Hindawi
Journal of Applied Mathematics
Volume 2023, Article ID 9991095, 13 pages
https://doi.org/10.1155/2023/9991095

Research Article

Detection of COVID-19 Using Protein Sequence Data via Machine Learning Classification Approach

Siti Aminah, Gianinna Ardaneswari, Mufarrido Husnah, Ghani Deori, and Handi Bagus Prasetyo

Department of Mathematics, Faculty of Mathematics and Natural Sciences, Universitas Indonesia, 16424, Indonesia

Correspondence should be addressed to Siti Aminah; aminah@sci.ui.ac.id

Received 31 January 2023; Revised 7 August 2023; Accepted 4 September 2023; Published 28 September 2023

Academic Editor: Kannan Krithivasan

The emergence of severe acute respiratory syndrome coronavirus 2 (SARS-CoV-2) in late 2019 resulted in the COVID-19 pandemic, necessitating rapid and accurate detection of pathogens through protein sequence data. This study is aimed at developing an efficient classification model for coronavirus protein sequences using machine learning algorithms and feature selection techniques to aid in the early detection and prediction of novel viruses. We utilized a dataset comprising 2000 protein sequences, including 1000 SARS-CoV-2 sequences and 1000 non-SARS-CoV-2 sequences. Feature extraction provided 27 essential features representing the primary structural data, achieved through the Discere package. To optimize performance, we employed machine learning classification algorithms such as K-nearest neighbor (KNN), XGBoost, and Naïve Bayes, along with feature selection techniques like genetic algorithm (GA), LASSO, and support vector machine recursive feature elimination (SVM-RFE). The SVM-RFE+KNN model exhibited exceptional performance, achieving a classification accuracy of 99.30%, specificity of 99.52%, and sensitivity of 99.55%. These results demonstrate the model's efficacy in accurately classifying coronavirus protein sequences. Our research successfully developed a robust classification model capable of early detection and prediction of protein sequences in SARS-CoV-2 and other coronaviruses. This advancement holds great promise in facilitating the development of targeted treatments and preventive strategies for combating future viral outbreaks.

1. Introduction

In early 2020, the world witnessed the unprecedented emergence of COVID-19, caused by the novel coronavirus SARS-CoV-2, leading to a devastating global pandemic. The rapid spread of the virus necessitated urgent research efforts to combat the disease and develop effective strategies for early detection and treatment. Protein sequence analysis has emerged as a critical tool in understanding the molecular basis of diseases, making it a vital area of study in virology and bioinformatics. In December 2019, this virus first appeared in Wuhan, China. This virus has characteristics like pneumonia, such as fever, dry cough, fatigue, and occasional gastrointestinal diseases [1]. As of March 27, 2021, according to WHO (World Health Organization), the virus has infected 123,216,178 people, and 2,714,517 of them have died. In Indonesia, as of March 27, 2021, according to the data on the website https://covid19.go.id, as many as 1,471,225 people have been infected with COVID-19, and 39,865 of them have died.

Every living creature has different characteristics. These characteristics can be seen using protein sequences; hence, SARS-CoV-2 can be classified using protein sequences. Protein itself is the order of amino acid that binds the peptide bonds and plays an important role in sustaining life [2]. The protein sequences of SARS-CoV-2 can be obtained from the protein data bank UniProt.

While previous studies have explored various machine learning and bioinformatics approaches for classifying protein sequences related to COVID-19, they encountered challenges such as limited classification accuracy, high computation time, and lack of biological interpretability. Additionally, achieving high specificity and sensitivity in protein sequence classification remained a significant hurdle.

To address these issues, this paper presents a novel approach aimed at advancing the field of protein sequence classification for COVID-19. Our proposed methodology harnesses the potential of feature selection and classification algorithms to accurately distinguish between SARS-CoV-2 and non-SARS-CoV-2 protein sequences. In the context of existing work, our approach seeks to overcome the limitations faced by previous research and offer a comprehensive analysis of the selected features and classification outcomes.

One of the key issues we address is the identification of optimal features for effective COVID-19 protein sequence classification. To achieve this, we employ three feature selection techniques: genetic algorithm, LASSO, and SVM-RFE. These methods play a pivotal role in identifying the most relevant and informative features that significantly contribute to the classification task. By extracting a subset of discriminative features, our proposed approach enhances the interpretability of the model, shedding light on the biological significance of the selected features.

Furthermore, to overcome the limitations of prior studies, we adopt a hybrid approach that combines multiple classification algorithms. Specifically, we utilize K-nearest neighbor (KNN), XGBoost, and Naïve Bayes as our classification methods. Through a rigorous comparison of their performance, we aim to identify the most effective model for accurate COVID-19 protein sequence classification. The integration of feature selection and classification algorithms is expected to yield a robust and high-performing model, contributing to improved pathogen detection and healthcare applications.

The K-nearest neighbor (KNN) method is one of the simple and popular classification algorithms in machine learning applications that is included in *The Top Ten Algorithms in Data Mining* [3]. Although simple, this algorithm yields outstanding performance in its application in various research fields [4]. Based on research by Arian et al., the classification of protein kinase inhibitors using the K-nearest neighbor (KNN) method and genetic algorithm feature selection gives an accuracy of 90% with the parameter $K = 7$ [5]. This inspires us to use the KNN classification method and genetic algorithm feature selection for coronavirus protein sequence data in our research.

Georganos et al. used XGBoost in their research as classification method and RFE as feature selection in a very-high-resolution remote sensing object-based urban application [6]. They used 1880 data in total with 169 features. As shown in Table 1, the XGBoost method with no selection feature achieved 77.8% accuracy. Furthermore, the XGBoost method with RFE as feature selection achieved an accuracy of 79.8% and $f1$-score of 98.2% by using 23 selected features, an accuracy of 79.2% and $f1$-score of 97.2% by using 22 selected features, and an accuracy of 79.2% and $f1$-score of 97.0% by using 25 selected features. Therefore, in their research, the best accuracy was achieved by using 23 selected features. There is an increase in accuracy of up to 2% if feature selection is carried out. In addition, the prediction accuracy of the hybrid method in XGBoost for protein submitochondrial localization prediction was about 98% [7].

One of the machine learning and information mining algorithms, Naïve Bayes, has been proven to have viability as well as its central role in data retrieval in general [8]. Nitta et al., in their research, use LASSO-based feature selection and Naïve Bayes classifier for crime prediction and its type. The LASSO-Naïve Bayes gives an accuracy of 97.47%, precision of 92.89%, and recall of 86.83% according to Table 2 [9].

Table 1: Confusion matrix.

	Actual positive	Actual negative
Predicted positive	True positive (TP)	False positive (FP)
Predicted negative	False negative (FN)	True negative (TN)

In this research, binary classification of coronavirus protein sequence data that caused COVID-19 was carried out using the K-nearest neighbor (KNN), XGBoost, and Naïve Bayes methods as classification algorithms with genetic algorithm (GA), LASSO, and SVM-RFE as feature selection methods. There are nine models that we will simulate, which are GA+KNN, SVM-RFE+KNN, LASSO+KNN, GA+Naïve Bayes, SVM-RFE+Naïve Bayes, LASSO+Naïve Bayes, GA +XGBoost, SVM-RFE+XGBoost, and LASSO+XGBoost.

The main contributions of this paper are summarized as follows: (1) introduction of a novel approach for classifying COVID-19 protein sequences by combining feature selection methods and classification algorithms; (2) extensive evaluation of three feature selection techniques, genetic algorithm, LASSO, and SVM-RFE, to identify the most relevant features for classification; (3) comparative analysis of three classification algorithms, K-nearest neighbor (KNN), XGBoost, and Naïve Bayes, to identify the best-performing model for COVID-19 protein sequence classification; (4) examination of nine different model combinations, providing valuable insights into the effectiveness of hybrid methods for protein sequence classification; and (5) potential application of the proposed approach in early detection of pathogens and targeted treatments for COVID-19, with implications in healthcare and virology.

The remainder of this paper is organized as follows: Section 2 provides a detailed explanation of the feature extraction package named Discere and its significance in extracting 27 features from protein sequences and also presents the methodology, including the dataset used, feature selection methods, and classification algorithms. In Section 3, we present and discuss the experimental results for each model simulation, highlighting the performance metrics achieved at different percentages of training data. We discuss the potential limitations of this study and future directions for research. Finally, Section 4 concludes the paper by summarizing the main findings and emphasizing the significance of the proposed approach in protein sequence classification for COVID-19 and beyond.

2. Materials and Methods

2.1. Dataset. The dataset used in this study was taken from a protein data bank, UniProt (https://www.uniprot.org/). UniProt provides free access to protein sequence database online. This study will use 2000 data of protein sequences, with 1000 protein sequences of SARS-CoV-2 and 1000

Table 2: Protein characteristic based on Discere.

Feature no.	Characteristics	Number of features
1–20	AA-count	20
21	Aromaticity	1
22–24	Secondary structure fraction	3
25	Isoelectric point	1
26	Molecular weight	1
27	Instability index	1
Total		27

protein sequences of non-SARS-CoV-2. By balancing the dataset, we aimed to prevent the model from being biased toward the majority class (non-SARS-CoV-2) and to give equal consideration to both classes during the training process. This approach ensures that the model can effectively learn patterns and features from both classes, leading to a fair and accurate evaluation of its performance. The protein sequence data of SARS-CoV-2 was obtained from UniProt's website on June 8, 2021, in FASTA format.

2.2. Feature Extraction. Feature extraction is a crucial step in preparing protein sequence data for machine learning and deep learning tasks. The Discere package is a Python code designed to facilitate the extraction of informative features from protein sequences, making them suitable for predictive modeling and analysis. The feature extraction process involves calculating various characteristics and properties of the protein sequences. These features provide important information about the sequences and enable ML/DL models to learn patterns and relationships for classification or regression tasks.

The Discere package utilizes the Biopython library, a powerful and widely used library in bioinformatics, to handle protein-related data and perform specific operations on protein sequences. Biopython offers various functionalities, such as sequence parsing, motif finding, and biochemical property calculations, which are essential for protein feature extraction. The extracted features from protein sequences using Discere include the following:

(i) *AA-count (amino acid count)*: this feature calculates the number of occurrences of each type of amino acid in the protein sequence. Different amino acids play critical roles in protein structure and function, and their frequency can carry valuable information

(ii) *Aromaticity*: aromaticity measures the proportion of aromatic amino acids (e.g., phenylalanine, tyrosine, and tryptophan) present in the protein sequence. Aromatic amino acids are involved in various biological processes, and their presence can be indicative of certain protein functions

(iii) *Secondary_structure_fraction*: this feature quantifies the fraction of each secondary structure element (e.g., alpha-helix, beta-sheet, and coil) within the protein sequence. Secondary structures play a crucial role in protein folding and function

(iv) *Isoelectric_point*: the isoelectric point is the pH at which a protein has no net charge. It is an important property for understanding protein behavior in different environments

(v) *Molecular_weight*: this feature calculates the molecular weight of the protein sequence. Molecular weight is related to protein size and can influence various biological processes

(vi) *Instability_index*: the instability index assesses the stability of a protein based on its amino acid composition. It helps predict whether a protein is stable or prone to degradation

The package is designed to transform protein sequences into a set of 27 distinct features.

2.3. Feature Selection. In this study, the selection of the 27 features for protein sequences was a critical process aimed at identifying the most relevant and informative characteristics of the data. The significance of each selected feature was carefully evaluated to ensure its relevance to the classification task and its potential impact on machine learning algorithms. The feature selection stage is a crucial preprocessing step that helps reduce the dimensionality of the data and eliminate irrelevant or redundant features, which can have a negative impact on the performance of machine learning algorithms [10].

We employed three distinct feature selection methods to identify the most discriminative features for our classification task: genetic algorithm (GA), LASSO (least absolute shrinkage and selection operator), and SVM-RFE (support vector machine recursive feature elimination).

Genetic algorithms are optimization techniques inspired by the process of natural selection. They work by iteratively evolving a population of potential solutions through selection, crossover, and mutation. In the context of feature selection, GA searches for the best subset of features that maximize the performance of the machine learning model. LASSO is a linear regression technique that adds a penalty term to the regression equation, encouraging some regression coefficients to be exactly zero. In feature selection, LASSO helps to identify the most relevant features by shrinking the coefficients of less informative ones toward zero. SVM-RFE is an iterative feature selection method based on support vector machines. It recursively removes the least important features from the dataset until an optimal subset of features is identified, maximizing the SVM's classification performance.

Overall, each of the 27 features was carefully selected based on its potential to contribute meaningful information to the classification task and its relevance to understanding the characteristics of the protein sequences. By incorporating these selected features into the machine learning algorithms, we aim to enhance the performance and accuracy of the classification model for protein sequences.

2.3.1. Genetic Algorithm. Genetic algorithm is one of the feature selection methods with a wrapper technique that uses a small set of features to train the model. The basic idea of genetic algorithms is to manage a population of individuals and to obtain the best number of individuals (features) that represent candidate solutions to a problem where this system is, based on the principle of natural selection. This principle comes from the theory of evolution where individuals are constantly changing their genes to adapt to their environment or in other words, "Only strong individuals can survive." The process of natural selection involves changes in genes that occur in individuals through the process of reproduction, which ultimately results in the best offspring. From the principles of the theory of evolution, genetic algorithms can be used to solve problems in real life.

The genetic algorithm is described as follows [11]:

Step 1 (initialization). The first generation is formed by initializing the initial population.

Step 2 (evaluation). Each chromosome in the population is evaluated with the fitness function.

Step 3 (selection). The two best chromosomes (parent chromium) are selected for reproduction at the crossover stage.

Step 4 (crossover). Offspring chromosomes are obtained by exchanging bits from each pair of selected parental chromosomes.

Step 5 (mutation). The chromosomes of the offspring are selected to be mutated, then the mutation process is performed on the selected chromosome, and the fitness value of the chromosome is calculated.

Step 6 (formation of a new population). The fitness value of the offspring chromosome is compared with the fitness value of the parent chromosome. Chromosomes with higher fitness values will survive for the next generation, while other chromosomes will be discarded.

Step 7 (repetition of Steps 3 and 4). Steps 3 and 4 are repeated until they reach certain stopping criteria. The stopping criteria can be either a generation limit or a certain optimal value.

2.3.2. LASSO. Suppose given N sample with $\{(x_i, y_i)\}_{i=1}^{N}$, where $x_i = (x_{i1}, x_{i2}, \cdots, x_{ip})^T$, is a vector of features with p dimension, and y_i is a response variable of i. In the least square method, the estimator for (β_0, β) is based on minimizing squared error loss, like [12, 13]

$$\underset{\beta_0,\beta}{\text{minimize}} \left\{ \frac{1}{2N} \sum_{i=1}^{N} \left(y_i - \beta_0 - \sum_{j=1}^{p} x_{ij}\beta_j \right)^2 \right\}, \tag{1}$$

where $\beta = (\beta_1, \cdots, \beta_p) \in \mathbb{R}^p$ is a weighted vector of regression and a bias $\beta_0 \in \mathbb{R}$.

The LASSO method only adds the $\sum_{j=1}^{p} |\beta_j| \leq t$ to Equation (1). The $\sum_{j=1}^{p} |\beta_j| \leq t$ can be written as $\|\beta\|_1 \leq t$, where $\|\beta\|_1$ is a form of L1-norm of β. L1-norm is also known as the Manhattan distance [14]. The LASSO equation can be written as follows:

$$\underset{\beta_0,\beta}{\text{minimize}} \left\{ \frac{1}{2N} \sum_{i=1}^{N} \left(y_i - \beta_0 - \sum_{j=1}^{p} x_{ij}\beta_j \right)^2 \right\}, \quad \text{subject to } \|\beta\|_1 \leq t. \tag{2}$$

2.3.3. SVM-RFE. Support vector machine recursive feature elimination (SVM-RFE) is used during the feature selection process, and some features will be eliminated by first determining the number of most optimal features [15]. The main purpose of SVM-RFE is to compute the ranking weights for all features and sort the features according to weight vectors as the classification basis. Its steps for feature set selection are shown as follows:

Step 1. Use the current dataset to train the classifier.

Step 2. Compute the ranking weights for all features.

Step 3. Delete the feature with the smallest weight.

By using weight vectors from SVM, we can calculate the ranking criterion (c_i) value as the basis for ranking the feature as follows:

$$c_i = w_i^2, i = 1, 2, 3, \cdots, n, \tag{3}$$

with

$$w = \sum_{i=1}^{n} \alpha_i y_i x_i, \tag{4}$$

where x is the data value, y is the class label, α is the Lagrange coefficient, and w is the weight vectors.

2.4. Classification Methods. The three methods that we use in this research are K-nearest neighbor, Naïve Bayes, and extreme gradient boosting.

2.4.1. K-Nearest Neighbor. K-nearest neighbor (KNN) is a classification method for learning data objects that are closest to the object. KNN is one of the nonparametric methods used in the classification method. The KNN algorithm is simple, such that the algorithm works based on the shortest distance from the testing data to the training data by determining K closest neighbors from the testing data. The majority of the KNN obtained are used as predictions from the testing data. KNN has other terms such as lazy learning, instance-based learning, memory-based learning, and case-based learning [16]. The KNN algorithm has been used since 1970 in various applications such as statistical estimation and pattern recognition. The flowchart of KNN can be seen in Figure 1.

A simple calculation in the KNN method is the evaluation of the distance between the training data and the testing data. There are various methods of measuring the distance

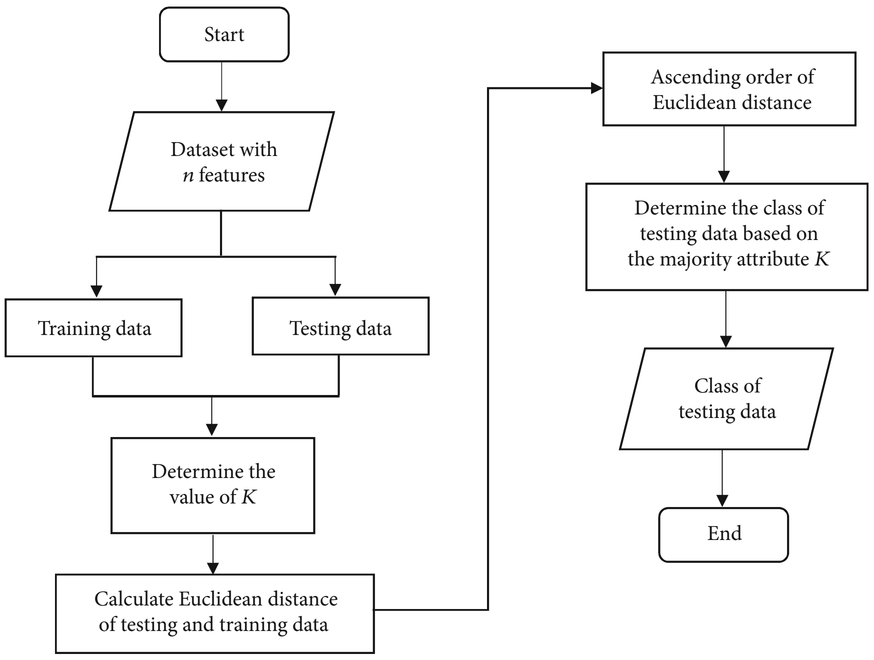

Figure 1: Flowchart of KNN. This diagram describes the KNN algorithm in detail.

```
**Input**: Dataset, the number of trees ($K$), the learning rate ($\eta$), the number of terminal nodes ($T$)
Initialize $\hat{y}_i^{(0)}$
**for** $k = 1, 2, \cdots, K$ **do**
Calculate $g_i^{(k)} = \partial_{\hat{y}_i^{(k-1)}} l(y_i, \hat{y}_i^{(k-1)})$
Calculate $h_i^{(k)} = \partial^2_{\hat{y}_i^{(k-1)}} l(y_i, \hat{y}_i^{(k-1)})$
Determine the structure $\{\hat{R}_{jk}\}_{j=1}^{T}$ by choosing split node with maximized *Gain*:
        $Gain = 1/2[((\sum_{i\in I_L} g_i)^2/\sum_{i\in I_L} h_i + \lambda) + ((\sum_{i\in I_R} g_i)^2/\sum_{i\in I_R} h_i + \lambda) - (\sum_{i\in I} g_i)^2/\sum_{i\in I} h_i + \lambda] - \gamma$
Determine the leaf weights $\{\omega_{jk}\}_{j=1}^{T}$, $\omega_{jk} = -\sum_{i\in I_j} g_i^{(k)}/\sum_{i\in I_j} h_i^{(k)} + \lambda$
Determine the decision tree $f_k(x_i) = \eta\sum_{j=1}^{T} \omega_j I[x \in \hat{R}_{jk}]$
  Add trees $\hat{y}_i^{(k)} = \hat{y}_i^{(k-1)} + f_k(x_i)$
**end**
**Output**: Prediction value $\hat{y}_i = \hat{y}_i^{(K)} = \sum_{k=1}^{K} f_k(x_i)$.
```

Algorithm 1: The algorithm of XGBoost.

between two objects with several attributes or features. Each measurement has three conditions [16]. Let $d(A, B)$ be the distance between two points A, B; then,

(i) $d(A, B) \geq 0$ and $d(A, B) = 0$ if only if $A = B$

(ii) $d(A, B) = d(B, A)$

(iii) $d(A, C) \leq d(A, B) + d(B, C)$

The third rule is also called triangular inequality. The three provisions state that the shortest distance between two points is a representation of a straight line. In general, the distance measurement used in KNN is the Euclidean distance measurement. According to the Euclidean distance formula given by Equation (5), for two sets of points x and y, given the number of features, the distance between the two points x and y is

$$d(x, y) = \sqrt{\sum_{k=1}^{n} (x_k - y_k)^2}, \tag{5}$$

where $x_k$and y_k are successively the k-th feature.

2.4.2. Naïve Bayes. Naïve Bayes is a classification method that is based on Bayesian theory. The Naïve Bayes method assumes that all features are independent, which is called class-conditional independence [17]. This assumption is not always true, but Naïve Bayes method often works well

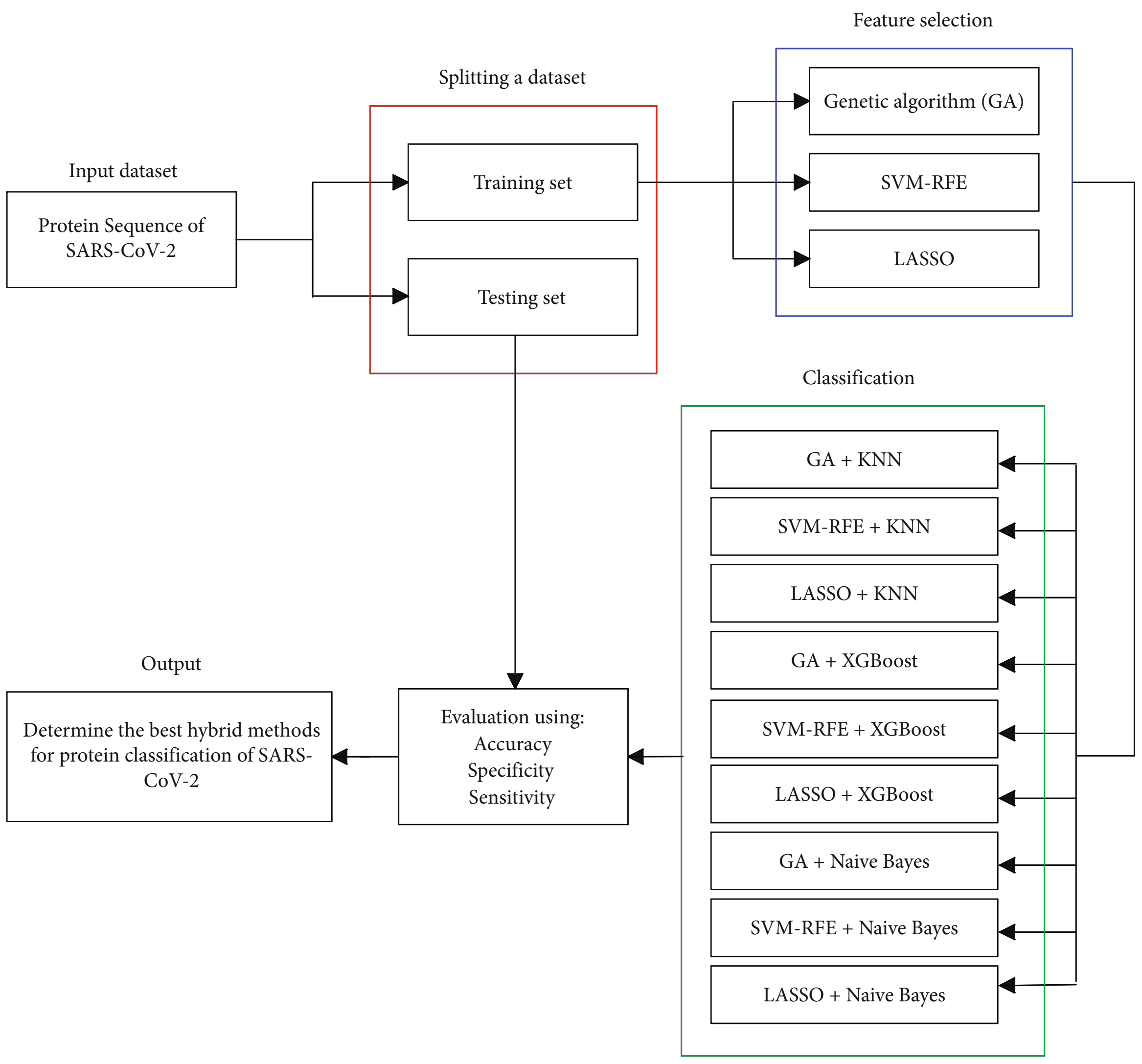

Figure 2: Stages of research. This diagram describes the flow of research from data input, steps of process, and output results.

Table 3: Feature subset and fitness function using genetic algorithm.

Running	Feature subset	Total number of features	Fitness value (%)
1	1, 2, 7, 10, 11, 13, 18, 19, 21, 22, 24, 25	12	93.85
2	7, 8, 10, 11, 12, 17, 19, 22, 24, 25, 26, 27	12	93.9
3	2, 7, 10, 11, 12, 18, 19, 25, 26, 27	10	94.3
4	2, 4, 7, 10, 11, 14, 19, 23, 24, 25	10	93.85
5	1, 7, 10, 11, 13, 14, 17, 19, 23, 25	10	94.25
6	7, 8, 10, 11, 14, 17, 18, 22, 25, 26, 27	11	93.95
7	5, 7, 10, 11, 16, 17, 22, 25, 26	9	94.15
8	1, 2, 4, 7, 8, 10, 11, 12, 14, 16, 22, 24, 25	13	93.9
9	**2, 4, 7, 10, 11, 17, 22, 23, 24, 25, 27**	**11**	**94.5**
10	2, 4, 5, 7, 10, 11, 15, 19, 25	9	94

We have selected the 11 features with the highest fitness scores, which are indicated in boldface.

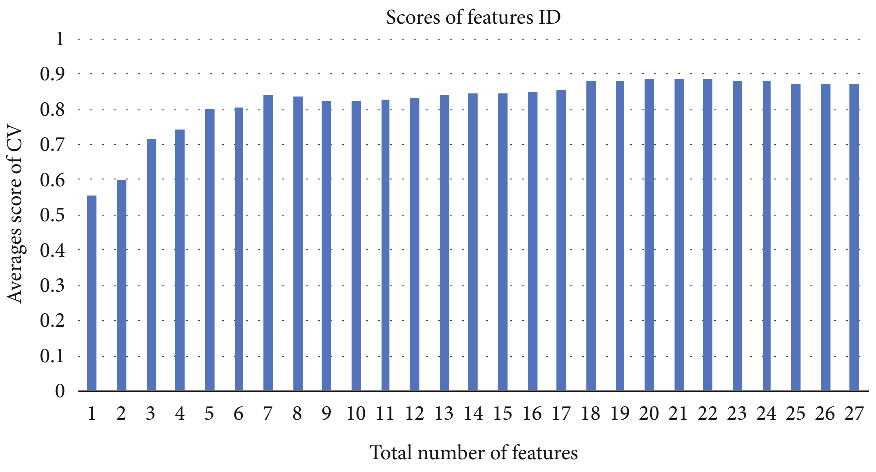

FIGURE 3: Cross-validation feature summary. This diagram describes the relationship between the number of features and the average score of cross-validation, providing insights into the impact of feature selection on model performance.

TABLE 4: Protein characteristic based on Discere.

Parameter	Value
learning rate (eta)	5
max_depth	1
min_child_weight	1.00
gamma (min split loss)	0.35
subsample	0.00
colsample_bytree	0.01
alpha	0.08
lambda	5

[18]. In Naïve Bayes, we will calculate posteriori probability with the following:

$$P(C|F_1, F_2, \cdots, F_n) = \frac{1}{Z} P(C) \prod_{i=1}^{n} P(F_i|C), \tag{6}$$

where $F_1, F_2, \cdots, F_n$ is an n feature that is independent for possible number of outcomes or classes C. Z is a scalar factor that is dependent on features $F_1, F_2, \cdots, F_n$. $P(C)$ is called prior probability of class C, while $P(C|F_1, F_2, \cdots, F_n)$ is a posteriori probability, and $P(F_i|C)$ is a conditional probability of F_i, given class C. The value of $P(F_i|C)$ can be calculated using the Gaussian distribution according to the following:

$$P(F_i|C_k) = \frac{1}{\sigma\sqrt{2\pi}} e^{(F_i-\mu)^2/2\sigma^2}, \tag{7}$$

where μ is the mean and σ is a standard deviation.

2.4.3. Extreme Gradient Boosting. Extreme gradient boosting (XGBoost) is an optimized distributed gradient boosting library designed to be highly efficient, flexible, and portable [19]. It implements machine learning algorithms under the gradient boosting framework. XGBoost provides a parallel tree boosting that solves many data science problems in a fast and accurate way. The basic idea of boosting is combining hundreds of simple trees with low accuracy to build a more accurate model. Every iteration will generate a new tree for the model. The value of the prediction function can have different interpretations depending on the program being run, that is, regression or classification.

An objective function usually has a form that contains two parts, training loss and regularization [20]. The objective function in XGBoost is defined as

$$\text{obj}(\theta) = \sum_{i=1}^{n} L(\theta) + \sum_{i=1}^{t} \Omega(\theta_i), \tag{8}$$

where L is the training loss function and Ω is the regularization term. The training loss measures how predictive our model is with respect to the training data. A common choice of L is the mean squared error, which is given by

$$L(\theta) = \sum_{i=1}^{n} \left(y_i - \hat{y}_i^{(t)}\right)^2, \tag{9}$$

where y is the class label and $\hat{y}$ is the predicted value of y. The algorithm of XGBoost is given as follows.

The research is aimed at classifying the coronavirus protein sequence and performing an evaluation based on the result. We simulated nine models by combining three different feature selection methods (genetic algorithm, SVM-RFE, and LASSO) with three classification methods (KNN, XGBoost, and Naïve Bayes). Each model represents a specific combination of a feature selection method and a classification method. By simulating and evaluating these nine models, we aim to identify the most suitable combination of feature selection and classification methods to achieve the highest classification accuracy and robustness in predicting the presence of SARS-CoV-2 protein sequences.

Figure 2 shows the stages of this research. Protein sequence data is the input. Then, the data must be extracted to numerical data. We implement the Discere

Table 5: Average accuracy of different models for different training percentages.

Training percentage	GA +KNN	SVM-RFE +KNN	LASSO +KNN	GA+Naïve Bayes	SVM-RFE +Naïve Bayes	LASSO +Naïve Bayes	GA +XGBoost	SVM-RFE +XGBoost	LASSO +XGBoost
60%	96.50%	98.55%	97.29%	81.21%	80.85%	79.37%	93.91%	97.25%	93.66%
70%	96.30%	98.99%	96.92%	81.55%	81.32%	80.35%	93.15%	97.33%	93.04%
80%	96.50%	99.19%	96.99%	82.42%	81.45%	81.44%	93.33%	97.50%	92.12%
90%	97.50%	**99.30%**	96.54%	82.19%	81.07%	80.03%	92.87%	96.00%	92.10%

Highest accuracy achieved at a training percentage of 90% using the SVM-RFE+KNN model, reaching 99.30%, as observed from the boldfaced entries.

Table 6: Average specificity of different models for different training percentage.

Training percentage	GA +KNN	SVM-RFE +KNN	LASSO +KNN	GA+Naïve Bayes	SVM-RFE +Naïve Bayes	LASSO+Naïve Bayes	GA +XGBoost	SVM-RFE +XGBoost	LASSO +XGBoost
60%	98.79%	98.93%	96.90%	90.65%	90.41%	87.72%	92.12%	98.01%	94.16%
70%	98.38%	98.40%	96.61%	91.38%	90.65%	87.98%	95.24%	98.01%	94.44%
80%	99.01%	99.34%	97.01%	91.51%	91.40%	88.24%	93.11%	98.08%	95.05%
90%	99.04%	**99.52%**	96.87%	91.90%	91.76%	88.16%	93.44%	96.08%	93.84%

Highest specificity achieved at a training percentage of 90% using the SVM-RFE+KNN model, reaching 99.52%, as observed from the boldfaced entries.

Table 7: Average sensitivity of different models for different training percentage.

Training percentage	GA +KNN	SVM-RFE +KNN	LASSO +KNN	GA+Naïve Bayes	SVM-RFE +Naïve Bayes	LASSO+Naïve Bayes	GA +XGBoost	SVM-RFE +XGBoost	LASSO +XGBoost
60%	94.04%	98.99%	97.61%	71.87%	75.12%	71.22%	91.16%	96.48%	92.15%
70%	94.13%	99.34%	97.29%	71.45%	75.79%	72.53%	93.75%	96.64%	93.83%
80%	93.87%	99.51%	96.97%	71.93%	76.15%	74.58%	92.56%	96.88%	93.67%
90%	95.78%	**99.55%**	96.17%	72.07%	76.46%	71.97%	92.91%	95.92%	93.34%

Highest sensitivity achieved at a training percentage of 90% using the SVM-RFE+KNN model, reaching 99.55%, as observed from the boldfaced entries.

package for feature extraction. After applying feature extraction, feature selection is conducted by three models separately: genetic algorithm, SVM-RFE, and LASSO. The performance of the model is observed after the feature is removed. After the features are selected, the next step is data classification.

In this study, we used varying percentages of the total dataset for training the machine learning model. Specifically, we fit the model using 60%, 70%, 80%, and 90% of the data as training sets. The remaining percentage was used as testing data to evaluate the model's performance. Using varying proportions of data for training and testing allowed us to investigate how the model's performance changes with different dataset sizes and determine the optimal amount of training data required for effective generalization. We assess the performance of the model using three important metrics: accuracy, specificity, and sensitivity.

Our proposed method is a comprehensive approach for classifying protein sequences to distinguish between SARS-CoV-2 and non-SARS-CoV-2 cases. The method involves several key steps, including feature extraction, feature selection using genetic algorithm (GA), LASSO (least absolute shrinkage and selection operator), and SVM-RFE (support vector machine recursive feature elimination), and the application of different classification algorithms such as K-nearest neighbor (KNN), XGBoost, and Naïve Bayes.

3. Results and Discussion

In this method of classification, performance measurement models can be incorporated by determining the matrix of confusion (confusion matrix), as shown in Table 1. According to Wu and Kumar [3, 21], confusion matrix is a tabulation where the actual class of various sample data (rows) is compared with the predicted data (columns). True positive (TP) is the actual data of the positive class that is correctly predicted as the positive class. False negative (FN) is actual data from the positive class that is predicted as the negative class. False positive (FP) is actual data from the negative class that is predicted as the positive class. True negative (TN) is the actual data of the negative class that is correctly predicted as the negative class. Evaluation on the formation of the classification model can be obtained from the confusion matrix by calculating several performance measurements, namely, accuracy, specificity, and sensitivity [22, 23].

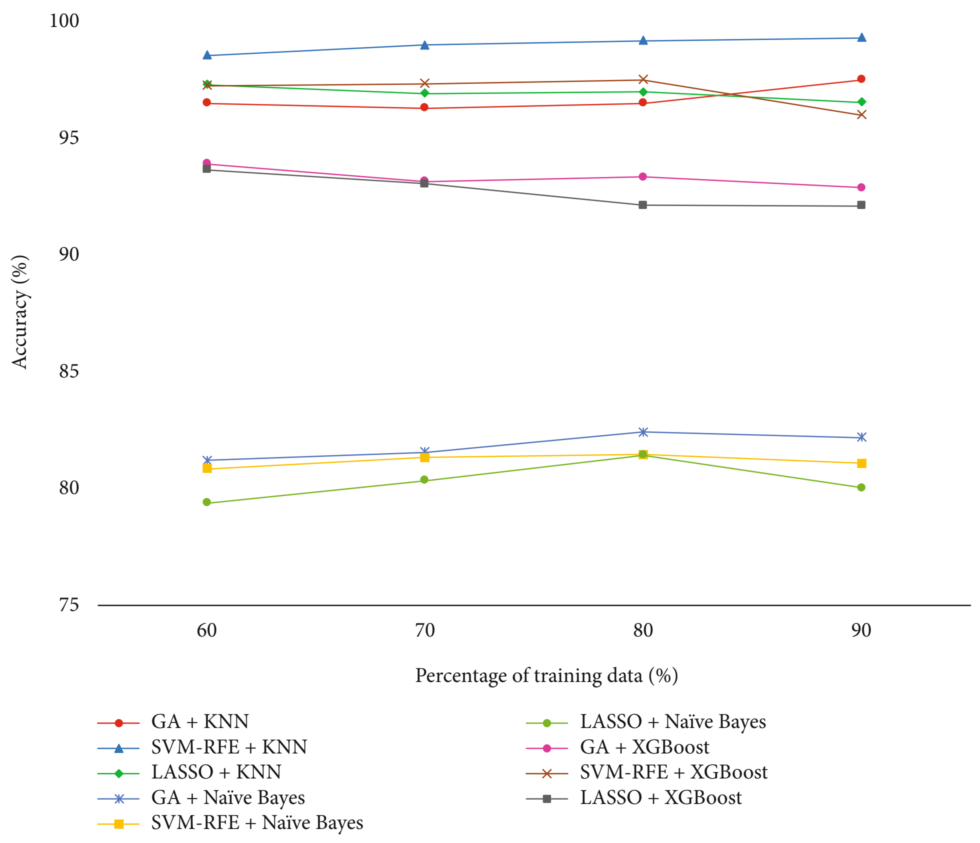

Figure 4: Comparison chart of accuracy. This figure describes the trend of the change of accuracy rate with the percentage of training data used.

$$\begin{aligned} \text{Accuracy} &= \frac{TP + TN}{TP + TN + FP + FN} \times 100\%, \\ \text{Specificity} &= \frac{TN}{TN + FP} \times 100\%, \\ \text{Sensitivity} &= \frac{TP}{TP + FN} \times 100\%. \end{aligned} \tag{10}$$

Accuracy is a measure that describes how precise the model is in classifying data. Specificity is a measure that describes how accurately the model classifies the negative class correctly. Sensitivity is a measure that describes how accurately the model classifies positive classes correctly.

The features in the protein sequence data used were extracted using the Python program Discere package [24]. This package will extract protein sequence into 27 features. The characteristics of protein based on Discere are shown in Table 2.

The results of the genetic algorithm feature selection are obtained from 10 running attempts since the random aspects of the genetic algorithm need to be considered. The average results obtained are 11 features. The feature subset used for classification in the KNN method is the feature that has the highest fitness value. It consists of features 2, 4, 7, 10, 11, 17, 22, 23, 24, 25, and 27, as shown in Table 3. These features are obtained from the characteristics of AA-count, secondary structure fraction, isoelectric point, and instability index.

In LASSO, the parameters used are selected from the interval [0.0001, 1]. Parameters will be repeated with 80% training data until the highest accuracy results are obtained, so that the obtained parameter value is 0.0533, with 4 features selected. The features are 18, 21, 22, and 26. The parameter used for the LASSO method is 0.0533.

In the recursive feature elimination (RFE) process, SVM models will be used. Using cross-validation, this process will try to eliminate the feature one by one from the importance level of the feature and calculate the average cross-validation value for each elimination [15]. Based on the selection process of these features, the optimal number of features is 18 for this data (as shown in Figure 3). The features are 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 13, 14, 17, 18, 19, 22, and 23.

To enhance the performance of XGBoost, there are several main parameters to be tuned to obtain a good model for this protein sequence classification problem. The parameters are the following:

(i) *learning rate* (*eta*): step size shrinkage employed to prevent overfitting. We shrink the feature weights to make the boosting process more conservative

(ii) *max_depth*: maximum depth of a tree; increasing this value will make the model more complex

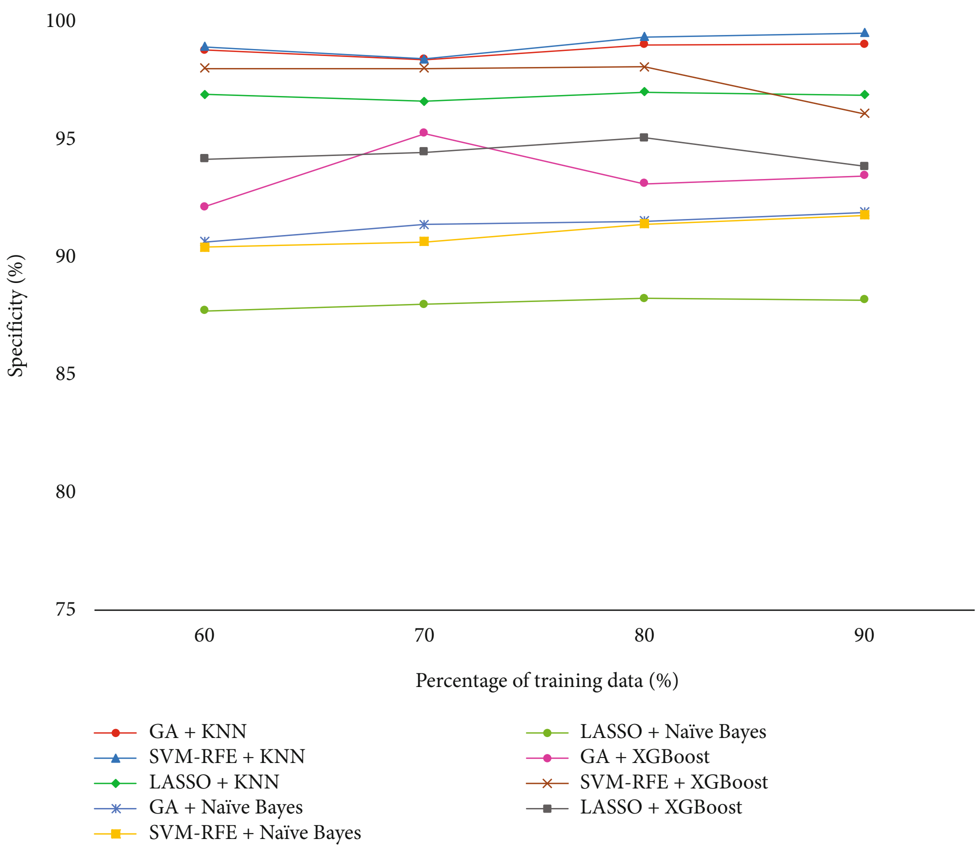

Figure 5: Comparison chart of specificity. This figure describes the trend of the change of accuracy rate with the percentage of training data used.

(iii) *min_child_weight*: minimum sum of instance weight needed in a child

(iv) *gamma*: minimum loss reduced required to make a further partition on a leaf node of the tree

(v) *subsample*: subsample ratio of the training instance

(vi) *colsample_bytree*: subsample ratio of features when constructing each tree

(vii) *alpha*: regularization term on XGBoost

(viii) *lambda*: regularization term on XGBoost

Grid search is a typical approach for parameter tuning; it methodically builds and evaluates a model for each combination of parameters in a specific grid. Our flow for optimizing parameters started from tuning tree-based parameters such as max_depth and min_child_weight; calibrating gamma, subsample, and colsample_bytree; balancing alpha and lambda; and, lastly, reducing the learning rate. The reason for such a flow is because the nature of XGBoost is robust enough not to be overfitting with increasing trees, but a high value for a particular learning rate could degrade its ability in predicting new test data. The hyperparameter used in XGBoost can be seen in Table 4.

The test results sought are the results of the classification of coronavirus protein sequence data using KNN with genetic algorithm feature selection, Naïve Bayes with LASSO feature selection, and XGBoost with SVM-RFE feature selection. The implementation requires 10 running attempts on the data for each model. Table 5 shows the result of accuracy from each model. Based on the results, SVM-RFE+KNN model achieved the best accuracy of 99.30%, with 90% of training data and parameter values $K = 3$ for the KNN method. From Tables 6 and 7, the best accuracy and sensitivity obtained are 99.52% and 99.55%, respectively, which are achieved by the SVM-RFE+KNN model with 90% of training data. The average accuracy, specificity, and sensitivity of different models for different training percentage can be seen in Figures 4–6, respectively. Therefore, based on our simulations, SVM-RFE+KNN is the best hybrid method to determine whether a protein sequence is SARS-CoV-2 or not. The experimental results show that KNN and XGBoost classification algorithms give nearly similar in accuracy, specificity, and sensitivity (greater than 90%). Furthermore, the average running time of the XGBoost and KNN classification method is shown in Table 8. The running time difference between XGBoost and KNN method is not too significant. The running time average of XGBoost is 1.443 seconds and KNN is 0.057 seconds.

Despite our best efforts, there are several potential limitations that should be acknowledged. Addressing these

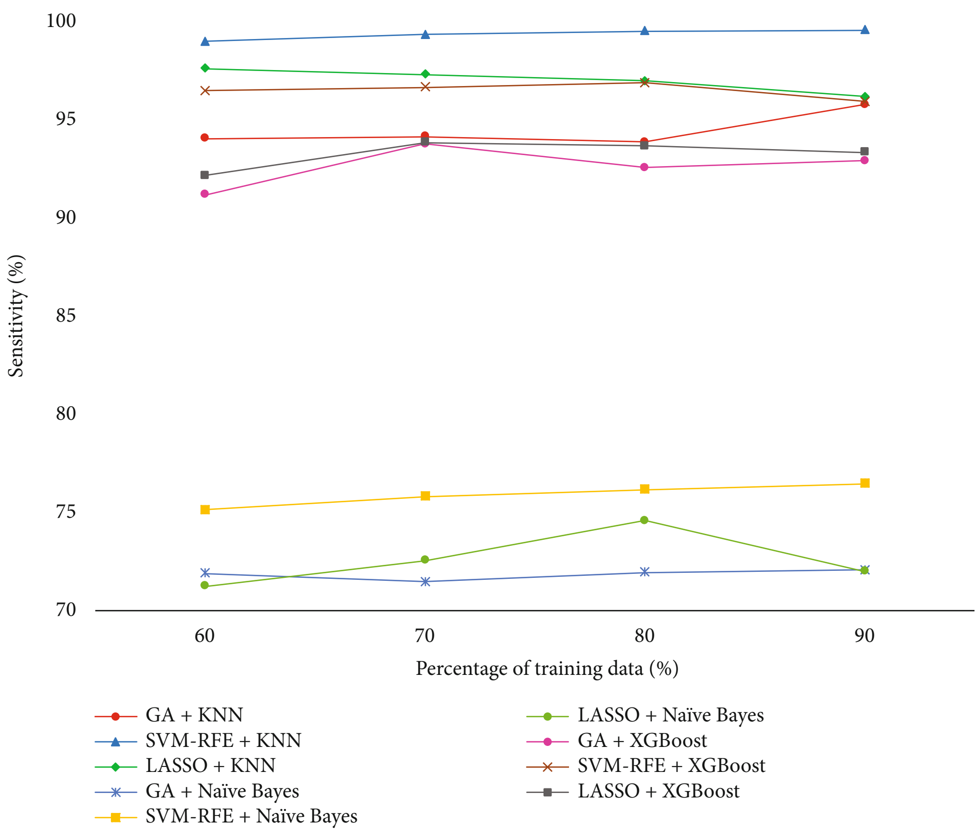

FIGURE 6: Comparison chart of sensitivity. This figure describes the trend of the change of accuracy rate with the percentage of training data used.

TABLE 8: Average running time of XGBoost and KNN.

Training data percentage	XGBoost running time (seconds)	KNN running time (seconds)
60%	1.45	0.049
70%	1.51	0.053
80%	1.47	0.062
90%	1.34	0.065
Average	**1.443**	**0.057**

The average runtime of KNN is faster compared to XGBoost, as observed from the boldfaced entries.

TABLE 9: A comparison with state of the art.

Method	Accuracy (%)	Specificity (%)	Sensitivity (%)
Proposed method (SVM-RFE+KNN)	99.30	99.52	99.55
Georganos et al. (XGBoost) [6]	79.8	—	—
Arian et al. (KNN) [5]	90.0	—	—
Nitta et al. (Naïve Bayes) [9]	97.47	92.89	86.83

limitations in future research can further enhance the validity and applicability of our findings. The dataset size of 1000 protein sequences for each class (SARS-CoV-2 and non-SARS-CoV-2) might be relatively small, which could impact the model's ability to generalize well. Future research could consider using larger and more diverse datasets to improve the model's performance and generalization capabilities. Although we balanced the dataset for this study, imbalances in real-world datasets can pose challenges. Exploring advanced techniques like data augmentation, synthetic sample generation, or leveraging specialized algorithms for handling imbalanced data can be beneficial. Table 9 provides a comprehensive comparison with state-of-the-art methods.

While we employed three feature selection methods, there might be other advanced techniques to explore in future research. Evaluating additional feature selection algorithms or dimensionality reduction methods could potentially yield more discriminative and informative features. In this study, we used grid search for hyperparameter tuning. Future research could investigate more sophisticated hyperparameter optimization methods like Bayesian optimization, which may yield better-tuned models and further enhance performance. Although our models achieved high accuracy, sensitivity, and specificity, the biological interpretability of the selected features and the underlying reasons for their importance could be explored further in future research. Understanding the biological significance of these features could provide deeper insights into the classification process. Since this study utilized the Discere package for feature extraction, it is important to consider the limitations of the package. Future research could explore alternative feature extraction techniques and assess their impact on the model's performance.

4. Conclusions

In this study, we proposed a classification approach for distinguishing between SARS-CoV-2 and non-SARS-CoV-2 protein sequences. The combination of feature selection methods, including genetic algorithm, LASSO, and SVM-RFE, with classification algorithms K-nearest neighbor (KNN), XGBoost, and Naïve Bayes, yielded promising results. The nine simulated models, encompassing various feature selection and classification techniques, demonstrated high accuracy, sensitivity, and specificity.

The feature selection stage revealed the optimal number of features for each method, with genetic algorithm selecting 11 features, LASSO selecting 4 features, and SVM-RFE selecting 18 out of 27 features. These findings indicate the relevance and significance of the selected features in distinguishing between the two classes of protein sequences.

The SVM-RFE+KNN model emerged as the top-performing model, achieving an impressive classification accuracy of 99.30%, specificity of 99.52%, and sensitivity of 99.55%. This high accuracy showcases the effectiveness of the proposed classification approach for COVID-19 protein sequence data.

Our research contributes to the field by providing valuable insights into the application of feature selection and classification algorithms for COVID-19 protein sequence classification. The proposed hybrid methods offer a robust and accurate means of identifying SARS-CoV-2 protein sequences, which can be pivotal in predicting protein sequences of newly emerging viruses.

Practically, our study holds several advantages, including the ability to swiftly detect pathogens and facilitate the development of targeted treatments for COVID-19. The models' high sensitivity ensures early detection, which can aid in timely interventions and reduce the spread of the virus. Moreover, the approach's computational efficiency can be advantageous for real-time applications and large-scale sequence analysis.

However, we acknowledge certain limitations in our study. The dataset size, though balanced, may not fully represent the vast diversity of protein sequences, potentially affecting the model's generalizability. Additionally, while we explored multiple feature selection and classification algorithms, other advanced techniques might yield further improvements.

For future research, we propose exploring hybrid methods to predict the biological sequence's nature, not limited to SARS-CoV-2, and enhancing performance while minimizing computation time. Further investigation into the biological significance of the selected features can improve our understanding of protein sequence classification. Additionally, conducting external validations on diverse datasets from different sources will validate the models' robustness and applicability in real-world scenarios.

In conclusion, our study demonstrates a promising approach for classifying protein sequences related to COVID-19. The combination of feature selection and classification methods exhibits high accuracy and practical advantages for early pathogen detection and potential treatment development. While we acknowledge the limitations, further research can build upon this work to advance our understanding and application of machine learning techniques in virology and healthcare.

Data Availability

The FASTA data used to support the findings of this study are available from the corresponding author upon request.

Conflicts of Interest

The authors declare that there is no conflict of interest regarding the publication of this paper.

Acknowledgments

The authors would like to thank the Faculty of Mathematics and Natural Sciences, Universitas Indonesia, for their support on this research. This research was funded by the Faculty of Mathematics and Natural Sciences, Universitas Indonesia, under research grant NKB-031/UN2.F3/HKP.05.00/2021.

References

[1] Y. C. Wu, C. S. Chen, and Y. J. Chan, "The outbreak of COVID-19: an overview," *Journal of the Chinese Medical Association: JCMA*, vol. 83, no. 3, pp. 217–220, 2020.

[2] E. Karunapala, *Protein Function Prediction Using Machine Learning*, 2015, https://dl.ucsc.cmb.ac.lk/jspui/handle/123456789/3104.

[3] X. Wu and V. Kumar, *The Top Ten Algorithms in Data Mining*, Data Mining and Knowledge Discovery Series, Taylor & Francis Group, LLC, Minnesota, 2009.

[4] M. Zong, X. Zhu, and D. Cheng, "Learning k for kNN classification," *ACM Transactions on Intelligent Systems and Technology*, vol. 8, 2017.

[5] R. Arian, A. Hariri, A. Mehridehnavi, A. Fassihi, and F. Ghasemi, "Protein kinase inhibitors' classification using K-nearest neighbor algorithm," *Computational Biology and Chemistry*, vol. 86, p. 107269, 2020.

[6] S. Georganos, T. Grippa, A. N. Gadiaga et al., "Geographical random forests: a spatial extension of the random forest algorithm to address spatial heterogeneity in remote sensing and population modelling," *Geocarto International*, vol. 36, no. 2, pp. 121–136, 2021.

[7] B. Yu, W. Qiu, C. Chen et al., "SubMito-XGBoost: predicting protein submitochondrial localization by fusing multiple feature information and eXtreme gradient boosting," *Bioinformatics*, vol. 36, no. 4, pp. 1074–1081, 2020.

[8] C. C. Le, P. W. C. Prasad, A. Alsadoon, L. Pham, and A. Elchouemi, "Text classification: Naïve Bayes classifier with sentiment lexicon," *IAENG International Journal of Computer Science*, vol. 46, no. 2, pp. 141–148, 2019.

[9] G. R. Nitta, B. Y. Rao, T. Sravani, N. Ramakrishiah, and M. BalaAnand, “LASSO-based feature selection and Naïve Bayes classifier for crime prediction and its type,” *Service Oriented Computing and Applications*, vol. 13, no. 3, pp. 187–197, 2019.

[10] L. Dey, S. Chakraborty, and A. Mukhopadhyay, “Machine learning techniques for sequence-based prediction of viral-host interactions between SARS-CoV-2 and human proteins,” *Biomedical Journal*, vol. 43, no. 5, pp. 438–450, 2020.

[11] S. M. Chen and C. Y. Chien, “Solving the traveling salesman problem based on the genetic simulated annealing ant colony system with particle swarm optimization techniques,” *Expert Systems with Applications*, vol. 38, no. 12, pp. 14439–14450, 2011.

[12] V. Fonti and E. Belitser, “Feature selection using Lasso,” *VU Amsterdam Research Paper in Business Analytics*, vol. 30, pp. 1–25, 2017.

[13] T. Hastie, R. Tibshirani, and M. Wainwright, *Statistical Learning with Sparsity: The Lasso and Generalizations*, Chapman and Hall/CRC, 2015.

[14] N. Heslot, H.-P. Yang, M. E. Sorrells, and J.-L. Jannink, “Genomic selection in plant breeding: a comparison of models,” *Crop Science*, vol. 52, no. 1, pp. 146–160, 2012.

[15] M. L. Huang, Y. H. Hung, W. M. Lee, R. K. Li, and B. R. Jiang, “SVM-RFE based feature selection and Taguchi parameters optimization for multiclass SVM classifier,” *The Scientific World Journal*, vol. 2014, Article ID 795624, 10 pages, 2014.

[16] M. Akhil, B. L. Deekshatulu, and P. Chandra, “Classification of heart disease using K- nearest neighbor and genetic algorithm,” *Procedia Technology*, vol. 10, pp. 85–94, 2013.

[17] D. V. S. D. M Narasimha Murty, *Pattern Recognition: An Algorithmic Approach (1st Ed.)*, Springer-Verlag London, 2011.

[18] S. R. P. Norvig, *Artificial Intelligence: A Modern Approach (4th Ed.)*, Pearson, 2020.

[19] T. Chen and C. Guestrin, “XGBoost: a scalable tree boosting system,” *The Journal of the Association of Physicians of India*, vol. 42, no. 8, p. 665, 1994.

[20] L. Zhang and C. Zhan, “Machine learning in rock facies classification: an application of XGBoost,” in *International Geophysical Conference*, pp. 1371–1374, Qingdao, China, 2017.

[21] K. Kuzmin, A. E. Adeniyi, A. K. DaSouza Jr. et al., “Machine learning methods accurately predict host specificity of coronaviruses based on spike sequences alone,” *Biochemical and Biophysical Research Communications*, vol. 533, no. 3, pp. 553–558, 2020.

[22] A. W. Salehi, P. Baglat, and G. Gupta, “Review on machine and deep learning models for the detection and prediction of coronavirus,” *Materials Today: Proceedings*, vol. 33, pp. 3896–3901, 2020.

[23] Y. Li, Y. Yang, J. Che, and L. Zhang, “Predicting the number of nearest neighbor for kNN classifier,” *IAENG International Journal of Computer Science*, vol. 46, no. 4, pp. 662–669, 2019.

[24] E. C. Carbo, I. A. Sidorov, J. C. Zevenhoven-Dobbe et al., “Coronavirus discovery by metagenomic sequencing: a tool for pandemic preparedness,” *Journal of Clinical Virology*, vol. 131, p. 104594, 2020.