Computers, Materials & Continua
DOI: 10.32604/cmc.2022.030934
Article

Tech Science Press

Ensemble Machine Learning to Enhance Q8 Protein Secondary Structure Prediction

Moheb R. Girgis, Rofida M. Gamal and Enas Elgeldawi*

Computer Science Department, Faculty of Science, Minia University, 61519, Minia, Egypt
*Corresponding Author: Enas Elgeldawi. Email: enas.elgeldawi@mu.edu.eg
Received: 06 April 2022; Accepted: 11 May 2022

Abstract: Protein structure prediction is one of the most essential objectives practiced by theoretical chemistry and bioinformatics as it is of a vital importance in medicine, biotechnology and more. Protein secondary structure prediction (PSSP) has a significant role in the prediction of protein tertiary structure, as it bridges the gap between the protein primary sequences and tertiary structure prediction. Protein secondary structures are classified into two categories: 3-state category and 8-state category. Predicting the 3 states and the 8 states of secondary structures from protein sequences are called the Q3 prediction and the Q8 prediction problems, respectively. The 8 classes of secondary structures reveal more precise structural information for a variety of applications than the 3 classes of secondary structures, however, Q8 prediction has been found to be very challenging, that is why all previous work done in PSSP have focused on Q3 prediction. In this paper, we develop an ensemble Machine Learning (ML) approach for Q8 PSSP to explore the performance of ensemble learning algorithms compared to that of individual ML algorithms in Q8 PSSP. The ensemble members considered for constructing the ensemble models are well known classifiers, namely SVM (Support Vector Machines), KNN (K-Nearest Neighbor), DT (Decision Tree), RF (Random Forest), and NB (Naïve Bayes), with two feature extraction techniques, namely LDA (Linear Discriminate Analysis) and PCA (Principal Component Analysis). Experiments have been conducted for evaluating the performance of single models and ensemble models, with PCA and LDA, in Q8 PSSP. The novelty of this paper lies in the introduction of ensemble learning in Q8 PSSP problem. The experimental results confirmed that ensemble ML models are more accurate than individual ML models. They also indicated that features extracted by LDA are more effective than those extracted by PCA.

Keywords: Protein secondary structure prediction (PSSP); Q3 prediction; Q8 prediction; ensemble machine leaning; boosting; bagging

1 Introduction

Proteins are essential to life, as they perform an enormous range of functions. They act as enzymes for catalyzing biochemical reactions. The collagen protein, for example, maintains the form of a cell as it is the main component of human skin, while the actin-myosin protein complex plays an important role in muscle contraction and thus macroscopic movement in living organisms. Proteins can function as molecular switches, altering the state of another protein [1]. In short, proteins are of vital importance to almost every biological process. Proteins are composed of linear chains of amino acids, which are linked by peptide bonds. The levels of protein structures are: the primary, secondary, tertiary and quaternary structures. The protein tertiary (native) structure is particularly interesting as it describes the 3D structure of the protein molecule, which reveals the functions of proteins which in turn helps in drug design and protein engineering.

Protein structure prediction (PSP) is very important in bioinformatics, medicine, theoretical chemistry, biotechnology and more. From the early days of biochemistry, the biochemists crucial concern was to know the protein structure (PS). In 1943, Fredrick has developed a way to spot amino acid sequence present in insulin [2]. However, this significant discovery didn't indicate whether an individual protein has multiple structures. In 1954, Anfinsen has found that amino acid sequence of a protein could fold into a 3D structure [3]. He has also investigated the amino acid sequence refolding and noticed that protein unfolded under extreme chemical environment, could refold back to 3D structure. Anfinsen has introduced a theory of folding, and said in his Nobel peace prize ceremony: "The native structure of protein is decided by the whole interactions between atomics and thus by the sequence of amino acids, in certain environment". His theorem has provided an insight into the dilemma of PSP. The PSP terminology can be divided into two groups: (1) If proteins within the protein databank [4] have similarities in structure, then the target structure are often constructed by imitating the framework of the solved protein. This method is named template-based modeling [5]. However, this method cannot help in figuring out how proteins fold to their native structures. (2) If the protein structure has no similarities in the databank, the structure might be constructed from the amino acid sequence. This method is named ab initio [6], and it is the most difficult of all. This method depends on two components: energy function, which calculates the level of energy of every structure, and an enquiry procedure, which finds out the optimal structure with minimum energy within the search space. By the end of 2013, about 52 million-protein sequence were added to the protein sequence databank [7]. In addition, the amount of known PSs in the protein databank is about 90,000 [4].

The problem of the 3D structure prediction of a protein, based only on its primary structure (linear sequence of amino acids), is vitally important because the determination of protein structure experimentally is very expensive, while using the DNA sequencing to obtain protein sequences is cheap. Protein secondary structure is the most commonly predicted one-dimensional structure of proteins [8]. Protein secondary structure prediction (PSSP) has a significant role in the prediction of protein tertiary structure, as it bridges the gap between the protein primary sequences and tertiary structure prediction. There are three traditional experimental techniques employed to determine the secondary structure of proteins. These techniques are: X-ray crystallography, Nuclear Magnetic Resonance (NMR) spectroscopy, and Circular Dichroism spectrometry. As experimental methods are expensive and the number of proteins with known sequence is growing than the number of experimentally determined secondary structures, devising computational approaches for PSSP becomes increasingly urgent [9]. The prediction of protein secondary structure from only its amino acid sequence is a very difficult task. Protein secondary structures can be categorized into two classes: 3-state category, which includes: helix (H), strand (E), and coil (C); and 8-state category, which has been proposed by the DSSP (Define Secondary Structure of Proteins) program [10], includes: H (α-helix), G (3_{10} helix), I

(π-helix), S (bend), B (β-bridge), E (β-stand), T (β-turn), and L (loop or irregular). Protein sequence normally includes 2 kinds of sequence information: long-range interdependencies and local context [11,12]. The local contexts are essential for PSSP. Specially, the most valuable features to determine the secondary structure of an amino acid are the information about the secondary structure category of the neighbors of this amino acid. Also, long-range interdependency amongst different kinds of amino acids shows essential indicators for the type of a secondary structure. Predicting the 3 states and the 8 states of secondary structures from protein sequences are called the Q3 prediction and the Q8 prediction problems, respectively. Most of the methods of PSSP have been heavily concentrated on Q3 prediction. Although, Q8 prediction is complex and more challenging than Q3 prediction, in this paper, we focus on Q8 prediction, because the 8 classes of secondary structures reveal more precise structural information for a variety of applications than the 3 classes of secondary structures.

Ensemble learning (EL) techniques have shown great promise in improving the performance of classifier models in areas such as ML, data mining and pattern recognition. They possess a strength that has encouraged their use in several realistic classification problems. In fact, they supersede conventional ML methods in several applications and they have drawn the attention as a way to enhance the accuracy of prediction in highly complex problems. Many integration rules have been explored for developing EL techniques, but it is argued that no particular rule is better than others for devising an ideal decision [13]. EL has already been utilized in various applications such as face recognition, optical character recognition, gene expression analysis, computer-aided medical diagnosis, text categorization, etc. In fact, EL can be utilized anywhere ML methods can be utilized [14]. Ensemble learning methods, which have been broadly used for building improved classifier models, can be applied to PSSP [13,15–17]. In PSSP, the data (protein sequences) sequential nature demands particular ensembles. The known ensembles that depend on resampling or injecting randomness couldn't be used since the order of amino acids in protein sequences is important for consistent predictions and the success of prediction depends on the dependencies of amino acids. Thus, special devotion must be given to simple aggregation rule-based ensembles to fully investigate their ability to generalize in PSSP [13].

To this end, this paper presents a proposed ensemble ML approach for 8-state (Q8) PSSP. This approach employs two different ensemble methods, namely Bagging and Boosting. The ensemble members (base learners) considered for constructing the ensemble models are well known classifiers, namely Support Vector Machines (SVM), K-Nearest Neighbor (KNN), Decision Tree (DT), Naïve Bayes (NB), and Random Forest (RF), with two feature extraction techniques, namely PCA (Principal Component Analysis) and LDA (Linear Discriminate Analysis). Experiments have been conducted for evaluating the performance of single models as well as ensemble models, with PCA and LDA, in Q8 PSSP. To the best of our knowledge, the proposed approach is the first one to use EL for Q8 PSSP, as all the previous EL approaches have concentrated on Q3 prediction. This paper is organized as follows: related work is given in Section 2. Section 3 introduces the proposed ensemble machine learning approach for PSSP, and describes the methods and techniques employed in it, namely feature extraction techniques: LDA and PCA, the 5 ML algorithms: KNN, SVM, DT, RF, and NB, the two ensemble methods: Bagging and Boosting, and the models evaluation metrics. Section 4 presents the results of experiments that have been conducted to evaluate the presented models performance. Section 5 provides the conclusion of the work presented in this paper. Finally, future work is given in Section 6.

2 Related Work

Today, machine learning algorithms is integrated in almost every scientific discipline such as networking [18–20], text analysis [21,22], image processing [23–30], cloud computing [31] and social networks [32,33]. This broad range of machine learning application disciplines is due to their promising predictive performance.

Since the 1950s, several techniques were developed for the prediction of secondary structure from amino acid sequences. Techniques extend from fields such as physical chemistry and biology to computer science and statistics [34–36]. This section presents a review of previous related work that applied ensemble ML techniques in PSSP.

Bittencourt et al. [15] have presented an empirical comparison of the performance of individual ML techniques (SVM, NB, KNN, Neural Network (NN) and DT) and ensemble techniques (boosting and bagging) in protein structural class prediction. They considered the prediction of the following classes: all-α, all-β, $\alpha + \beta$, and α/β in addition to a class for the proteins that do not belong to any of the previous classes (small). The ensembles generated with bagging and boosting techniques utilized as base classifiers the single ML techniques DT, SVM and NN, and excluded KNN and NB, as the preliminary experiments showed that they are not suitable for the task. The experiments indicated that the results of ensemble techniques that utilized DT as the base classifier have shown consistent improvement.

Lin et al. [16] have applied a multi-SVM ensemble to improve the performance of PSSP. The SVM ensemble consisted of 2 layers; the first layer consisted of 3 SVMs network, where the winner is determined by winner-take-all; the second layer is a bagging ensemble network consisted of 5 classifiers, where its output is decided by majority voting. The classifiers were trained to predict three classes of PSS, which are H (helices), E (strands) and C (coil). The 7-fold cross-validation test on RS126 dataset showed that the multi-SVM ensemble achieved better Q3 accuracy of 74.98%.

Bouziane et al. [13] have investigated the effect of combining M-SVMs (Multi-class SVMs), KNNs, and ANNs to enhance globular proteins SSP. An ensemble technique that merges the outputs of 3 M-SVM, 2 Fead Forword NNs (FFNNs), and KNN classifiers has been used. Ensemble members are merged by 2 variations of the majority voting rule. To explore how much improvement the ensemble method can provide compared to the single classifiers that form the ensemble, the authors have applied the presented ensemble method on CB513 and RS126 datasets, by incorporating PSI-BLAST PSSM profiles as inputs. The experimental results indicated, in terms of Q3 accuracy, that performance of the proposed ensemble method was significantly better than that of the best single classifier. As can be seen from the above review, all the previous EL approaches have concentrated only on Q3 prediction.

3 Proposed Approach

This section describes the proposed approach for Q8 PSSP that uses ensemble machine learning to improve the accuracy of predicting protein secondary structures. The proposed approach to building an ensemble ML model for PSSP encompasses the following phases: (1) Data Preprocessing, (2) Features Extraction, (3) Data Splitting (Cross-validation method), (4) Ensemble Model Training, (5) Model Evaluation, and (6) Classification, as shown in Fig. 1. The phases of the proposed technique are explained in detail in the following subsections.

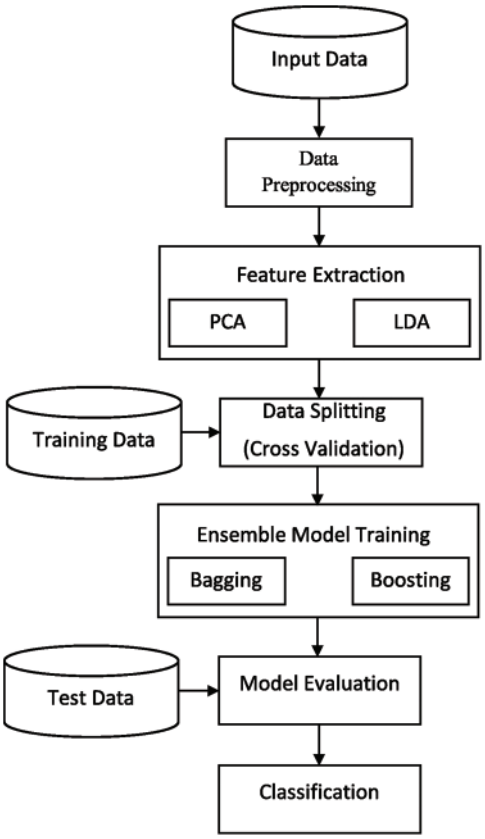

Figure 1: System model

3.1 Data Preprocessing

Given a set of proteins, the first step in PSSP is to convert the set of sequences from strings of alphabetic characters in capital letters, corresponding to the twenty naturally occurring amino acids, into a feature-based representation. The training dataset used is CB6133, which is produced by PISCES CullPDB server [37]. This dataset includes 6128 non-homologous sequences, each of 39900 features. In the 6128 proteins, 500 proteins are training samples. The dataset is available for public use through literature [11]. Protein sequences in CB6133 have a similarity less than 25%, a resolution better than 3.0Å and an R factor of 1.0. The redundancy with test datasets was removed using cd-hit [38]. The 500 proteins $\times$ 39900 features were reshaped into 500 proteins $\times$ 700 amino acids $\times$ 57 features. The test dataset used is CB513, which is obtained from [11]. It is broadly used to evaluate the PSSP methods performance. It consists of 26143 α-helix, 1180 β-bridge, 17994 β-strand, 3132 3_{10} helix, 30 π-helix, 10008 Turn, 8310 Bend, and 17904 Coil.

3.2 Feature Extraction

The extraction of the important features is an essential phase because irrelevant features often affect the classification efficiency of the ML classifier. The selection of features appropriately enhances classification accuracy and reduces the training time for the model. In the proposed ensemble ML approach for PSSP, features are extracted either by PCA (Principal Component Analysis) or LDA (Linear Discriminate Analysis) methods. These two techniques are described below.

3.2.1 Principal Component Analysis (PCA)

PCA is a multivariate technique. It is one of the most well-known techniques for the reduction of linear dimensionality [39]. PCA obtains the principal components in data by utilizing the covariance matrix and its eigenvectors and eigenvalues. These principal components are uncorrelated eigenvectors; each represents some percentage of variance in the data [40]. Let $X = \{x_1, x_2, \ldots, x_m\}$ denotes a set of training data, x_i represents a variable with dimensionality N, which stands for protein sequence in this work. The aim of PCA is two-fold: (a) get the highly significant information from X, and (b) reduce the dimensionality of X by preserving the significant information only. PCA is considered as an orthogonal projection of the initial N-dimensional data onto a new r-dimensional space ($r < N$), the projected data variance is the objective to be minimized [41]. The Initial letter of each notional word in all headings is capitalized.

3.2.2 Linear Discriminant Analysis (LDA)

LDA is a method for reducing dimensionality. It is utilized as a pre-processing stage in ML and pattern classification applications. LDA is primarily utilized in classification problems that have a categorical output variable. It can be used in binary and multi-class classifications. The LDA goal is to map the features in a high dimension space onto a low dimension space to overcome the dimensionality problem and decrease dimensional and resources costs [42]. The LDA model is comprised of the input data statistical attributes that have been computed for each class. In the case of having multiple variables, the same attributes are computed over the multivariate Gaussian. The predictions are made according to the probability that a new input dataset belongs to each class. The output class will be the one with the largest probability and accordingly the LDA does a prediction [42].

3.3 Data Splitting

In data splitting phase, the cross-validation technique is used to split the training data set into testing data and training data. Cross-validation is a standard method used to estimate the performance of any ML algorithm on unseen data. One cycle of cross-validation includes splitting a sample of data into disjoint subsets, doing the analysis on one subset (named the training set), then validating the analysis on the other subset (named the testing set or the validation set). In our experiments, we have used 9-fold cross-validation. In this case, the dataset is split into 9 subsets: the first 8 are utilized to train the model, and the 9th is utilized to validate the model. This operation is repeated, allowing each of the 9 subsets of the split dataset an opportunity to be the test subset.

3.4 Ensemble Model Training

The proposed ensemble machine learning approach for Q8 PSSP combines two of the widely known ensemble techniques, namely Boosting [43,44] and Bagging [45], with five ML algorithms, namely K-Nearest Neighbor (KNN) [45], Support Vector Machines (SVM) [46], Decision Tree (DT) [47], Random Forest (RF) [48], and Naïve Bayes (NB) [49], as base learners.

EL is a ML technique in which several learners are trained to solve the same problem. Unlike conventional ML techniques, which attempt to learn one hypothesis from training data, ensemble techniques attempt to build hypotheses set and merge them for usage [49]. The two ensemble techniques used in our approach, namely Boosting and Bagging, are described below.

Boosting: The term "boosting" refers to a group of techniques that can convert weak models to strong models. The weak model is the one that has a considerable error rate, but the performance is not random (causing an error rate of 0.5 in the case of binary classification). Boosting gradually develops

an ensemble by training each model with the same dataset but with adjusting the instances weights based on the last prediction error [38]. The basic concept is driving the models to concentrate on the instances that are misclassified. Then, the ensemble technique improves its efficiency by merging the trained weak models.

There are several variants of boosting. In this work, we use Adaptive Boosting (AdaBoost), which is a very famous boosting algorithm. It is proposed by Freund and Schapire [43]. Let X be the instance space, {–1, +1} be the set of class labels, and $D = \{(x_1, y_1); (x_2, y_2), \ldots, (x_N, y_N)\}$ is a given training data set, $x_i \in X$ and $y_i \in \{-1, +1\}$ $(i = 1, \ldots, N)$, $(i = 1, \ldots, N)$. AdaBoost works as follows: Firstly, it gives same weights to all the training instances. Let w_t denotes the weights distribution at the t-th learning round. From w_t and the training data set, the algorithm creates a base learner h_t: $X \rightarrow \{-1, +1\}$ by invoking the base learner. Then, it utilizes the training instances to test h_t, and increases the weights of the misclassified instances. Thus, a modified weight distribution w_{t+1} is gained. By invoking the base learner again with w_{t+1} and the training data set, AdaBoost creates another base learner. This process is iterated for M times, each iteration is named a round. The final learner is obtained by using weighted majority voting of the M base learners, where the learners' weights are established during the training process. Fig. 2 presents the pseudo-code of AdaBoost, which is adapted from [49].

Algorithm 1 AdaBoost algorithm /*for binary classification*/

Input: Dataset $Z = \{z_1, z_2, ..., z_n\}$, where
$z_i = (x_i, y_i)$, $x_i \in X$ and $y_i \in \{-1, +1\}$
M the maximum number of classifiers
C Base learner

Output: A classifier $H : X \rightarrow \{-1, +1\}$

Initialize the weights $w_i^{(1)} = 1/N, i \in \{1, ..., N\}$
for $m \leftarrow 1$ **to** M **do**
  Run C on Z, utilizing weights w_i, producing classifier $h_m : X \rightarrow \{-1, +1\}$
  Calculate $err_m = \sum_{i=1}^{n} w_i^{(m)} h(-y_i h_m(x_i))$, the weighted error of h_m.
  Calculate $\alpha_m = \frac{1}{2} \ln \frac{1-err_m}{err_m}$ /* Weak classifier weight h_m */
  foreach *Sample* $i = 1, ..., N$ **do**
    update the weights $v_i^{(m)} = w_i^{(m)} \exp(-\alpha_m y_i h_m(x_i))$
  end
  Renormalize the weights:
  calculate $S_m = \sum_{j=1}^{n} v_j$, and
  for $i \leftarrow 1$ **to** N **do**
    $w_i^{(m+1)} = \frac{v_i^{(m)}}{S_m}$;
  end
end
Generate the final classifier as a majority vote: $H(x) = sign(\sum_{j=1}^{m} \alpha_j h_j(x))$

Figure 2: AdaBoost algorithm

The function $h : R \rightarrow \{0, 1\}$ utilized in Algorithm 1 to calculate err_m, the weight error rate of the m^{th} classifier, is the Heaviside function, which is defined as:

$$h(x) = \begin{cases} 1 \; if \; x \geq 0 \\ 0 \; if \; x < 0 \end{cases} \tag{1}$$

Accordingly, since both $h_{m(x\_i)}$ and y_i take values in {–1, +1}, we have that

$$h(-y_i h_m(x_i)) = \begin{cases} 1 \; if \; y_i \neq h_m(x_i) \\ 0 \; if \; y_i = h_m(x_i) \end{cases} \tag{2}$$

It is possible to utilize any ML algorithm as a base classifier with boosting, if it allows weights on the training set.

Bagging: Bootstrap Aggregation, which is referred to as Bagging [43], is a method that utilizes bootstrap sampling to decrease the variance and/or enhance the accuracy of some predictor. It may be utilized in regression and classification. It trains a number of base learners each from a distinct bootstrap sample by invoking a base learning algorithm. A bootstrap sample is found by subsampling with replacement the training data set, where the sample size equals to the training data set size.

Consider a size-N dataset $Z = \{z_1, z_2, \ldots, z_N\}$, where $z_i = (\mathbf{x}_i, y_i)$, $\mathbf{x}_i \in X$ and y_i is a class label, in classification problems, or a real number, in regression problems. The main idea of bagging is to learn a set of B predictors (each from a bootstrap sample $Z_b^* \subseteq Z$, for $b = 1, \ldots, B$), then generate a final predictor by merging (majority voting, in classification, or by averaging, in regression) this group of predictors. The merging of several predictors reduces the expected error as it decreases the variance component of the bias-variance decomposition. The decrease of this variance component is proportionate to the classifiers number employed in the ensemble. The algorithm in Fig. 3 presents the bagging algorithm pseudo-code for binary classification, which is adapted from [49].

Algorithm 2 Bagging Algorithm /* for Binary Classification */

Input: Dataset $Z = \{z_1, z_2, ..., z_n\}$, where
$z_i = (x_i, y_i)$, $x_i \in X$ and $y_i \in \{-1, +1\}$
B number of bootstrap samples
Base learning algorithm C

Output: A final classifier $H : X \to \{-1, +1\}$

for $b \leftarrow 1$ **to** B **do**
  Pick, with replacement, N samples from Z, getting the $b-th$ bootstrap sample Z_b^*
  Run C on the bootstrap sample Z_b^*, yielding classifier $h_b : X \to \{-1, +1\}$.
end
Generate the final classifier as a majority vote of $h_1, ..., h_B$, that is, $H(x) = sign(\sum_{b=1}^{B} h_b(x))$

Figure 3: Bagging algorithm

It should be noted that, in the above description of bagging and boosting methods, binary classification is considered for simplicity, but in this work, they are applied to Q8 PSSP, which is a multi-class classification problem. The Q8 secondary structure classes are: 3_{10} helix (G), α-helix (H), π-helix (I), β-stand (E), β-bridge (B), β-turn (T), bend (S), and loop or irregular (L).

3.5 Model Evaluation

To assess the Q8 prediction performance of the presented ensemble models, we have used the following evaluation metrics: Accuracy, Recall, Precision, F-score, and AUC-ROC (Area Under the Curve - Receiver Operating Characteristics) curve.

4 Experiments

This section presents the results of the experiments that we have conducted to evaluate the performance of the 5 individual predicting models, constructed using the 5 ML algorithms, KNN, SVM, DT, RF, and NB, and the ensemble models, constructed by combining each of the ensemble ML methods, bagging and boosting with each of the 5 ML algorithms, and the two feature extraction techniques, PCA and LDA, using cross validation tests, in Q8 PSSP. The individual predicting models and the ensemble models are applied to both CB6133 training dataset and CB513 test data.

4.1 Features Extracted by PCA and LDA

The number of important features that were extracted by both PCA and LDA equal 56 features for each amino acid (see Tab. 1).

Table 1: Features extracted by both PCA and LDA with their scores. (Note that Feature 47 is not extracted among the 57 features of each amino acid, which represents the absolute solvent accessibility

Feature no.	PCA score	LDA score	Feature no.	PCA score	LDA score
Feature: 0	89.32225	94.32225	Feature: 29	90.15872	90.15872
Feature: 1	90.23555	93.23555	Feature: 30	84.36941	90.36941
Feature: 2	91.42222	93.42222	Feature: 31	84.8651	90.8651
Feature: 3	88	94	Feature: 32	89.2156	89.2156
Feature: 4	87.02111	94.02111	Feature: 33	89.32651	89.32651
Feature: 5	89.83333	94.83333	Feature: 34	89.96847	89.96847
Feature: 6	91.52344	93.52344	Feature: 35	89.75486	89.75486
Feature: 7	90.44331	93.44331	Feature: 36	89.21548	89.21548
Feature: 8	84.11222	94.11222	Feature: 37	89.66325	89.66325
Feature: 9	93.91335	93.91335	Feature: 38	87.56981	91.56981
Feature: 10	93.33581	93.33581	Feature: 39	85.86547	91.86547
Feature: 11	92.5478	94.5478	Feature: 40	86.67891	91.67891
Feature: 12	92.15872	94.15872	Feature: 41	87.65874	91.65874
Feature: 13	92.20075	94.20075	Feature: 42	90.00001	90.00001
Feature: 14	92.22368	93.22368	Feature: 43	85.0021	91.0021
Feature: 15	92.13212	94.13212	Feature: 44	89.01203	89.01203
Feature: 16	92.91234	93.91234	Feature: 45	89	96
Feature: 17	90.36941	94.36941	Feature: 46	87.32015	95.32015
Feature: 18	86.81444	93.81444	Feature: 48	88.89521	88.89521
Feature: 19	88.98741	92.98741	Feature: 49	86.32104	88.32104
Feature: 20	89.66621	93.66621	Feature: 50	89.00021	89.00021
Feature: 21	90.75841	93.75841	Feature: 51	88.33215	96.33215
Feature: 22	90.89213	92.89213	Feature: 52	90.56841	90.56841
Feature: 23	90.32154	94.32154	Feature: 53	86.38749	90.38749
Feature: 24	88.55521	93.55521	Feature: 54	86.21213	90.21213
Feature: 25	86.68749	92.68749	Feature: 55	86.68749	91.68749
Feature: 26	85.75846	92.75846	Feature: 56	86.87932	91.87932
Feature: 27	84.21354	92.21354	Feature: 57	0.300121	0.984712
Feature: 28	90.23651	90.23651			

Tab. 1 shows the scores given to every feature, by PCA, according to the projected features variance, and by LDA, based on the distance between features. Among these 56 features, 22 represent the primary structure (20 amino acid, 1 unknown or any amino acid, 1 'NoSeq' - padding), 22 represent

the Protein Profile (same as primary structure), 2 represent N- and C-terminals, 1 represents relative solvent accessibility and 9 represent the secondary structure (8 possible states, 1 'NoSeq' - padding).

4.2 Experimental Results

We have conducted 6 different experiments: In the first two experiments, we have applied the 5 ML algorithms to features extracted by PCA and LDA, respectively. In the next two experiments, we have applied the ensemble models, constructed by combining the bagging technique with each of the 5 ML algorithms, to features extracted by PCA and LDA, respectively. In the last two experiments, we have applied the ensemble models, constructed by combining the boosting technique (AdaBoost) with each of the 5 ML algorithms, to features extracted by PCA and LDA, respectively.

4.2.1 Experiment 1: Applying the ML Algorithms to Features Extracted by PCA

In the first experiment, the 5 ML algorithms have been applied to PSSP, utilizing the features selected by PCA. Tab. 2 displays the results of this experiment. Tab. 2 indicates that DT has the highest performance with accuracy = 64.00%, precision = 0.6970, Recall = 0.4647, F-score = 0.5576 and AUC = 0.80, while NB has the worst performance with accuracy of 63.50%, recall of 0.3939, and precision of 0.7105. The KNN technique has been applied with various number of nearest neighbors k = 1, 2, 3, 5, and 9. The best value was k = 1 that achieved accuracy of 63.80%, recall of 0.4527, precision of 0.7119, and AUC of 0.77. It is obvious from Tab. 2 that DT with PCA technique outperforms the other ML techniques, followed by KNN.

Table 2: Results of applying ML techniques using features extracted by PCA

Techniques	Accuracy	Precision	Recall	F-score	AUC
KNN	63.80	0.7119	0.4527	0.5535	0.77
SVM	63.70	0.6801	0.4316	0.5281	0.70
DT	64.00	0.6970	0.4647	0.5576	0.80
RF	63.77	0.6853	0.4313	0.5294	0.76
NB	63.50	0.7105	0.3939	0.5066	0.73

4.2.2 Experiment 2: Applying the ML Algorithms to Features Extracted by LDA

In the second experiment, features have been selected by LDA. Tab. 3 shows that DT, KNN (k = 1), and RF have the best performance with accuracy = 64.31%, precision = 0.6970, recall = 0.4647, F-score = 0.5576, and AUC = 0.82. The results shown in Tab. 3 indicate that DT, KNN, and RF with LDA technique outperform both SVM and NB.

4.2.3 Experiment 3: Applying the Bagging Technique to Features Extracted by PCA

In the third experiment, the ensemble models, constructed by combining bagging with each of the 5 ML algorithms, with 9-fold cross-validation method, have been applied to PSSP, utilizing features extracted by PCA. Tab. 4 shows that DT has reached the best accuracy of 65.81%, recall of 0.4647, AUC of 0.84, and F-score of 0.5576, but KNN (k = 1) has reached the best precision 0.7119. NB has reached the worst accuracy of 64.00%, recall of 0.3939, AUC of 0.72 and F-score of 0.5066, while SVM has reached the worst precision of 0.6801. These results indicate that DT outperforms the other ML techniques when applying the bagging technique with them using features extracted by PCA.

Table 3: Results of applying ML techniques using features extracted by LDA

Techniques	Accuracy	Precision	Recall	F-score	AUC
KNN	64.31	0.6970	0.4647	0.5576	0.82
SVM	63.70	0.6801	0.4316	0.5281	0.75
DT	64.31	0.6970	0.4647	0.5576	0.82
RF	64.31	0.6970	0.4647	0.5576	0.82
NB	63.55	0.7105	0.3939	0.5066	0.77

Table 4: Bagging with the 5 ML techniques using features extracted by PCA

Techniques	Accuracy	Precision	Recall	F-score	AUC
KNN	64.77	0.7119	0.4527	0.5535	0.80
SVM	64.7	0.6801	0.4316	0.5281	0.79
DT	65.81	0.6970	0.4647	0.5576	0.84
RF	65.00	0.6853	0.4313	0.5294	0.82
NB	64.00	0.7105	0.3939	0.5066	0.72

4.2.4 Experiment 4: Applying the Bagging Technique to Features Extracted by LDA

Tab. 5 shows the results when LDA is used, it can be seen that DT, KNN (k = 1) and RF have reached the best accuracy of 66.00% and F-score of 0.5294, while NB has reached the best precision of 0.7105, SVM has reached the best recall of 0.4316, and KNN (k = 1) has reached the best AUC of 0.84.

Table 5: Bagging with the 5 ML techniques using features extracted by LDA

Techniques	Accuracy	Precision	Recall	F-score	AUC
KNN	66.00	0.6853	0.4313	0.5294	0.84
SVM	65.21	0.6801	0.4316	0.5281	0.80
DT	66.00	0.6853	0.4313	0.5294	0.82
RF	66.00	0.6853	0.4313	0.5294	0.79
NB	65.00	0.7105	0.3939	0.5066	0.72

4.2.5 Experiment 5: Applying the Boosting Technique to Features Extracted by PCA

In the fifth experiment, the ensemble models, constructed by combining boosting with each of the 5 ML algorithms, with 9-fold cross-validation method, have been applied to PSSP, utilizing features extracted by PCA. From Tab. 5, we can see that RF achieved the best accuracy = 66.50%, recall = 0.4647, F-score = 0.5576, and AUC = 0.85, while DT achieved the best precision of 0.7119. In addition, Tab. 5 showed that SVM reached the lowest accuracy, precision, and AUC of 64.00%, 0.6801, 0.75, respectively, while NB reached the lowest recall and F-score of 0.3939 and 0.5066, respectively.

Table 6: Boosting with the 5 ML techniques using features extracted by PCA

Techniques	Accuracy	Precision	Recall	F-score	AUC
KNN	65.80	0.6853	0.4313	0.5294	0.79
SVM	64.00	0.6801	0.4316	0.5281	0.75
DT	65.71	0.7119	0.4527	0.5535	0.83
RF	66.50	0.6970	0.4647	0.5576	0.85
NB	64.70	0.7105	0.3939	0.5066	0.80

4.2.6 Experiment 6: Applying the Boosting Technique to Features Extracted by LDA

In the sixth experiment, the ensemble models, constructed by combining boosting with each of the 5 ML algorithms, with 9-fold cross-validation method, have been applied to PSSP, utilizing features extracted by LDA. Tab. 7 shows that RF achieved the best accuracy = 66.52%, recall = 0.4647, F-score = 0.5576, and AUC = 0.85, while DT achieved the best precision of 0.7119. In addition, Tab. 7 showed that SVM reached the lowest accuracy, precision, and AUC of 63.50%, 0.6801, 0.75, respectively, while NB reached the lowest recall and F-score of 0.3939 and 0.5066, respectively.

Table 7: Boosting with the 5 ML techniques using features extracted by LDA

Techniques	Accuracy	Precision	Recall	F-score	AUC
KNN	65.80	0.6853	0.4313	0.5294	0.79
SVM	64.00	0.6801	0.4316	0.5281	0.75
DT	65.71	0.7119	0.4527	0.5535	0.83
RF	66.50	0.6970	0.4647	0.5576	0.85
NB	64.70	0.7105	0.3939	0.5066	0.80

4.3 Discussion

Tab. 8 summarizes the best values of the evaluation metrics: accuracy, recall, precision, F-score, and AUC, obtained in the presented experiments.

Table 8: The best evaluation metrics values obtained by the ML ensemble learning techniques

Techniques	Accuracy	Precision	Recall	F-score	AUC
PCA + Bagging	DT 65.81	DT 0.4647	KNN 0.7119	DT 0.5576	DT 0.84
LDA + Bagging	DT, KNN, RF 66.00	SVM 0.4316	NB 0.7105	DT, KNN, RF 0.5294	KNN 0.84
PCA + Boosting	RF 66.50	RF 0.4647	DT 0.7119	RF 0.5576	RF 0.85
LDA + Boosting	RF 66.52	RF 0.4647	DT 0.7119	RF 0.5576	RF 0.85

Fig. 4 shows a comparison between accuracy values of applying ML algorithms with and without ensemble learning using features extracted by PCA. This comparison indicates that, in general, the accuracy of the ML techniques has been improved by using ML ensemble learning as follows:

- The accuracy of KNN, RF and NB, were better with boosting than with bagging
- The accuracy of SVM and DT, were better with bagging than with boosting.

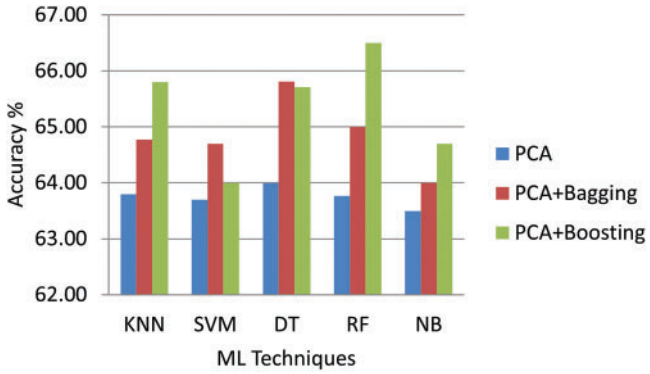

Figure 4: Accuracy values of applying ML algorithms with and without Ensemble Learning using features extracted by PCA

Fig. 5 shows a comparison between AUC values of applying ML algorithms with and without ensemble learning using features extracted by PCA. This comparison indicates that, in general, the AUC values of the ML techniques have been improved by using ML ensemble learning, as follows:

- The AUC values of RF and NB, were better with boosting than with bagging
- The AUC values of KNN, SVM and DT, were better with bagging than with boosting.

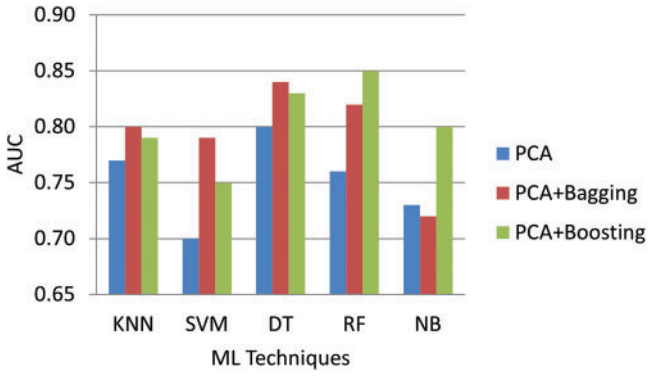

Figure 5: AUC values of applying ML algorithms with and without Ensemble Learning using features extracted by PCA

Fig. 6 shows a comparison between Accuracy values of applying ML algorithms with and without ensemble learning using features extracted by LDA. This comparison indicates that, in general, the accuracy of the ML techniques has been improved by using ML ensemble learning, as follows:

- The accuracy of KNN, SVM, DT and NB, were better with bagging than with boosting.
- The accuracy of RF was better with boosting than with bagging.

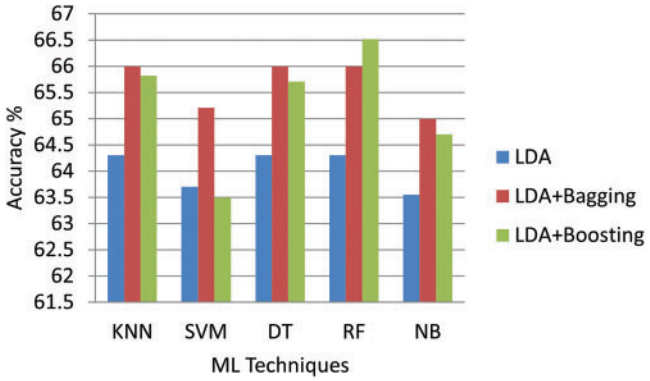

Figure 6: Accuracy values of applying ML algorithms with and without Ensemble Learning using features extracted by LDA

Fig. 7 shows a comparison between AUC values of applying ML algorithms with and without ensemble learning using features extracted by LDA. This comparison indicates that, in general, the AUC values of the ML techniques have been improved by using ML ensemble learning, as follows:

- The AUC values of DT, RF, and NB, were better with boosting than with bagging
- The AUC values of KNN and SVM were better with bagging than with boosting.

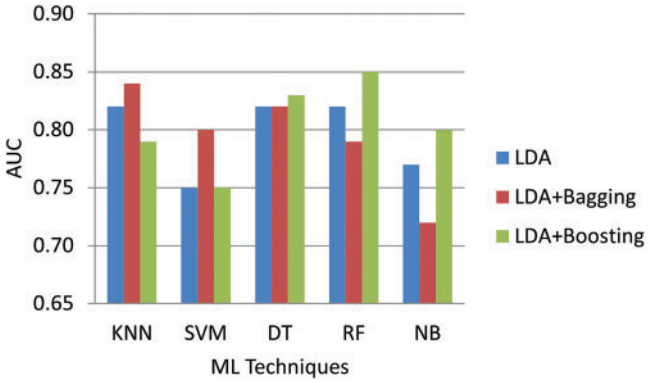

Figure 7: AUC values of applying ML algorithms with and without Ensemble Learning using features extracted by LDA

5 Conclusion

This paper presented a proposed ensemble machine learning approach to improve the performance of Q8 PSSP. In this approach, ensemble models for PSSP are constructed by combining two ensemble methods, namely Bagging and Boosting, with 5 ML algorithms, namely KNN, SVM, DT, RF, and NB, as ensemble members (base learners), and two feature extraction techniques, namely PCA and LDA.

We have conducted experiments for evaluating the performance of the 5 individual predicting models, constructed using the 5 ML algorithms, KNN, SVM, DT, RF, and NB, and the ensemble models, constructed by combining each of the ensemble ML methods, bagging and boosting, with

each of the 5 ML algorithms, and the two feature extraction techniques, PCA and LDA, using cross validation tests on CB6133 training dataset and CB513 test data, in Q8 PSSP.

The experimental results indicated that:

- The best of all accuracy values, 66.52% and 66.50%, were obtained by Boosting with RF using features extracted by PCA and LDA, respectively.
- With features extracted by PCA, the accuracy of KNN, RF and NB, were better with boosting than with bagging, while the accuracy of SVM and DT, were better with bagging than with boosting.
- With features extracted by LDA, the accuracy of KNN, SVM, DT and NB, were better with bagging than with boosting, while the accuracy of RF was better with boosting than with bagging.
- The ML techniques KNN, DT, RF and NB achieved better accuracy when using features extracted by LDA than those extracted by PCA, while SVM achieved same accuracy when using both PCA and LDA.
- All bagging ensemble models achieved better accuracy when using features extracted by LDA than those extracted by PCA.
- Boosting with DT and NB achieved same accuracy when using both PCA and LDA, while boosting with KNN and RF achieved a little bit higher accuracy when using LDA than PCA and boosting with SVM achieved better accuracy when using PCA than LDA.
- For individual models, the best accuracy value was 64.31%, achieved by KNN, DT, and RF algorithms, when using features extracted by LDA.
- For bagging ensemble models, the best accuracy value was 66.00%, achieved with KNN, DT, and RF algorithms, when using features extracted by LDA.
- For boosting ensemble models, the best accuracy values were 66.52% and 66.50%, achieved with RF algorithm, when using features extracted by LDA and PCA, respectively.

These results clearly confirm that ensemble ML models are more accurate than individual ML models. They also indicate that features extracted by LDA are more effective than those extracted by PCA.

Funding Statement: The authors received no specific funding for this study.

Conflicts of Interest: The authors declare that they have no conflicts of interest to report regarding the present study.

References

[1] L. Hunter, "Knowledge-based biomedical Data Science," *EPJ Data Science*, vol. 1, no. 2, pp. 19–25, 2017.
[2] P. Berg, "Fred Sanger: A memorial tribute," *Proceedings of the National Academy of Sciences*, vol. 111, no. 3, pp. 883–884, 2014.
[3] W. Sun, X. Chen, X. R. Zhang, G. Z. Dai, P. S. Chang *et al.,* "A multi-feature learning model with enhanced local attention for vehicle re-identification," *Computers, Materials & Continua*, vol. 69, no. 3, pp. 3549–3560, 2021.
[4] PDB, "Protein data bank," 2015. [Online]. Available: http://www.rcsb.org/pdb/home/home.do.
[5] R. Adiyaman and L. McGuffin, "Methods for the refinement of protein structure 3D models," *International Journal of Molecular Sciences*, vol. 20, no. 3, pp. 2301, 2019.
[6] B. Berger and F. Leighton, "Protein folding in the hydrophobic-hydrophilic (hp) is np-complete," *Journal of Computational Biology*, vol. 5, no. 1, pp. 27–40, 1998.

[7] UniProt, "Protein sequence database," [Online]. Available:, 2014. [Online]. Available: http://www.uniprot.org/.
[8] W. Sun, G. C. Zhang, X. R. Zhang, X. Zhang and N. N. Ge, "Fine-grained vehicle type classification using lightweight convolutional neural network with feature optimization and joint learning strategy," *Multimedia Tools and Applications*, vol. 80, no. 20, pp. 30803–30816, 2021.
[9] J. Zhou, H. Wang, Z. Zhao, R. Xu and Q. Lu, "CNNH PSS: Protein 8-class secondary structure prediction by convolutional neural network with highway," *BMC Bioinformatics*, vol. 19, no. 4, pp. 60–71, 2018.
[10] W. Kabsch and C. Sander, "Dictionary of protein secondary structure: Pattern recognition of hydrogen-bonded and geometrical features," *Biopolymers*, vol. 22, no. 12, pp. 2577–2637, 1983.
[11] J. Zhou and O. Troyanskaya, "Deep supervised and convolutional generative stochastic network for protein secondary structure prediction," ArXiv abs/1403.1347, 2014.
[12] Z. Li and Y. Yu, "Protein secondary structure prediction using cascaded convolutional and recurrent neural networks," ArXiv abs/1604.07176, 2016.
[13] H. Bouziane, B. Messabih and A. Chouarfia, "Effect of simple ensemble methods on protein secondary structure prediction," *Soft Computing*, vol. 19, no. 6, pp. 1663–1678, 2014.
[14] Z. Zhou, "Ensemble learning," in *Encyclopedia of Biometrics*, Springer, Boston, 2009.
[15] V. G. Bittencourt, M. C. Abreu, M. C. de Souto and A. Canuto, "An empirical comparison of individual machine learning techniques and ensemble approaches in protein structural class prediction," in *Proc. of 2005 IEEE Int. Joint Conf. on Neural Networks*, Montreal, Quebec, pp. 527–5311, 2005.
[16] L. Lin, S. Yang and R. Zuo, "Protein secondary structure prediction based on multi-SVM ensemble," in *2010 Int. Conf. on Intelligent Control and Information Processing*, Dalian, China, pp. 356–358, 2010.
[17] G. Zong, J. Liu, Y. Zhang and L. Hou, "Delay-range-dependent exponential stability criteria and decay estimation for switched hopfield neural networks of neutral type," *Nonlinear Analysis: Hybrid Systems*, vol. 2010, no. 4, pp. 583–592, 2010.
[18] S. S. Bacanli, F. Cimen, E. Elgeldawi and D. Turgut, "Placement of package delivery center for UAVs with machine learning," in *2021 IEEE Global Communications Conf. (GLOBECOM)*, Madrid, Spain, pp. 1–6, 2021.
[19] A. A. Radwan, T. M. Mahmoud and E. Elgeldawi, "Improving the efficiency of the flow deviation method for solving the optimal routing problem in a packet-switched computer network," *International Journal of Applied Mathematics*, vol. 5, no. 2, pp. 171–187, 2001.
[20] A. A. Radwan and E. Elgeldawi, "Solving the optimal routing problem in a packet-switching computer network using decomposition," *Egyptian International Journal*, vol. 4, no. 9, pp. 1–13, 2003.
[21] E. Elgeldawi, A. Sayed, A. R. Galal and A. M. Zaki, "Hyperparameter tuning for machine learning algorithms used for Arabic sentiment analysis," *Informatics*, vol. 8, no. 4, pp. 79, 2021.
[22] A. A. Sayed, E. Elgeldawi, A. M. Zaki and A. R. Galal, "Sentiment analysis for Arabic reviews using machine learning classification algorithms," in *Proc. of 2020 Int. Conf. on Innovative Trends in Communication and Computer Engineering (ITCE)*, Aswan, Egypt, pp. 56–63, 2020.
[23] W. Wang, X. Huang, J. Li, P. Zhang and X. Wang, "Detecting COVID-19 patients in X-ray images based on MAI-nets," *International Journal of Computational Intelligence Systems*, vol. 14, no. 1, pp. 1607–1616, 2021.
[24] Y. Gui and G. Zeng, "Joint learning of visual and spatial features for edit propagation from a single image," *Visual Computer*, vol. 36, no. 3, pp. 469–482, 2020.
[25] W. Wang, Y. T. Li, T. Zou, X. Wang, J. Y. You *et al.,* "A novel image classification approach via Dense-MobileNet models," *Mobile Information Systems*, vol. 2020, no. 7602384, pp. 1–8, 2020.
[26] S. R. Zhou, J. P. Yin and J. M. Zhang, "Local binary pattern (LBP) and local phase quantization (LBQ) based on Gabor filter for face representation," *Neurocomputing*, vol. 2013, no. 116, pp. 260–264, 2013.
[27] Y. Song, D. Zhang, Q. Tang, S. Tang and K. Yang, "Local and nonlocal constraints for compressed sensing video and multi-view image recovery," *Neurocomputing*, vol. 2020, no. 406, pp. 34–48, 2020.

[28] D. Zhang, S. Wang, F. Li, S. Tian, J. Wang *et al.,* "An efficient ECG denoising method based on empirical mode decomposition, sample entropy, and improved threshold function," *Wireless Communications and Mobile Computing*, vol. 2020, no. 2, pp. 1–11, 2020.
[29] F. Li, C. Ou, Y. Gui and L. Xiang, "Instant edit propagation on images based on bilateral grid," *Computers, Materials & Continua*, vol. 61, no. 2, pp. 643–656, 2019.
[30] Y. Song, Y. Zeng, X. Y. Li, B. Y. Cai and G. B. Yang, "Fast CU size decision and mode decision algorithm for intra prediction in HEVC," *Multimedia Tools and Applications*, vol. 76, no. 2, pp. 2001–2017, 2017.
[31] E. Elgeldawi, M. Mahrous and A. Sayed, "A comparative analysis of symmetric algorithms in cloud computing: A survey," *International Journal of Computer Applications*, vol. 182, no. 48, pp. 7–16, 2019.
[32] E. Elgeldawi, A. A. Radwan, F. Omara and T. M. Mahmoud, "Detection and characterization of fake accounts on the pinterest social networks," *International Journal of Computer Networking, Wireless and Mobile Communication*, vol. 4, no. 3, pp. 21–28, 2014.
[33] A. A. Radwan, H. V. Madhyastha, F. Omara and T. M. Mahmoud, "Pinterest attraction between users and spammers," *International Journal of Computer Science Engineering and Information Technology Research*, vol. 4, no. 1, pp. 63–72, 2014.
[34] M. R. Girgis, E. Elgeldawi and R. M. Gamal, "A comparative study of various deep learning architectures for 8-state protein secondary structures prediction," in *Proc. of the Int. Conf. on Advanced Intelligent Systems and Informatics 2020*, Cairo, Egypt, Springer, pp. 501–513, 2021.
[35] W. Wardah, M. Khan, A. Sharma and M. A. Rashid, "Protein secondary structure prediction using neural networks and deep learning: A review," *Computational Biology and Chemistry*, vol. 2019, no. 81, pp. 1–8, 2019.
[36] G. Wang and R. L. Dunbrack, "Pisces: A protein sequence culling server," *Bioinformatics*, vol. 19, no. 12, pp. 1589–1591, 2003.
[37] W. Li and A. Godzik, "CD-hit: A fast program for clustering and comparing large sets of protein or nucleotide sequences," *Bioinformatics*, vol. 22, no. 13, pp. 1658–1659, 2006.
[38] C. Bishop and N. Nasrabadi, "Pattern recognition and machine learning," *Journal of Electronic Imaging*, vol. 16, no. 4, pp. 1–16, 2007.
[39] Z. M. Hira and D. Gillies, "A review of feature selection and feature extraction methods applied on microarray data," *Advances in Bioinformatics*, vol. 2015, no. 112, pp. 1–13, 2015.
[40] D. Zhang, L. Zou, X. Zhou and F. He, "Integrating feature selection and feature extraction methods with deep learning to predict clinical outcome of breast cancer," *IEEE Access*, vol. 6, pp. 28936–28944, 2018.
[41] P. Xu, G. N. Brock and R. S. Parrish, "Modified linear discriminant analysis approaches for classification of high-dimensional microarray data," *Computational Statistics & Data Analysis*, vol. 53, no. 5, pp. 1674–1687, 2009.
[42] Y. Freund and R. Schapire, "A decision-theoretic generalization of on-line learning and an application to boosting," *Journal of Computers and System Sciences*, vol. 55, no. 1, pp. 119–139, 1997.
[43] R. Schapire, "The strength of weak learnability," *Machine Learning*, vol. 5, no. 2, pp. 197–227, 2005.
[44] L. Breiman, "Bagging predictor," *Machine Learning*, vol. 24, no. 2, pp. 123–140, 1996.
[45] S. Jukic, M. Saracevic, A. Subasi and J. Kevric, "Comparison of ensemble machine learning methods for automated classification of focal and nonfocal epileptic eeg signals," *Mathematics*, vol. 8, no. 9, pp. 1–16, 2020.
[46] S. Abdullah, N. Rostamzadeh, K. Sedig, D. Lizotte, A. X. Garg *et al.,* "Machine learning for identifying medication-associated acute kidney injury," *Informatics*, vol. 7, no. 2, pp. 18, 2020.
[47] L. Breiman, "Random forests," *Machine Learning*, vol. 2004, no. 45, pp. 5–32, 2004.
[48] H. Li and S. Misra, "Robust machine-learning workflow for subsurface geomechanical characterization and comparison against popular empirical correlations," *Expert Systems with Applications*, vol. 177, no. 77, pp. 114942, 2021.
[49] J. Wang, A. Zelenyuk, D. Imre and K. Mueller, "Big data management with incremental K-means trees-GPU-accelerated construction and visualization," *Informatics*, vol. 4, no. 3, pp. 24, 2017.