.

AuSOME-ML

After training and cross-validation, the model was serialized for further usage via the Python module *pickle*. The model is an integral part of *AuSOME-ML*. When loading an FSA file from the testing dataset, *AuSOME-ML* detects all the peaks existing between 35 bp and 500 bp allele in the size standard and create their regions. For each of these regions, all 4 features were extracted and fed into the trained model. Only peaks for which the model returns a label of 1 were called as a true peak corresponding to an allele call and display its corresponding length in bp.

Results

Hyperparameter tuning

After 5-fold cross validation of the model using GridSearchCV, the best parameters is shown in Table 2.

Parameter	Value
criterion	‘entropy’
max_features	‘auto’
n_estimators	50

Table 2. Best parameters for GridSearchCV for cross validation of the model.

Model performance

The confusion matrix was obtained to measure the accuracy of the model in classifying the testing dataset as shown in figure 3. Out of 572 peaks in the testing dataset labelled as noise (0), 569 were correctly predicted by the model as noise (true negatives) while 3 were predicted as true peaks (false negatives). On the other hand, in 177 peaks labelled as true peaks in the dataset, 164 were correctly predicted (true positives) while 13 were incorrectly predicted by the model (false positives) .

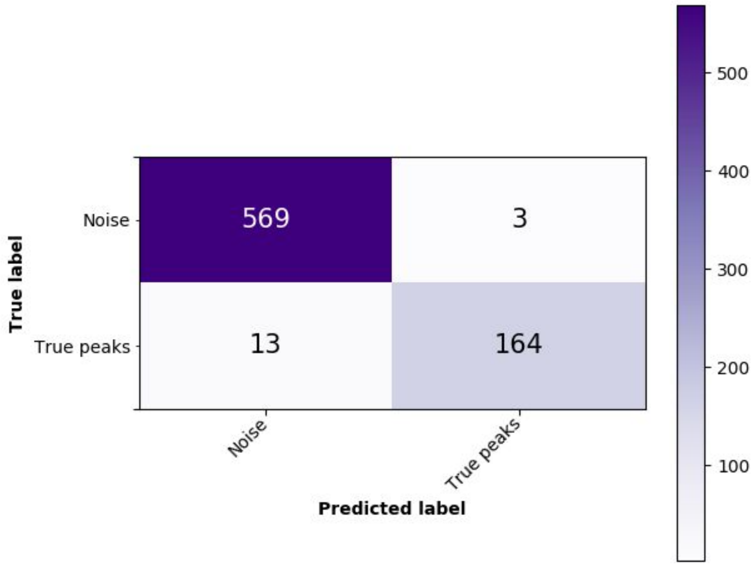

Figure 3. Confusion matrix for the Random Forest Classifier Model.

Table 3 below shows the classification report for the model which shows precision, recall, f1 score, and accuracy as metric of its performance. The model got a 0.98 precision both in predicting noise and true peaks while the recall (sensitivity) values were 0.99 and 0.93 for predicting noise and true peaks respectively. Noise prediction got an f1 score of 0.99 while true peak prediction got an f1 score of 0.95. The overall accuracy of the model against the testing dataset composed of 749 regions is 0.98.

	precision	recall	f1-score	support
0	0.98	0.99	0.99	572
1	0.98	0.93	0.95	177
Accuracy			0.98	749

Table 3. Accuracy report for the model

To better determine the capability of the model to differentiate between the two prediction classes, noise (0) and true peaks (1), the Receiver Operating Characteristics (ROC) was obtained as shown in figure 4. The Area Under the Curve (AUC) from the ROC is 0.99727

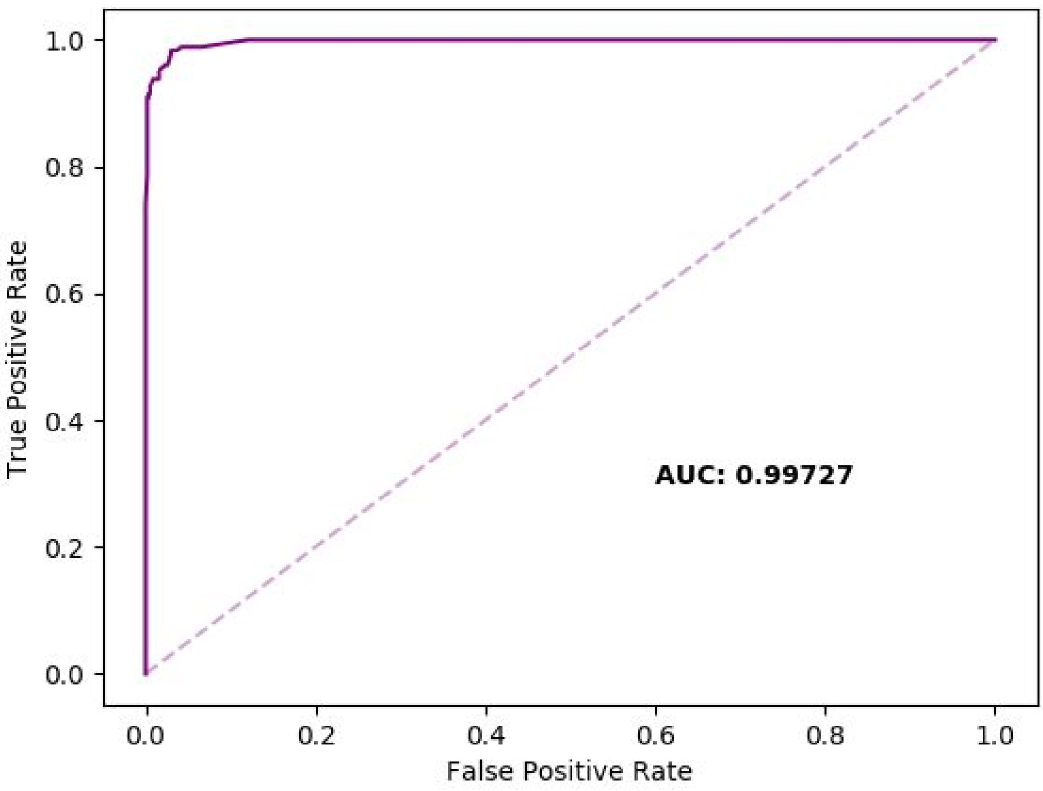

Figure 4. Receiver Operating Characteristics Curve and Area Under the Curve

Feature Importance

Evaluating the importance of features in an artificial learning model is important to infer which of the features give more weight in the prediction. Feature importance was obtained using the attribute feature_importances_ of RandomForestClassifier. Figure 5 shows the feature importance for the 4 features used in the model; area under the region, number of peaks, corresponding length of the peak in bp, and the height of the peak with the feature importance of 0.237, 0.043, 0.315, and 0.405 respectively.

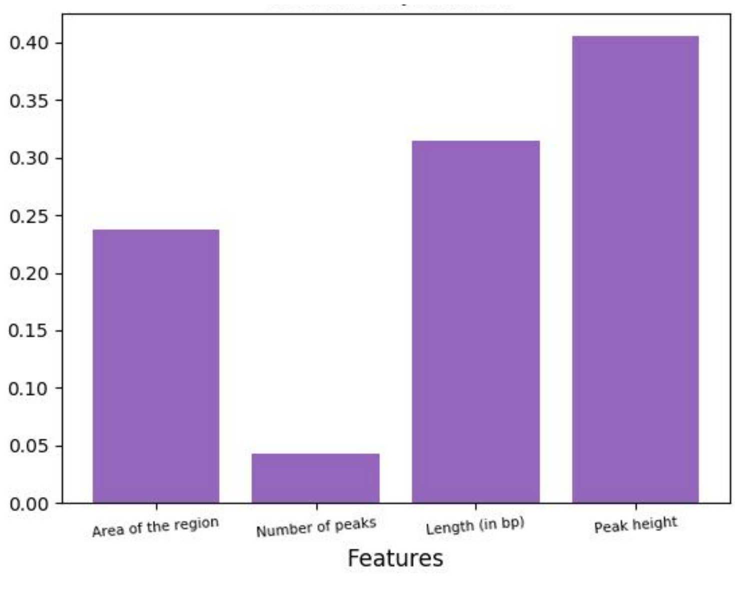

Figure 5. Feature importance

Discussion

Feature Selection

The features that were used were the area under the peak and peak height.

Daniels et al (1998) showed that the area under the peaks is a viable differentiator of peaks from noise. Although the methods they used were slow, including taking screenshots and measuring the area under the peak using pixel data. *AuSOME-ML* streamlines the process and eliminate possible information loss by skipping over pixel measurement and directly analyzing the raw sequencing data.

Peak height, is generally a good indicator that most fragment analysis packages rely on it in suggesting peaks (Covarrubias-Pazaran *et al.,* 2016; Biomatters Ltd. 2018).

In the model dataset, there is a larger number of noise peak calls than true peak calls instead of them being in equal number. This might skew the model into training to call out noise instead of calling true peaks. However FSA files do tend to have peaks which are noise than peaks.

Model performance

Accuracy in regression models generally include comparison of the training data to novel testing data. The confusion matrix in figure 3 is generally a good metric of the performance of a model. However, the data for the labels is imbalanced, with more data being labelled as noise (0) than true peaks (1). As shown in table 3, the testing data set is also imbalanced having 572 data labelled as noise and only 177 labelled as true peaks. This will generally skew the model to have higher accuracy in predicting noise rather than true peaks. However, in a real FSA file there are also generally more noise than true peaks that a model with more sensitivity to noise would be a good classifier. This is also reflected in table 3 where the f1 score for both predictions is high. Furthermore, the ROC curve for the model also shows that it can distinguish well between the two classes with a high AUC.

Feature importance

By evaluating the feature importances, it was found out that the feature which gives the highest weight in predicting the classes is the height of the peak followed by the corresponding length of the peak in bp and then the area under the region. The feature with the least importance is the number of peaks in the region which suggests that this feature is not informative in predicting whether a peak is either a noise or a true peak.

Test validation

Figure 5 shows the example output of *AuSOME-ML* for some testing dataset. While the performance metrics shows that the model has high accuracy as shown in panels b and d, panels a and c are the cases where the model either classifies the true peak as nosie (panel a) or classifies noise or stutter patterns as true peaks (panel c).

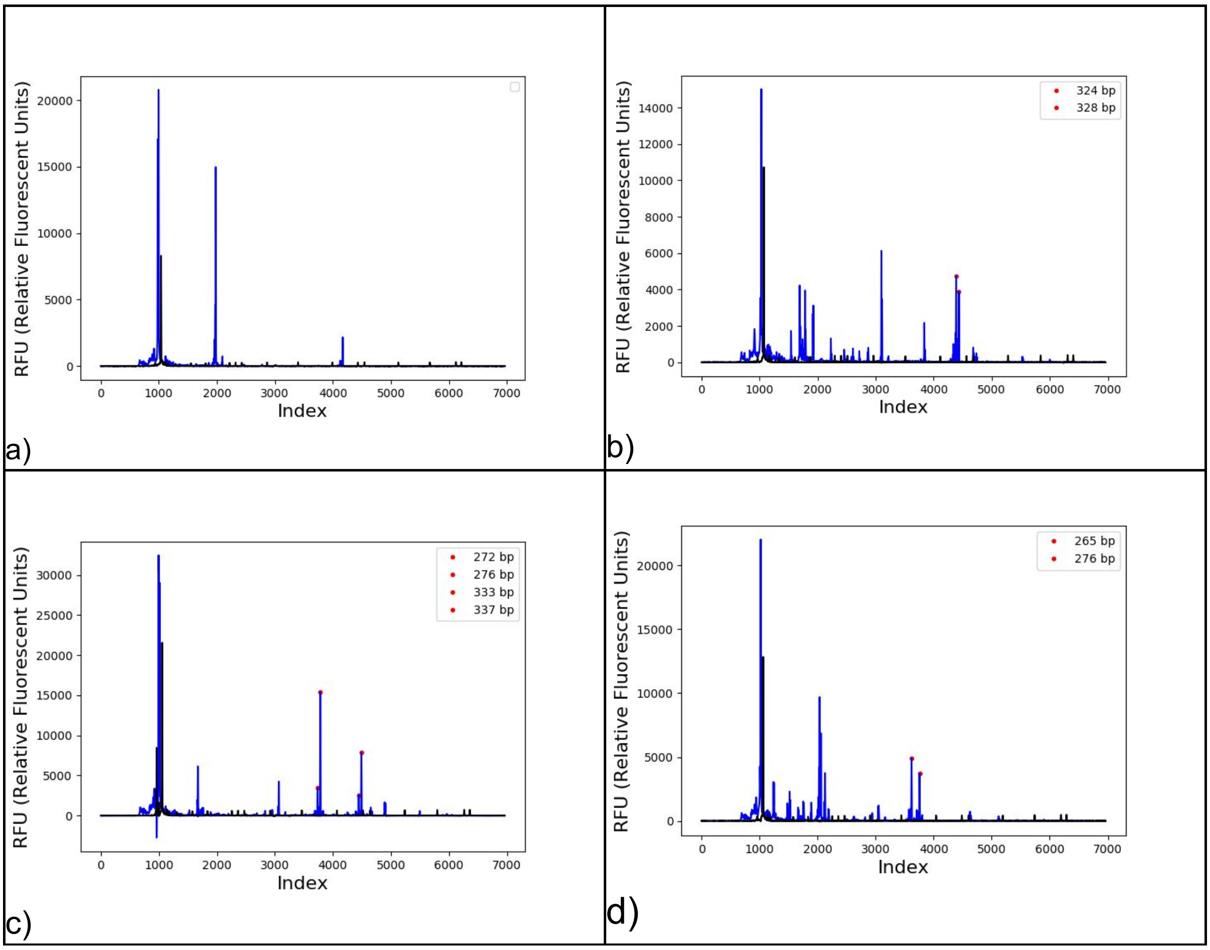

Figure 6. Mistaken calls by AuSOME-ML. a) A_COR_12_11_Hos null calls. It was not able to call the peak in the graph probably because the peak height is relatively low. **b)** A_GUI_12_10_Hos the model accurately determined the true allele calls to be 324 bp and 328 bp. **c)** A_TWI_12_18_Hos in this case, the model predicted the stutters of the true peaks as peaks, resulting in two extra calls. **d)** A_COR_12_26_Hos the model accurately determined the true peaks with 265 bp and 276 bp as the corresponding lengths.

Conclusion

AuSOME-ML is a capable predictive model that can automate the allele calling of microsatellite data. Specifically it can:

1. Directly interact with FSA file formats, and
2. Predict the peaks of alleles in sequencing data.

References

Applied Biosystems. (2006). Applied Biosystems Genetic Analysis Data File Format. pp. 36-37. Accessed on 9 October, 2019 at https://web.archive.org/web/20191010165226/https://projects.nfstc.org/workshops/resources/articles/ABIF_File_Format.pdf.

Brinkmann, B., Klintschar, M., Neuhuber, F., Hühne, J., & Rolf, B. (1998). Mutation rate in human microsatellites: influence of the structure and length of the tandem repeat. The American Journal of Human Genetics, 62(6), 1408-1415.

Brooker, R. J. (1999). Genetics: analysis & principles. Reading, MA: Addison-Wesley.

Biomatters Ltd. (2018). Geneious Microsatellite Plugin, Geneious Prime. Accessed on 11 October 2019 at https://web.archive.org/web/20170718173138/https://assets.geneious.com/documentation/geneious/GeneiousMicrosatsManual.pdf.

Cock PA, Antao T, Chang JT, Chapman BA, Cox CJ, Dalke A, Friedberg I, Hamelryck T, Kauff F, Wilczynski B and de Hoon MJL (2009) Biopython: freely available Python tools for computational molecular biology and bioinformatics. Bioinformatics, 25, 1422-1423

Covarrubias-Pazaran, G., Diaz-Garcia, L., Schlautman, B., Salazar, W., & Zalapa, J. (2016). Fragman: an R package for fragment analysis. *BMC genetics*, *17*(1), 62.

Daniels, J., Holmans, P., Williams, N., Turic, D., McGuffin, P., Plomin, R., & Owen, M. J. (1998). A simple method for analyzing microsatellite allele image patterns generated from DNA pools and its application to allelic association studies. The American Journal of Human Genetics, 62(5), 1189-1197.

Oliveira, E. J., Pádua, J. G., Zucchi, M. I., Vencovsky, R., & Vieira, M. L. C. (2006). Origin, evolution and genome distribution of microsatellites. Genetics and Molecular Biology, 29(2), 294-307.

Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., ... & Vanderplas, J. (2011). Scikit-learn: Machine learning in Python. Journal of machine learning research, 12(Oct), 2825-2830.

Ravago-Gotanco, R., & Kim, K. M. (2019). Regional genetic structure of sandfish *Holothuria* (*Metriatyla*) *scabra* populations across the Philippine archipelago. Fisheries research, 209, 143-155.

Slavič J. (2015). Py-tools/Findpeaks. GitHub Repository, accessed on 9 October 2019 at https://web.archive.org/web/20191009062915/https://github.com/jankoslavic/py-tools/blob/master/findpeaks/findpeaks.py

Schlötterer, C., & Pemberton, J. (1998). The use of microsatellites for genetic analysis of natural populations—a critical review. In Molecular approaches to ecology and evolution (pp. 71-86). Birkhäuser, Basel.

Remya, K. S., Joseph, S., Lakshmi, P. K., & Akhila, S. (2010). Microsatellites in varied arenas of research. *Journal of Pharmacy And Bioallied Sciences*, *2*(2), 141.

Minter, E. J., Lowe, C. D., Brockhurst, M. A., & Watts, P. C. (2015). A rapid and cost-effective quantitative microsatellite genotyping protocol to estimate intraspecific competition in protist microcosm experiments. Methods in Ecology and Evolution, 6(3), 315-323.

Gauffre, B., Galan, M., Bretagnolle, V., & Cosson, J. F. (2007). Polymorphic microsatellite loci and PCR multiplexing in the common vole, Microtus arvalis. Molecular Ecology Notes, 7(5), 830-832.