06

Construction of a nucleotide sequence using machine learning methods in the „Nanofor SPS“ sequencer

© V.V. Manoilov, A.G. Borodinov, A.I. Petrov, I.V. Zarutskiy, B.V. Bardin, A.Yu. Yamanovskaya, A.S. Saraev, V.E. Kurochkin

Institute for Analytical Instrumentation, Russian Academy of Sciences,
198095 St. Petersburg, Russia
e-mail: alex.niispb@yandex.ru

Received February 12, 2024
Revised June 14, 2024
Accepted July 8, 2024

The development of mathematical methods and information technologies for data processing plays an essential role in establishing various features in the analyzed nucleic acids and is a necessary element in the development and improvement of instruments and devices for practical use in biology and medicine. The technology of mass parallel sequencing of nucleic acids includes the process of measuring the intensities of fluorescence signals based on mathematical processing of images obtained from video cameras, and then constructing a sequence of nucleotides based on the results of these measurements. The paper considers the methods of information processing, which are divided into two parts. The first part includes methods for filtering images, detecting fluorescence clusters, and evaluating the parameters of fluorescence signals, both for single clusters and for clusters „superimposed“ on each other. The second part of the information processing methods considered in this work includes methods for constructing a sequence of letter codes of DNA nucleotides based on the intensities of fluorescence signals obtained directly from the results of image processing. No adjustments have been made to such signals related to intensity changes due to phenomena such as Phasing/Prephasing, signal attenuation and Cross-talk. These methods use classifiers based on machine learning. It is shown that as a result of the performed approbation of various machine learning models for the task of constructing a sequence of nucleotides, the results obtained showed sufficiently high quality indicators of genetic analysis. The quality indicators of the Phred score were in the range from 29 to 35 for the reference genome of the bacteriophage Phix174.

Keywords: sequencing, nucleic acids, image processing, improving the quality of genetic analysis, machine learning.

DOI: 10.61011/TP.2024.09.59293.35-24

Introduction

The use of modern information technologies and mathematical methods of data processing to study the analyzed nucleic acids is an important element of the successful development of genomic sequencing. The Institute of Analytical Instrumentation of the Russian Academy of Sciences (IAI RAS) developed a hardware and software complex (HSC) for decoding the sequence of nucleic acids by mass parallel sequencing („Nanofor SPS“) [1]. The solution of the problem of decoding the genome in the HSC is divided into a number of stages of processing the initial data. One of the important initial stages of data processing is the estimation of the intensity values of fluorescence signals for different wavelengths on the image frames of the reaction cell for several sequencing cycles by synthesis method. Such an assessment is performed using image processing programs, the algorithms of which are described in Ref. [2,3].

The technique of mass parallel sequencing is based on the principle of DNA synthesis using fluorescently labeled nucleotides. The process begins with the preparation of libraries, where special adapters are attached to DNA fragments. These fragments are fixed on the surface of the reaction cell, forming a dense array of cloned DNA chains — clusters. Thus, each cluster is a set of copies of the same DNA fragment.

During sequencing, fluorescently labeled nucleotides are alternately added to the growing DNA strands. Each nucleotide carries a unique fluorophore that emits light at a specific wavelength when excited by a laser. The resulting fluorescent signal is passed through light filters tuned to different wavelengths corresponding to the emission of the labelled nucleotides. After passing through the filters, the fluorescence signal is recorded by video cameras. Four video cameras are installed in the sequencer, each of which registers signals of one of the types of nucleotides (channels): adenine (A), cytosine (C), guanine (G) and thymine (T).

The image is captured on each of the four channels after the addition of nucleotides to the DNA fragments. The next stage begins upon completion of recording of fluorescent signals along the entire length of the reaction cell. Reagents that remove the dye (fluorophore) and stop the synthesis process are passed through microchannels at this stage. New reagents are then added to start the next synthesis cycle.

The process is repeated cyclically, adding nucleotides one by one, recording their signals with video cameras until the synthesis of the entire sequence is completed.

Image processing programs developed at the IAI RAS solve the problems of estimating the intensity values of fluorescence signals and further decoding the nucleotide sequence (base-calling), but they have a number of disadvantages associated with incomplete correction of errors caused by a number of factors that distort the results of genetic analysis. Such factors include: changes of the values of the recorded intensities due to phenomena such as phasing/prephasing, signal attenuation and cross-talk [4,5].

Machine learning (ML) methods, which are considered in this paper, are promising for leveling these shortcomings and conducting genetic analysis without correcting the described interference.

The use of ML in DNA sequencing tasks includes the creation and evaluation of models using algorithms capable of recognizing, classifying and predicting certain results based on the data obtained [6]. ML approaches are divided into unsupervised learning, semi-supervised learning, and supervised learning [7]. For example, often the purpose of supervised ML applied to sequencing data is to build a model based on a training set of collected observations with a known nucleotide sequence in order to predict a nucleotide for an arbitrary sample with an unknown target value of the type of nucleotide being determined, for example, with a sequence of bacteriophage Phix174 nucleotides (reference genome). In this case, the input variables are often called features, and the corresponding samples are called observations.

ML can play a key role in improving the accuracy and speed of this process. It is used in the following stages of the analysis.

• Data preprocessing. The sequencing data includes raw signals from detectors, such as electrical signals, fluorescence, or intensity graphs. ML helps to filter noise and calibrate data to improve the quality of the original signal.

• Model training. ML models are trained on a large volume of annotated data where the correct nucleotide sequences are known. This helps the models recognize complex patterns in the signals corresponding to different nucleotides.

• Classification and prediction. Modern ML methods, especially of the Deep Learning methods, such as convolutional neural networks (CNN) and recurrent neural networks (RNN), allow models to classify signals and predict nucleotide sequences with high accuracy. These models can take into account contextual information and sequences of neighboring nucleotides for more accurate base-calling.

• Error correction. ML algorithms can also be used to justify and correct errors resulting from sequencing. This includes analyzing the contextual frequencies of occurrence of certain nucleotides and using alignment algorithms.

• Integration with other bioinformatics tools. Base-calling results are often integrated with other genomic data analysis tools for further annotation and interpretation, which may also include additional ML steps, for example, for Oxford Nanopore Technologies sequencing technologies.

An overview of machine learning methods for solving nucleotide sequencing problems was provided in the article Ref. [8], and several examples of ML applications for data processing of the „Nanofor SPS“ sequencer were considered. In addition, various ways to combat the problem of overfitting are considered: by regularizing the model, choosing a simpler model with fewer parameters, and reducing the dimension of the feature space for learning.

The purpose of this work is to search for methods of image processing of fluorescence signals that can improve the quality of genetic analysis.

The following main tasks need to be solved for achieving this goal:

1. Analysis of the image processing algorithms [2,3] performing image filtering, detection of fluorescence clusters, evaluation of parameters of fluorescence signals for both single clusters and clusters „superimposed“ on each other. Development of a new algorithm for separating clumped objects, which will increase the density of clusters in the analyzed sample and thereby the number of nucleotide bases in the results of genetic analysis.

2. Demonstration of the prospects of the ML method for solving the problem of constructing a sequence of nucleotides using a number of sequencing data from „Nanofor SPS“ device. Improvement of the quality of genetic analysis as a result of application of various ML models to the task of constructing a sequence of nucleotides.

3. Evaluation of the possibility of simplification of ML models by reducing the dimension of features to perform the learning process based on the experimental data of „Nanofor SPS“ device.

Improving the accuracy and reliability of genetic analysis is especially important for its application in biomedicine. High-quality base-calling leads to increased reliability of subsequent stages of genomic data analysis, such as alignment, annotation and interpretation. If the base-calling procedure fails, incorrect genetic interpretations may occur, which is especially critical for diagnostic studies.

1. Analysis of image processing algorithms in the sequencer „Nanofor SPS“

The paper [2] lists the main stages of image processing of fluorescence signals in the device „Nanofor SPS“ and describes the algorithms for their implementation. Among these stages, the most important are: sharpening filtering (SF), object detection, correction of the background component and separation of „clumped“ objects.

An explanation of the operation of the SF algorithm is given in section 1.1, an algorithm for determining the

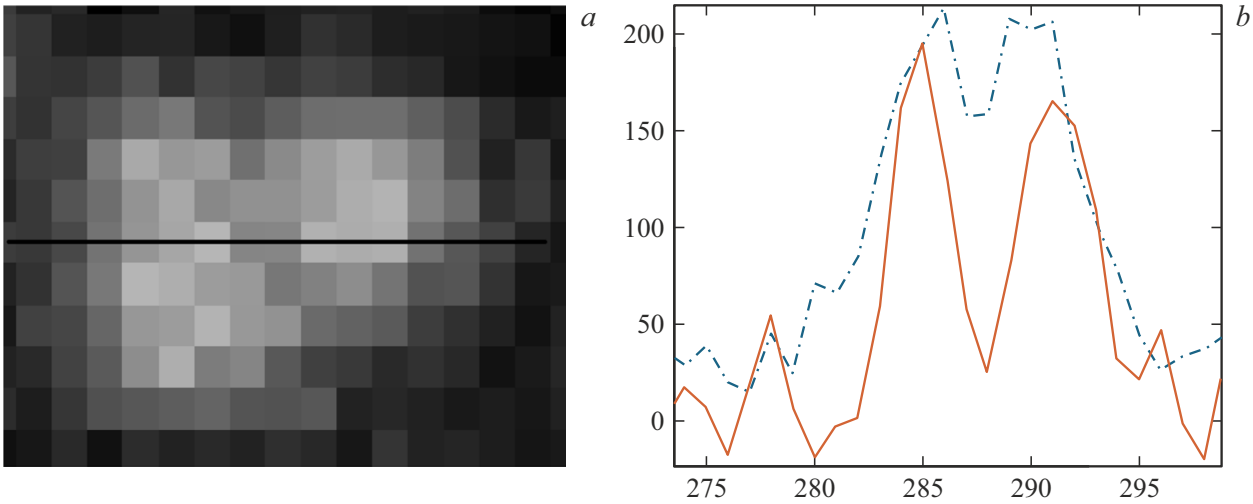

Figure 1. Comparison of signal profiles before and after sharpening filtering. Before processing — dashed dotted line. After processing — solid line. On the abscissa axis — pixel numbers on the horizontal axis.

threshold for detecting objects is provided, and a new algorithm for separating „clumped" objects is described.

1.1. SF algorithm

SF image processing allows narrowing objects for separation when they stick together. In addition, SF restores the original signal distorted by the hardware function [9,10]. The SF algorithm is based on the inverse two-dimensional transformation of the Fourier product of the Fourier image of the original image and the Fourier image of the second-order derivative of the Gaussian function with a width equal to about half the average width of the image of the fluorescence object [2,3,10]. The algorithm described in Ref. [2] is used to remove the background component.

1.1.1. Fluorescence signals before and after SF

Let us provide an example of how the SF program works. To do this, we will plot a profile of the line shown by the black line passing through the pixel with the maximum brightness of the image fragment shown in Fig. 1, *a*. The original signal is shown in Fig. 1, *b* as a dotted line and the signal after processing using the SF algorithm SF is shown by a solid line. The figure shows that as a result of the SF, the two fluorescence clusters that almost stuck together separated.

1.2. Plotting histograms to determine the threshold for detecting objects

It is important to determine a threshold that would reliably separate the „signal" (object) from noise and interference for detection of fluorescence objects against a background of noise and find the coordinates of their centers.

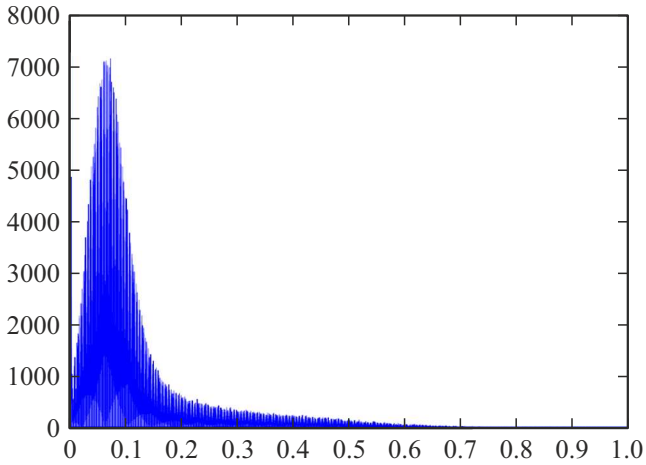

Figure 2. Histogram of signal intensities in channel A. Intervals of normalized intensities are plotted on the abscissa axis. The number of intensities that fall within a certain interval is plotted on the ordinate axis.

The signal intensity distributions of different nucleotides (A, C, G, T) differ from each other, and therefore the threshold values for images of fluorescence signals of each of the nucleotides will be different. Let's call the images obtained for the nucleotide fluorescence signals A, C, G, T, respectively, channels A, C, G, T. Histograms of the distribution of signal intensities normalized to the maximum value in each pixel are plotted to determine the threshold values for each of the channels. Intensity intervals from 0 to 1 with a step of 0.001 are postponed along the horizontal axis of the histogram. Figure 2 shows a histogram of the intensity distribution of channel A.

As can be seen from Fig. 2, the intensity distribution of fluorescence signals is a single-modal asymmetric function. Intensities other than noise „contribute" to the asymmetric

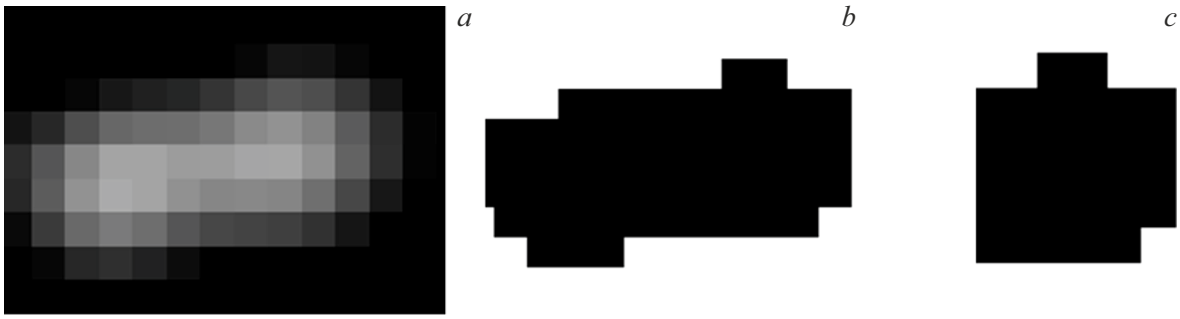

Figure 3. An example of two objects sticking together and the results of processing using an iterative algorithm: *a* — image of two objects sticking together in grayscale, *b* — binary image of two objects sticking together after threshold processing before the start of the iterative procedure, *c* — a binary image of one of the two objects sticking together after the first iteration, as a result of which the brighter object is removed.

part of the function. Determining the threshold using histograms with an asymmetric distribution function is a little more complicated than for histograms with a two-modal distribution function, which is described in Otsu's paper [11]. The threshold for two-modal distribution functions is defined as the average value between two maxima in the distribution function. In our case, the detection threshold was determined using histograms as follows. The root mean square (RMS) value of the noise was estimated. The noise RMS estimate is the value of the half-width at the half-height of the peak of the histogram. The threshold value is equal to the product of the coefficient k by the RMS estimate. The coefficient k was selected experimentally by processing a large number of images of fluorescence signals. The coefficient $k = 9$ turned out to be suitable for determining the threshold for the majority of tasks.

IAI RAS developed programs based on algorithms for processing aggravating filtration, detection and evaluation of parameters of clusters of fluorescence signals and these programs were implemented in prototypes of devices of „Nanofor SPS“. An analysis of the results of these programs demonstrated that the programs reliably detect clusters of fluorescence signals with a minimum amplitude of the useful signal of 80 conventional units and a RMS noise value of about 15 conventional units. The background component in the device of „Nanofor SPS“, as a rule, is a nonlinear function with intensity values from 50 to 150 units. The initial image size in the device of „Nanofor SPS“ is 2000×2400 pixels. Due to the fact that the width of the cluster at its half-height is from 6 to 10 pixels, the background component under the cluster can be considered linear. As noted in Ref. [2], the sharpening filtering almost completely corrects the impact of the background component, which can also be seen from Fig. 1. Negative signal values that occur after exacerbating filtration are replaced by zeros and do not affect the quality of the nucleotide sequence construction.

1.3. Iterative algorithm for separating clumped objects

Figure 3, *a* shows an image of two objects sticking to each other in grayscale.

The essence of the iterative algorithm is as follows. As a result of threshold processing, a binary image is obtained in which the areas of the original image exceeding the threshold are replaced with zeros and depicted in black, and the remaining areas are replaced with ones and depicted in white. Regions consisting of zeros that have no connections with other regions consisting of zeros are searched in the binary image obtained as a result of threshold processing, presented as an example in Fig. 3, *a*. An example of such an area is shown in Fig. 3, *b*. Next, an image of the same area is used, but in grayscale. The coordinates of the brightest pixel are in this region. Then the area (usually 7×7 or 9×9 pixels) of the brightest object in a binary image is replaced with units, and this area becomes white. Thus, the brightest of them is removed from the image of two objects sticking together and the procedure for searching for coordinates corresponding to fluorescence objects continues. If there were fragments in the original image consisting of three objects sticking together, then the coordinates of the brightest object are first determined and remembered. Then this object is deleted and a fragment consisting of two objects already stuck together is analyzed, and so on. In the working program of the sequencer of „Nanofor SPS“ the procedure for separating objects sticking together lasts up to 5 iterations, i.e. fragments of images are separated, which can contain from two to five objects sticking together, which is sufficient for those images that are obtained in the device at different densities of fluorescence objects.

The application of this iterative algorithm made it possible to distinguish clusters of nucleic acids with a more than twofold increase in the loading density of the detected objects, which is approximately 106 clusters per square millimeter of the reaction cell and depends on the concentration of the test sample.

When developing a program for separating fluorescence objects sticking together, the „watershed“ algorithm was also studied, which is described in Ref. [12]. The implementation of this algorithm requires a large number of operations and cannot be implemented in real time. It is supposed to be used in post-processing to increase the reliability of genetic analysis.

2. ML in nucleotide sequence construction (base-calling) tasks

2.1. Problem formulation

The base-calling task pertains to class of classification tasks typical for the application of ML algorithms. Clusters of amplified DNA strands are detected in the images for various fluorescence channels in the form of patterns of various sizes and localization. Special software described in the paper [2,3] is used to determine the position of the spots and their intensity characteristics together with the parameters of the surrounding background.

Spot positions and radii are used to extract a number of characteristics from each spot and its immediate background. These functions are then used as input to ML algorithms. Feature extraction is performed by examining the light intensity of each pixel of a certain rectangular area of the spot and some surrounding background. Each image set consists of four microscopic images, one for each base. These indicators are used by the ML model for the task of base classification. Output data comprise the sequence of nucleotides with different bases [13].

The following features are extracted from images of fluorescence signals from each fluorescence object (cluster) and its immediate background for the ML algorithm: for background (BG) — max, mean, median and mode; for the central area of the cluster image (FG) — max, mean, pct90 and pct99, where max — maximum intensity value, mean — arithmetic mean, mode — the most common value, pct90 and pct99 — 90- the 10th and 99th percentiles, respectively. These features form the rows of the matrix M. Thus, the information about each nucleotide contains 8 signs (4 features for FG and 4 features for BG). The last column of such a matrix contains the label — the letter code of the nucleotide obtained from the data of a pre-sequenced sequence previously mapped to a known (reference) sequence, for example, the bacteriophage Phix174. The matrix M has 34 columns: the cluster number, 32 features for each nucleotide and the letter code of the nucleotide. The number of rows in the matrix M is determined by the number of clusters of fluorescence objects, information about which is used to build a training sample.

The following classification methods were used to solve the base-calling task:

- Perceptron model [14];
- Logistic regression model [15];
- model based on the support vector machine (SVM) [16];
- Decision tree model [17];
- Random forest model [18];
- model k-nearest neighbors (k-nearest neighbors) [19].

2.2. The Scikit-learn platform as a database of ML models

The Scikit-learn platform is used to apply various ML algorithms, which supports an easy-to-use interface that is closely integrated with Python [20]. The Scikit-learn API is optimally designed to work with ML methods. The main design principles according to the paper [21] are listed below:

- Consistency of object calling. All objects share a consistent and simple interface for calling functions.
- Estimator. Any object that is capable of estimating parameters based on a dataset is called an estimator (for example, an imputer designed for data recovery is an estimator). The estimation itself is performed using the fit method, which takes a single data set as a parameter (or two for learning algorithms with a teacher; the second data set contains labels). Any other parameter necessary to control the estimation process is considered a hyperparameter (for example, strategy in imputer) and should be specified as an instance variable.
- Transformers. Some estimators (such as imputer) can also transform a dataset; they are called transformers. The API interface is quite simple: the transformation is performed by the transform method, to which the data set to be transformed is passed in the parameter. It returns a transformed dataset. All transformers have a convenient fit_transform method, which is the equivalent of calling fit sequentially and then transform.
- Predictors. Finally, some estimators are able to make predictions with a set of data; they are called predictors. The predictor has a predict method that accepts a dataset with new samples and returns a dataset with corresponding predictions. The predictor also has a score method that evaluates the quality of predictions using a specified test set and appropriate labels in the case of learning algorithms with a teacher.
- Inspection. All predictor hyperparameters are available directly through instance variables (for example, imputer.strategy), and all studied predictor parameters are also available through open instance variables with an underscore suffix (for example, imputer.statistics_).
- Non-proliferation of classes. Datasets are represented as NumPy arrays or sparse SciPy matrices, instead of self-made classes. Hyperparameters are just regular Python strings or numbers.
- Ease of composition. Existing building blocks are reused as much as possible. For example, it is easy to create a Pipeline predictor from an arbitrary sequence of transformers, followed by a call to the final predictor.
- Standard default parameter values. Scikit-learn provides reasonable standard values for most parameters, making it easier to quickly create a basic working system.

We will divide the dataset into train and test datasets to assess the quality of the trained model with previously unknown data. To do this, we will use the train_test_split function from the model_selection module of the Scikit-learn library, randomly dividing data arrays into train and test samples. In the case of the classification task in the base-calling task, we perform random data splitting, allocating 30% of the data for testing and 70% for training. The special function Scikit-learn train_test_split already performs internal mixing of the training data before splitting. Otherwise, the order of examples from different classes in the training datasets could be critical, which is undesirable. By using the RandomState parameter with a fixed random initial number, we guarantee reproducibility of the results in subsequent learning processes. In addition, as a rule, we use built-in stratification support in all machine learning methods. Stratification implies that the train_test_split method returns training and test subsets with the same proportions of class labels as in the original dataset.

Many ML and optimization algorithms require scaling of the input data for optimal performance. The StandardScaler class from the Scikit-learn preprocessing module is used for this function. It is important to use the same scaling parameters to standardize the training and test datasets so that the values in both datasets are comparable.

There are several accuracy metrics in ML that are used to evaluate the performance and adequacy of learning models. The accuracy of classification in the process of model selection was evaluated in this paper using the following metrics[22]:

1. Accuracy. Accuracy measures the proportion of correct predictions of a model relative to the total number of predictions. $\mathrm{ACC} = (\mathrm{TP} + \mathrm{TN})/(\mathrm{TP} + \mathrm{TN} + \mathrm{FP} + \mathrm{FN})$, where TP — true positive, TN — true negative, FP —false positive and FN — false negative predictions.

2. Completeness (Recall). It shows how much of the positive cases the model is able to correctly detect out of all the real positive ones. $\mathrm{Recall} = \mathrm{TP}/(\mathrm{TP} + \mathrm{FN})$.

3. Precision. Determines the proportion of truly positive predictions relative to all positive predictions of the model. $\mathrm{Precision} = \mathrm{TP}/(\mathrm{TP} + \mathrm{FP})$.

4. F-measure (F1-score). It is the average harmonic between completeness and accuracy. $\text{F1-score} = 2 \cdot (\mathrm{Precision} \cdot \mathrm{Recall})/(\mathrm{Precision} + \mathrm{Recall})$.

5. The area under the ROC curve (ROC AUC). Evaluates the quality of binary classification by measuring the area under the ROC curve (Receiver Operating Characteristic). It shows how well the model distinguishes between positive and negative classes.

These accuracy measures were used in all the studied learning models. In our final results, we focused on evaluating the accuracy of learning in terms of classification error (misclassification error $= 1 - \mathrm{ACC}$), since it most accurately corresponds to the Phred quality score (explained below), which is a generally accepted measure of the quality of identification of nitrogenous bases obtained by automatic DNA sequencing.

Also, cross-validation was used in all training methods as a method of evaluating the ML model, which allows you to evaluate how well the model generalizes data, — a necessary process when working with limited amounts of data [23]. The data were divided into several parts for cross-validation and these parts were called „folds“ (in practice, from 3 to 5 parts). The model was then trained on several combinations of these folds and evaluated on the remaining parts of the data. By repeating this process by cross-checking, an integral evaluation of the model was obtained for each part of the data. Cross-validation helps to use the available data more effectively to evaluate the model and make decisions about its accuracy, which is especially important in the task under consideration, in which there is critically insufficient data for training.

2.3. Results of the selection of ML models

As a result of applying various ML models to the base-calling problem, the obtained results on prediction accuracy can be presented in a summary table. The values of the intensities of cluster fluorescence signals obtained from the parallel sequencing system „Nanofor SPS“ were used as input data. Then, based on these intensities, statistical characteristics such as mean, median, pct99, pct90, etc. were extracted. At the same time, corrections of intensity changes due to phenomena such as Phasing/Prephasing, signal attenuation and cross-talk were not performed.

The right column of the table shows the Phred quality score quality indicators adopted in bioinformatics. Phred quality estimates are logarithmically related to the probability of errors in the construction of a sequence of letters of nucleotides and are defined as

$$Q = -10 \cdot \log_{10} P.$$

This ratio can be written as

$$P = 10^{\frac{-Q}{10}}.$$

For example, Phred assigns a letter a quality score of 30. The probability that this letter in the sequence was named incorrectly is 1 in 1000. In other words, the probability of the letter being correct is 99.9%.

Results of using different ML models

ML Model	Error classifications	Phred quality score
Perceptron	0.0008	30.9
Logistic regression	0.0003	35.2
SVM	0.0006	32.2
Decision tree	0.0012	29.2
Random forest	0.0005	33.0
KNN	0.0003	35.2

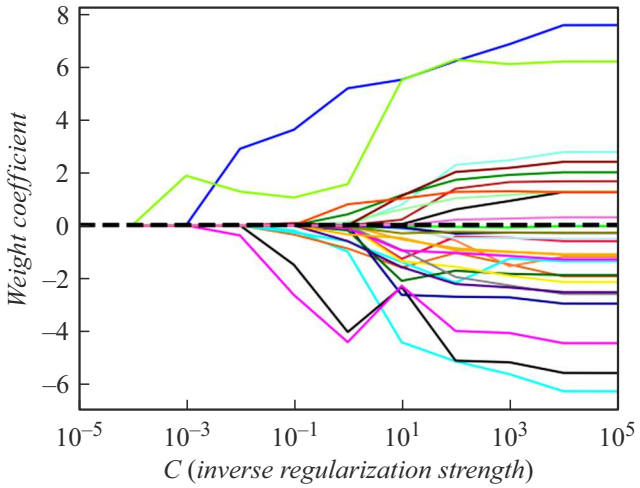

Figure 4. Demonstration of the regularization effect in the form of reducing the weights of logistic regression when changing the regularization parameters.

All ML models have demonstrated high accuracy in predicting the nucleotide sequence in the base-calling process. It is interesting to note that the achieved prediction accuracy based on the experimental data of the „Nanofor SPS“ device largely corresponds to the Phred quality score calculated in accordance with the generally accepted protocols of Illumina [24,25]. The Random forest model predictably surpassed the Decision tree model. The excellent results of the simple KNN model, comparable to the regularized logistic regression model, indicate the need to optimize the selection of features using the dimensionality reduction method.

3. The optimal choice of features to reduce the dimension

As mentioned earlier, in the process of testing various models, it is noted that in many cases the model demonstrates significantly higher accuracy on the training data set than on the test set, which indicates retraining. When using the Scikit-learn library, overfitting means that the model adjusts parameters too precisely to specific observations in the training dataset, but does not generalize new data well, which manifests itself in the high variance of the model. The main reason for retraining is the excessive complexity of the model for the available training data. Common approaches to reducing generalization error include the following [26]:

- using more data in the training set;
- introducing a complexity penalty by regularizing the model;
- choosing a simpler model with fewer parameters;
- reducing the dimension of the data.

Collecting more training data is effective, but often not applicable. This paper considers common ways to reduce overfitting by regularization and dimensionality reduction through feature selection. This leads to simplification of models by reducing the number of parameters. The regularization parameters L1 and L2 are used as a penalty for the complexity of the model for the logistic regression model. And if the regularization of L2 uses an approach to reduce the complexity of the model by penalizing large individual weights for all parameters, then the regularization of L1 usually gives sparse feature vectors, and the weights of most features will be zero. In this sense, the regularization of L1 can be understood as a method for selecting features and reducing the dimension of the model. The above graph provides additional information about the regularization behavior of L1 (Fig. 4). The figure shows the change of the weight coefficients of 32 features (8 for each channel) from the inverse value of the regularization parameter C. Dependences of some features on the regularization parameter C are not visible, since they „merged“ with dependence on other features. It can be seen from the information presented in the figure that for the regularization parameter C, less than 0.1, the weight coefficients are not zero for only 4 features. This situation gave rise to studies of reduction of the feature space.

PCA (Principal component analysis) is an effective method for optimal feature selection for a machine algorithm [27]. The transformation of data containing information about fluorescence signals by the principal component method, and subsequent classification by the *k*-means method, allowed creating e an optimal sample for identifying the letter code of the nucleotide in this paper. The linear discriminant analysis (LDA) method is also applicable to solving the problem of dimension reduction. The general concept of LDA is very similar to PCA, but although PCA tries to find the orthogonal axes of the maximum variance components in the dataset, the goal of LDA is to find a feature subspace that optimizes class separability. Fig. 5 a shows the representation of the division of classes by belonging to a particular nucleotide (4 classes in the classification problem) when reducing the multidimensional feature space to two discriminants.

The use of the PCA algorithm and classification by the *k*-means method when processing the data of the „Nanofor SPS“ device showed that the amount of erroneous classification of nucleotides did not exceed 0.7%.

Finally, the *t*-SNE method was used to visualize multidimensional features in two-dimensional space [28]. It builds a data model based on their pairwise distances in a multidimensional feature space. Then this method finds the probability distribution of pairwise distances in a new space of lower dimension, close to the probability distribution of the same pairwise distances in the original space (Fig. 5, *b*). In other words, *t*-SNE is trained to map data points to a smaller space in such a way that pairwise distances in the original space are preserved. Fig. 5, *b* provides reason to be sure that the space of learning features can be effectively compressed to a minimum number of dimensions, although it does not give a specific implementation of such a reduction.

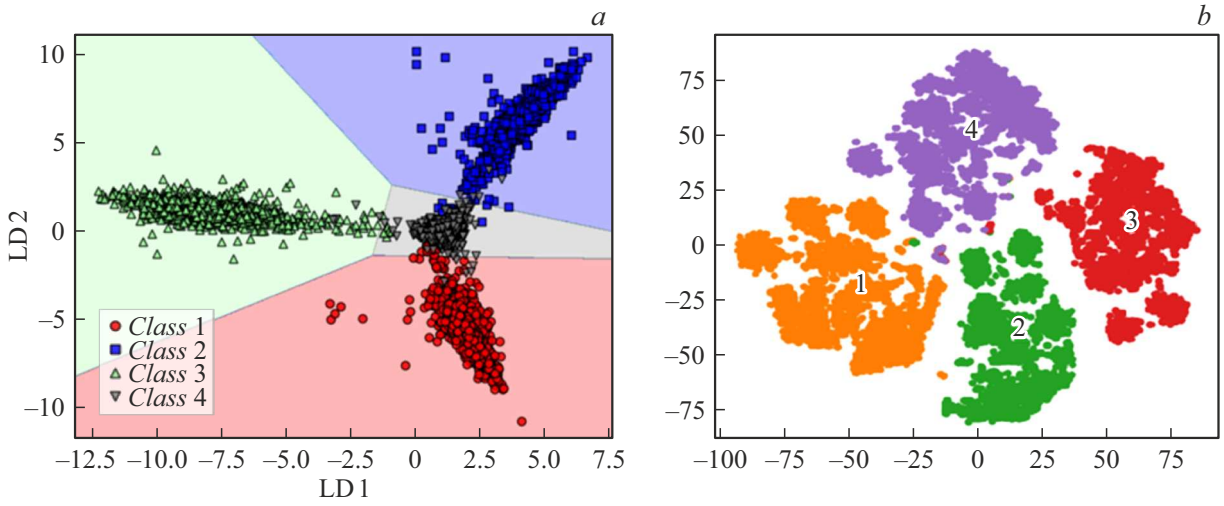

Figure 5. *a* —distribution of features of the training sample in the space of two discriminants, *b* — *t*-SNE reduction of the feature space of the training sample to two dimensions.

Conclusion

The considered algorithms and programs for preprocessing images are the development and refinement of the algorithms and programs described in the works [2,3,8]. An iterative algorithm processing grayscale and binary (black-and-white) images was added to the programs of separation of „superimposed“ fluorescence clusters. This algorithm makes it possible to increase the density of detectable clusters and thereby increase the number of nucleotide bases detected in the processed sample.

The results of testing various machine learning models for the base-calling task demonstrated fairly high quality of genetic analysis. The Phred score turned out to be in the range from 29 to 35 units, whereas the value of this score for performing base-calling in the „Nanofor SPS“ device without using ML methods is usually 30.

The values of the intensities of cluster fluorescence signals obtained from the parallel sequencing system of the „Nanofor SPS“ device were used as input data. Then these intensities were used to extract statistical characteristics of each cluster, such as mean, median, pct99, pct90, so that information about each nucleotide contains 8 features (4 features for the cluster image, 4 features for the background). A total of 32 freatures are obtained for four channels. At the same time, corrections of intensity changes due to phenomena such as phasing/prephasing, signal attenuation and cross-talk were not performed.

Based on the results of testing, it should be noted that decision trees are especially attractive if we care about interpretability, i.e. we are interested in explicitly highlighting the most informative features in the classification task when determining the type of nucleotide. Logistic regression is not only a useful model for real-time learning of new sequencing data (when using SGD, stochastic gradient descent in solving an optimization problem), but also allows us to predict the probability of classification truth.

Although SVMs are powerful linear models that can be extended to non-linear problems using the kernel trick, they require optimal tuning of many parameters to achieve good predictions. Ensemble methods, such as random forests, do not require time-consuming parameter tuning and avoid the effect of overfitting (unlike decision trees), which makes them attractive models for many practical problem areas. The KNN classifier offers an alternative approach to classification through „lazy learning“, which allows you to make predictions without any model training, but with a more computationally expensive prediction step.

It should also be noted that for the selected machine learning methods, the results were investigated and obtained to reduce the feature space from the characteristics of the data used, which made it possible to reduce the number of classification errors and simplify the calculation processes, since instead of 32 features of the characteristics of each cluster, only 4 were used.

The considered methods were implemented using the tools of the Scikit-learn system, which made it possible to ensure simplicity and clarity in the compilation of algorithms and programs.

Funding

The work was carried out with the financial support of the Ministry of Science and Higher Education of the Russian Federation within the framework of the draft Federal Scientific and Technical Program for the Development of Genetic Technologies for 2019–2027 (Agreement№ 075-15-2021-1057).

Conflict of interest

The authors declare that they have no conflict of interest.

References

[1] V.E. Kurochkin, Ya.I. Alekseev, D.G. Petrov, A.A. Evstrapov. Izvestia of the Russian VMA, **40** (3), 69 (2021) (in Russian). DOI: 10.33917/es-3.189.2023.36-41

[2] V.V. Manoilov, A.G. Borodinov, A.S. Saraev, A.I. Petrov, I.V. Zarutsky, V.E. Kurochkin. ZhTF, **92** (7), 985 (2022) (in Russian). DOI: 10.21883/JTF.2022.07.52655.318-21

[3] V.V. Manoilov, A.G. Borodinov, I.V. Zarutsky, A.I. Petrov, V.E. Kurochkin. Trudy SPIIRAN, **18** (4), 1010 (2019) (in Russian). DOI: 10.15622/sp.2019.18.4.1010-1036

[4] Kao, Wei-Chun. *Algorithms for Next-Generation High-Throughput Sequencing Technologies* (Thesis, University of California, 2011), https://escholarship.org/uc/item/86b9c87d

[5] RTA Theory of Operations v1.13 ILLUMINA PROPRIETARY Pub. No. 770-2009-020, current as of 09 Nov. 2011

[6] S. Paliwal, A. Sharma, S. Jain, S. Sharma. *Machine Learning and Deep Learning in Bioinformatics. In Bioinformatics and Computational Biology* (Chapman and Hall/CRC, 2024), p. 63–74

[7] H. Izadkhah. *Deep Learning in Bioinformatics: Techniques and Applications in Practice* (Academic Press, 2022)

[8] A.G. Borodinov, V.V. Manilov, I.V. Zarutsky, A.I. Petrov, V.E. Kurochkin, A.S. Saraev. Informatika i avtomatizaciya, **21** (3), 572 (2022) (in Russian). DOI: 10.15622/ia.2022.3.21

[9] R. Gonzalez, R. Woods. *Digital Image processing* (Tekhnosfera, M., 2005) (in Russian).

[10] B.V. Bardin, I.V. Chubinsky-Nadezhdin. Nauchn. Priborostr., **19** (4), 96 (2009) (in Russian).

[11] N. Otsu. IEEE Transactions on Systems, Man, and Cybernetics, **9** (1), 62 (1979).

[12] L. Najman, M. Schmitt. Signal Processing, **38** (1), 99 (1994).

[13] E. Tegfalk. *Application of Machine Learning Techniques to Perform Base-Calling in Next-Generation DNA Sequencing* (2020). https://www.diva-portal.org/smash/get/diva2:1465444/FULLTEXT01.pdf

[14] S.I. Gallant. IEEE Transactions on Neural Networks, **1** (2), 179 (1990). DOI: 10.1109/72.80230

[15] S. Dreiseitl, L. Ohno-Machado. J. Biomed. Inform., **35** (5–6), 352 (2002). DOI: 10.1016/S1532-0464(03)00034-0

[16] J. Abello, G. Carmode. (eds.). *Discrete Methods in Epidemiology.* DIMACS Series in Discrete Mathematics and Theoretical Computer Science, **70**, 13 (2004).

[17] L. Breiman, J.H. Friedman, R.A. Olshen, C.J. Stone. *Classification and Regression Trees* (Wadsworth & Brooks/Cole Advanced Books & Software, Monterey, CA, 1984) DOI: 10.1201/9781315139470

[18] G. Biau, E. Scornet. Test, **25**, 197 (2016). DOI: 10.1007/s11749-016-0481-7

[19] T. Hastie, R. Tibshirani, J. Friedman. *The Elements of Statistical Learning: Data Mining, Inference, and Prediction* (Springer, N Y., 2009), DOI: 10.1007/978-0-387-84858-7

[20] J. Hao, T.K. Ho. J. Educational and Behavioral Statistics, **44** (3), 348 (2019). DOI: 10.3102/1076998619832248

[21] L. Buitincketal. *API Design for Machine Learning Software: Experiences from the Scikit-Learn Project.* arXiv preprint arXiv:1309.0238. 2013.

[22] J. Zhou, A.H. Gandomi, F. Chen, A. Holzinger. Electronics, **10** (5), 593 (2021). DOI: 10.3390/electronics10050593

[23] F. Masoodi, M. Quasim, S. Bukhari, S. Dixit, S. Alam. (eds.). *Applications of Machine Learning and Deep Learning on Biological Data* (CRC Press, 2023), DOI: 10.1201/9781003328780

[24] *Quality Scores for Next-Generation Sequencing* (Illumina Inc., San Diego, CA, 2011)

[25] A.G. Borodinov, V.V. Manjilov, I.V. Zarutskiy, A.I. Petrov, V.I. Kurochkin. *Quality Control Metrics at Different Stages of Genomic Assembly in the Parallel Sequencing Using the Nanofor SPS.* XV International scientific-technical conference on actual problems of electronic instrument engineering (APEIE), IEEE, 516 (2021). DOI: 10.1109/APEIE52976.2021.9647574

[26] X. Li, L. Zhang, J. Yang, F. Teng. J. Med. Biolog. Engineering, **44**, 231 (2024). DOI: 10.1007/s40846-024-00863-x

[27] V.V. Manoilov, A.G. Borodinov, A.I. Petrov, I.V. Zarutsky, V.E. Kurochkin. Nauchn. Priborostr., **33** (2), 35 (2023) (in Russian).

[28] G.C. Linderman, S. Steinerberger. SIAM J. Mathematics of data Science, **1** (2), 313 (2019). DOI: 10.1137/18M1216134

Translated by A.Akhtyamov