Research Article
PSO-LocBact: A Consensus Method for Optimizing Multiple Classifier Results for Predicting the Subcellular Localization of Bacterial Proteins

Supatcha Lertampaiporn, Sirapop Nuannimnoi, Tayvich Vorapreeda, Nipa Chokesajjawatee, Wonnop Visessanguan, and Chinae Thammarongtham

Biochemical Engineering and Systems Biology Research Group, National Center for Genetic Engineering and Biotechnology (BIOTEC), King Mongkut's University of Technology Thonburi, Bangkok 10150, Thailand
Food Biotechnology Laboratory, National Center for Genetic Engineering and Biotechnology (BIOTEC), 113 Phahonyothin Rd., Khlong Luang, Pathumthani 12120, Thailand

Correspondence should be addressed to Chinae Thammarongtham; chinae@biotec.or.th

Received 12 July 2019; Revised 3 October 2019; Accepted 30 October 2019; Published 19 November 2019

1. Introduction

The prediction of the subcellular localization of proteins is a significant step in protein function annotation, providing useful insights into biological functions and interactions. Information involving the subcellular localization of proteins in bacteria can support the development of drugs and vaccines [1]. Bacterial cell surfaces and secreted proteins are of interest for their potential as vaccine candidates or diagnostic targets. Using experimental techniques, identifying the subcellular localization of a protein is relatively laborious and time consuming. However, reliable and accurate computational methods of predicting subcellular localization can accelerate this process. Over the past decades, numerous prediction methods have been proposed as a result of independent efforts by various research teams (summarized in Table 1). Yu et al. [2, 3] developed CELLO, a multilayered SVM classification system that uses 4 types of sequence coding schemes, namely, amino acid composition, dipeptide composition, partitioned amino acid composition, and physicochemical-property-based sequence composition, to predict protein locations. Bhasin et al. [5] developed PSLpred, which includes various SVM modules based on features such as amino acid composition, dipeptide composition, physicochemical properties, and evolutionary information from PSI-BLAST. Later, SLP-Local [6] was
developed to predict the subcellular localization of proteins based only on the local compositions of amino acids and twin amino acids and the local frequencies of the distances between successive amino acids. SOSUI-GramN [1] was proposed as a predictive software system developed specifically for assessing the subcellular localization of proteins in Gram-negative bacteria. It utilizes only the physicochemical parameters of the N- and C-terminal signal sequences and the total sequence. In particular, SOSUI-GramN offers markedly improved accuracy for the localization prediction of extracellular proteins, which is commonly known as a weakness of other methods. Gneg-mPLoc and Gpos-mPLoc were developed by Shen et al. [7, 9] as components of Cell-PLoc [8, 13], a web server for predicting the subcellular localization of proteins in various organisms. These tools can be used for cases in which a query protein may simultaneously exist in more than one location. PSORTb 3.0, the latest version of a well-known method for bacterial protein analysis [10], uses information on amino acid composition, similarity to proteins of known localization, the presence of a signal peptide, transmembrane alpha-helices, and motifs corresponding to specific locations found for each given protein to determine its subcellular localization. By using a probabilistic method, PSORTb 3.0 outperforms CELLO, Cell-PLoc, SLP-Local, and the previous versions of the same tool. King and Guda [11] proposed an n-gram-based Bayesian subcellular localization classifier called ngLOC. As part of its output, ngLOC provides a set of probabilistic scores for the top three possible locations of each given protein. Later, in early 2014,
Goldberg et al. [12] presented LocTree3, a profile kernel SVM with the addition of homology-based inference, for protein subcellular localization prediction. Yu et al. [4] presented a new version of CELLO called CELLO2GO, which combines the original technique with information regarding gene ontology (GO) categories to describe the functions of genes and gene products across species.

Nevertheless, each prediction program has unique weaknesses and strengths depending on the adopted training strategies and algorithms. Specifically, these tools differ in three notable aspects: the underlying biological model, location coverage, and prediction accuracy [14]. A given tool may not be able to accurately predict the exact localization of every protein. It often happens that one predictor performs better for some cases while another predictor performs better for another compartment or under other circumstances. During the genome annotation process, a user may consider results from multiple prediction programs to confirm the final prediction and may encounter conflicting predictions. It is difficult for users to arrive at sensible decisions when faced with two or more contradictory predictions made by multiple programs [15]. To address this problem, the combination of multiple predictive models via a consensus classifier has become a promising solution. Efforts have been made to combine results from multiple predictors to generate a final prediction. In 2012, a meta-predictor for protein localization in Gram-negative bacteria was introduced by Magnus et al. [16]. Their predictor combines the results from various prediction tools by using 5 one-versus-rest binary logistic regression models. This approach was developed based on the conversion of the multiclass classification problem into a set of independent binary logistic regression classification problems. On this basis, the class label corresponding to the logistic regression classifier with the highest probability will be returned as the final prediction. However, naively comparing the probabilities of separate and independent binary logistic regression classifiers may result in irrelevant decision boundaries that will affect the correctness of the final prediction due to imbalances between the classes. Therefore, the motivation of this work is to instead estimate the probabilities of all classes simultaneously; hence, the interdependence of all classes will also be estimated as part of the joint classification process.

To this end, we propose a new subcellular localization predictor for bacterial proteins using particle swarm optimization (PSO) that efficiently combines prediction results from preexisting predictors to improve the overall predictive accuracy and resolve incongruent results from different predictors. To date, many subcellular localization predictors have been proposed. The goal of this work is not to develop another trained classifier based on certain selected features; instead, the aim is to introduce a PSO-based consensus classifier to combine and enhance the strengths of the previous methods. The main reasons for choosing PSO instead of another optimization method for this multiclass problem are its iterative search capability for identifying the global optimum in a multidimensional space and its ease of continuous data representation, which permits easy modification in the case of removing or adding predictors. Moreover, PSO does not rely on the gradient of the problem to be optimized; thus, PSO does not require that the optimization problem be differentiable, as is required by classic optimization methods [17–19]. Recently, a PSO-based consensus method has been successfully applied to classify eukaryotic protein localization results [20].

In this work, the application of PSO in optimizing the weights and biases of various prediction methods enhances the accuracy of a prediction model for protein localization in bacterial genome sequences. This method can be used to identify the locations of the proteins from 5 locations in Gram-negative bacteria (extracellular region or secreted proteins, outer membrane, periplasm, inner membrane or cytoplasmic membrane, and cytoplasm) or 4 locations in Gram-positive bacteria (extracellular region, cell wall, inner membrane, and cytoplasm). Empirical experiments performed under various circumstances suggest that the proposed PSO-based consensus classifier offers significantly improved performance compared with the individual predictors.

2. Materials and Methods

The flowchart of the proposed method is illustrated in Figure 1.

2.1. Data Collection. Protein sequences with known locations were extracted from UniProtKB [21]. Only sequences with the reviewed (Swiss-Prot manually annotated) status were collected. Duplicated proteins with over 90% sequence identity were removed by using CD-HIT [22]. We randomly selected 2,150 Gram-negative and 1,866 Gram-positive nonredundant bacterial proteins with less than 90% sequence identity from the resulting dataset. For each dataset, approximately 80% of the data were used as a training set, and the remaining proteins after removal were used as a test set. The test dataset for Gram-negative bacterial proteins covered five locations, with 86 proteins for each location. The test dataset for Gram-positive bacterial proteins consisted of 311 proteins, including 79 sequences from cytoplasm, 79 sequences from inner membranes, 77 sequences from cell walls, and 76 sequences from extracellular regions. After data collection, the following individual predictors for bacterial protein subcellular localization were employed as the selected classifiers: CELLO [2, 3], PSORTb 3.0 [12], CELLO2GO [4], SOSUI-GramN [1], SLP-Local [6], ngLOC [11], Gneg-mPLoc [9], Gpos-mPLoc [7], PSLpred [5], and LocTree3 [12]. Some of them are available for local standalone installation, whereas some are available only on web servers. For servers that do not accept one file containing multiple protein sequences, we used the screen-scraping technique with Python to submit inputs and fetch outputs (the screen-scraping codes are also provided with the software). CELLO, PSORTb 3.0, ngLOC, and SLP-Local yield scores for the probabilities of class assignment, whereas the other programs provide only the location predictions; hence, in the latter case, we assigned a label of 1 to the
predicted location and a label of 0 to the other locations. Once all results had been obtained in the form of numerical vectors, we simply combined them into one CSV file to serve as the input for the PSO classifier.

In addition to the data described above, we also employed the benchmark dataset S taken from [7, 9]. This dataset includes 523 proteins (4 locations) for Gram-positive bacteria and 1,404 proteins (5 locations) for Gram-negative bacteria. None of the proteins included in dataset S has a pairwise sequence identity of >25% with respect to any other in the same subcellular location. This dataset S is much more rigorous in excluding homology bias and redundancy. Moreover, this dataset is well documented and has been used in benchmarking various predictors [7, 9, 23–30].

2.2. Experimentation. This section briefly explains the experimentation involving the performance comparison of the proposed method in different settings. All predictors were evaluated on the same test datasets. The steps of the algorithm are described below.

(1) The result score matrix was prepared and loaded for the PSO classifier. This score matrix was used for weight optimization in the PSO algorithm.

(2) The weights were multiplied by the scores. Scores for the same location were summed together, and the results were then sorted in descending order. The location with the maximum score was selected for comparison with the given class label of each protein sequence.

(3) The performance of the method was further evaluated by considering the following 9 experimental cases:

(i) The classifier with the highest accuracy was removed to observe how its removal would influence the result.

(ii) Tools that exhibited an accuracy lower than 90% were removed.

(iii) As a complement to 3.2, all other tools with an accuracy of 90% or higher were removed to determine whether the proposed method could improve the prediction accuracy in the case of only relatively inaccurate predictors.

(iv) Tools that exhibited an accuracy lower than 80% were removed.

(v) As a complement to 3.4, all other tools with an accuracy of 80% or higher were removed.

(vi) Tools that exhibited an accuracy lower than 70% were removed.

(vii) As a complement to 3.6, all other tools with an accuracy of 70% or higher were removed.

(viii) Tools that exhibited an accuracy lower than 60% were removed.

(ix) As a complement to 3.8, all other tools with an accuracy of 60% or higher were removed.

All experimental results are reported and compared to illustrate the effects of the different settings on the proposed method. In every step of the evaluation, the overall prediction accuracy was calculated as shown in

\[
\text{accuracy (ACC)} = \frac{\text{number of correct answers}}{\text{number of instances}},
\]

2.3. Particle Swarm Optimization. PSO is a metaheuristic method because it makes few or no assumptions regarding the problem being optimized. A basic variant of the PSO algorithm [17, 31] works by using a population of candidate solutions (also known as particles) to explore the feasible search space. Each of these \(k\) particles is represented by a position vector \(X_k\) and a velocity vector \(V_k\). The movements
of the particles are driven by their best-known positions Pb (local best) in addition to the entire swarm’s best-known position Pg (global best) in the search space, as shown in

\[ V_k^{d} = \omega V_k^{d} + c_1 r_1 \left( \text{Pb}_k^{d} - X_k^{d} \right) + c_2 r_2 \left( \text{Pg}_k^{d} - X_k^{d} \right), \]  \hspace{1cm} (2)

\[ X_k^{d} = X_k^{d} + V_k^{d} \]  \hspace{1cm} (3)

where \( d \) is the dimensionality of the problem, or the number of decision variables to be optimized. The PSO algorithm searches for the optimal solution in an iterative manner. In each iteration, the velocity \( V \) is updated by using the most recent velocity \( V' \) as well as the cognitive coefficient \( c_1 \) of the particle and the social coefficient \( c_2 \) of the members of the swarm multiplied by random variables \( r_1 \) and \( r_2 \), respectively. The new position \( X_k \) is updated with respect to the previous position \( X_k' \) in accordance with the updated \( V_k \). A flowchart of the PSO algorithm is shown in Figure 2.

In this work, the time-varying acceleration coefficients proposed in [18] are adopted. In this version of the PSO algorithm, the cognitive coefficient \( c_1 \) and the social coefficient \( c_2 \) are defined to be adaptable. Beginning with a larger cognitive component and a smaller social component, the particles move around the search space instead of immediately moving toward the population’s best solution. After several objective function calls, each particle has explored and collected adequate information about the search space, and the coefficients are correspondingly modified to obtain a smaller cognitive component and a larger social component to directly drive convergence to the global optimum. The modification of these two acceleration coefficients can be represented as follows:

\[ c_1 = (c_{1 \text{max}} - c_{1 \text{min}}) \cdot \left( \text{MAXCALL} - \text{calls} \right) / \text{MAXCALLS} + c_{1 \text{min}}, \]  \hspace{1cm} (4)

\[ c_2 = (c_{2 \text{max}} - c_{2 \text{min}}) \cdot \left( \text{MAXCALL} - \text{calls} \right) / \text{MAXCALLS} + c_{2 \text{max}}, \]

where the maximum coefficient values \( c_{1 \text{max}} \) and \( c_{2 \text{max}} \) and the minimum coefficient values \( c_{1 \text{min}} \) and \( c_{2 \text{min}} \) are constants, \text{calls} is the most recent count of objective function calls, and MAXCALLS is the maximum allowed number of objective function calls. Moreover, this method uses a time-varying inertial weight factor \( \omega \), as shown in

\[ \omega = (\omega_{\text{max}} - \omega_{\text{min}}) \cdot \left( \text{MAXCALL} - \text{calls} \right) / \text{MAXCALLS} + \omega_{\text{min}}, \]  \hspace{1cm} (5)

where \( \omega_{\text{max}} \) and \( \omega_{\text{min}} \) are the initial and final values, respectively, of the inertial weight factors. This factor balances the local and global search capabilities during the optimization process. With a larger inertial weight factor at the beginning, the particles move more broadly and quickly around the search space. In contrast, a smaller inertial weight enables the particles to more precisely explore the search space surrounding the global optimum.

For this problem, the weights for all tools are represented in the PSO algorithm by the position vector of each particle. The New Result Vector is structured as follows:

\[ \text{New Result Vector} = \left[ \sum_{i=1}^{n} (w_1 x_{i1}), \sum_{i=1}^{n} (w_1 x_{i2}), \ldots, \sum_{i=1}^{n} (w_n x_{im}) \right], \]  \hspace{1cm} (6)

where \( w_1, w_2, \ldots, w_n \) are the weights for all \( n \) classifiers and \( x_{i1}, x_{i2}, \ldots, x_{im} \) are the elements of the normalized result score vector corresponding to each of the \( m \) locations generated by each classifier used in this work.

2.4. Decision-Making. While the PSO algorithm is running, the New Result Vector is used to determine the protein location. Only the location with the maximum score in the matrix is determined as the final answer. Therefore, the decision rule is as follows:

\[ \text{seq} \rightarrow \text{location} = \sum_{k=1}^{n} (w_j x_{ik}) = \max \sum_{j=1}^{n} (w_j x_{ij}). \]  \hspace{1cm} (7)

Later, the answer is automatically checked against the class label to evaluate the performance of the PSO weights in terms of accuracy. In our study, following the results of an empirical study by Shi and Eberhart [17], the PSO parameters were set to widely used values. \( c_1 \) and \( c_2 \) were set to vary over time from 2.5 to 0.5 and from 0.5 to 2.5, respectively. The inertial weight \( \omega \) was decreased linearly from approximately 0.9 to 0.4 throughout a run. We set the number of particles to 25, and we adopted a maximum allowed number of objective functions calls of 1,000 per run as the termination criterion.

2.5. Software Package. The PSO-LocBact software package was developed in Python and Perl using Spyder with Python 2.7 and Perl v5.22.1. Detailed documentation is provided with the package. The program offers cross-platform compatibility. The original dataset files are included in FASTA file format. The user manual takes users through the basic usage of the software package and the settings in the configuration file for the summarization of the prediction results from other classifiers. With the guidelines provided in the user manual, users can also create and apply their own training datasets. By changing the settings in the configuration file, users can add new predictor programs and
weights for their results. The PSO algorithm will consider these weights along with the probabilistic scores resulting from each predictor in the calculation of the final results. Since the software package was developed entirely in scripting languages, no additional source code is needed. Any desired modifications can be easily and freely made to the software package.

3. Results and Discussion

3.1. Predictive Performance Comparison. We assessed the performance of the 10 predictors used in this study (as summarized in Table 1). Table 2 shows the prediction performance of each tool used in this study. Table 2 confirms the hypothesis that some tools are better than others in predicting localization in certain compartments. Additionally, the results from each predictor are not reliable for identifying the localization of proteins in every compartment. For example, PSORTb 3.0 is the most accurate classifier, but it is not as accurate as ngLOC, CELLO, and LocTree3 in classifying cytoplasmic proteins. As another example, SLP-Local outperforms SOSUI-GramN, Gneg-mPLoc, and PSLpred for the prediction of periplasmic proteins despite its limited overall prediction competence. Similarly, despite its lack of performance in identifying cytoplasmic proteins in Gram-negative bacteria, Gpos-mPLoc (a complementary software package to Gneg-mPLoc) performs well for cytoplasmic protein samples from Gram-positive bacteria. To combine the strengths of these various predictive programs, we take advantage of PSO as a computational intelligence technique to optimize the weights associated with the different output classes for each predictive tool and combine their results to obtain a final decision. Generally, PSO has been proven to be an efficient optimization algorithm for finding an optimal solution in various fields by searching an entire multidimensional problem space. The advantages of PSO include its good robustness, simplicity, and fast convergence speed, with relatively few parameters to adjust [32–35].

As shown in Table 2, for both Gram-negative and Gram-positive bacteria, the PSO-based combination of predictors leads to a performance improvement over any single individual predictor.

3.2. Effect of PSO as a Combiner in PSO-LocBact. We also compared our PSO-based method with other consensus classifiers and the recently proposed single predictor called FUEL-mLoc [23]. Since no other consensus classifiers specifically designed for predicting the localization of bacterial proteins are available, various consensus classifiers using various fusion algorithms to combine a set of predictors (the set of predictors used for Gram-negative bacterial proteins consisted of CELLO, PSORTb 3.0, CELLO2GO, SOSUI-GramN, SLP-Local, ngLOC, Gneg-mPLoc, PSLpred, and LocTree3, and the set of predictors used for Gram-positive bacterial proteins consisted of CELLO, PSORTb 3.0, CELLO2GO, ngLOC, Gpos-mPLoc, and LocTree3) were implemented using the Weka machine learning workbench [36]. All consensus classifiers were trained on the training set using the 10-fold cross-validation strategy and then tested with the test sets. As shown in Table 3, our PSO-based tool shows high overall accuracy when compared with the other consensus classifiers.

Compared to the majority voting method, the PSO-based method yields increased prediction accuracies for secreted (extracellular), periplasmic, and cytoplasmic proteins in the Gram-negative bacterial protein datasets and for cell wall and extracellular proteins in the Gram-positive bacterial protein datasets. In the PSO-based method, an appropriate weight can be assigned to each class for each predictor instead of an equal weight for each predictor, which is especially important in the case of multiclass classification. Moreover, this method provides probabilistic scores indicating the confidence of the protein localization predictions. These probabilistic scores can be used to identify multiple locations of proteins. In the case of multilocation proteins, which are colocalized at or move between two or more different subcellular compartments, our method is able to contribute to the simultaneous prediction of multiple subcellular locations. For individual query sequences, the predicted location with the highest score should be assigned as the most promising location of a particular protein, while the second ranking can be suggested as an alternative location for such a multilocation protein.

3.3. Performance of PSO-LocBact under Different Circumstances. The choice of the individual predictors considered in a consensus classifier also affects the prediction results. Since, under most circumstances, users may not know the limitations and merits of individual predictors, the aim of this section is to investigate how well PSO-LocBact performs in terms of accuracy and robustness with a limited number of predictor programs. To this end, we designed 9 experimental cases to represent various circumstances to evaluate the performance of the proposed method by removing certain programs (based on the performance results from Table 2) and then investigating the effects of this removal on the final prediction results (see Table 4). In the first experimental case, PSORTb 3.0, which achieved the highest overall accuracy, was removed from the system. With the best predictor in the list removed, the PSO classifier needs to rely on other, less efficient tools. Its overall accuracy for Gram-negative bacteria in this case is slightly decreased to 97.67% compared to the result reported for PSO-LocBact with the all-program strategy in Table 2.

As the complement to the second experimental case, the third experiment was carried out by removing all predictors with an overall accuracy higher than 90%. The predictors removed in this case for Gram-negative bacteria were PSORTb 3.0 and CELLO2GO. As shown in Table 4, our PSO-LocBact can improve the prediction performance in this case. Each predictor included in this case achieves an overall accuracy of
In the sixth experimental case, the predictors with overall accuracies lower than 70% were removed: CELLO, SLP-Local, Gneg-mPLoc, and PSLpred for Gram-negative bacteria and Gpos-mPLoc and LocTree3 for Gram-positive bacteria.

less than 90%. By contrast, the overall prediction result of PSO-LocBact in this case is 90.69%, beyond the level attained by any of the individual predictors (CELO, SOSUI-GramN, SLP-Local, ngLOC, Gneg-mPLoc, PSLpred, and LocTree3).
bacteria. As shown in Table 4, the results for the Gram-positive experiment in this case are even better than those of PSO-LocBact with the all-program strategy, as reported in Table 2. His finding indicates that the combination of only a few efficient tools is also adequate to produce reliable solutions.

In experimental case 9 for Gram-positive bacteria, since Gpos-mPLoc is the only classifier with an accuracy of less than 60%, we could not test our model under this condition.

Based on these 9 different experiments carried out in this study to determine the effectiveness of the PSO-LocBact method under various circumstances, we conclude that the proposed method can provide users with more confidence in the obtained predictions. These results also confirm that PSO-LocBact can increase performance and/or provide more reliable prediction results in all experimental cases. Moreover, new prediction programs can be easily added to our method; thus, any novel predictors that may be developed in the future can be easily included to further improve the prediction accuracy.

<table>
<thead>
<tr>
<th>Location</th>
<th>Gram-negative bacterial proteins</th>
<th>Gram-positive bacterial proteins</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Extracellular region (%)</td>
<td>Outer membrane (%)</td>
</tr>
<tr>
<td></td>
<td>Colony (%)</td>
<td>Periplasm (%)</td>
</tr>
<tr>
<td></td>
<td>Inner membrane (%)</td>
<td>Cytoplasm (%)</td>
</tr>
<tr>
<td></td>
<td>Overall (%)</td>
<td></td>
</tr>
<tr>
<td>Experimental case 1</td>
<td>98.84</td>
<td>100</td>
</tr>
<tr>
<td>Experimental case 2 (&lt;90)</td>
<td>100</td>
<td>100</td>
</tr>
<tr>
<td>Experimental case 3 (&lt;90)</td>
<td>80.23</td>
<td>91.86</td>
</tr>
<tr>
<td>Experimental case 4 (&gt;80)</td>
<td>100</td>
<td>100</td>
</tr>
<tr>
<td>Experimental case 5 (&lt;80)</td>
<td>84.88</td>
<td>54.65</td>
</tr>
<tr>
<td>Experimental case 6 (&gt;70)</td>
<td>100</td>
<td>100</td>
</tr>
<tr>
<td>Experimental case 7 (&lt;70)</td>
<td>44.19</td>
<td>66.28</td>
</tr>
<tr>
<td>Experimental case 8 (&gt;60)</td>
<td>100</td>
<td>100</td>
</tr>
<tr>
<td>Experimental case 9 (&lt;60)</td>
<td>76.74</td>
<td>96.51</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Location</th>
<th>Extracellular region (%)</th>
<th>Colony (%)</th>
<th>Inner membrane (%)</th>
<th>Cytoplasm (%)</th>
<th>Overall (%)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Experimental case 1</td>
<td>96.05</td>
<td>100</td>
<td>100</td>
<td>98.73</td>
<td>98.71</td>
</tr>
<tr>
<td>Experimental case 2 (&gt;90)</td>
<td>96.34</td>
<td>96.39</td>
<td>100</td>
<td>100</td>
<td>98.20</td>
</tr>
<tr>
<td>Experimental case 3 (&lt;90)</td>
<td>89.02</td>
<td>54.22</td>
<td>98.82</td>
<td>100</td>
<td>85.63</td>
</tr>
<tr>
<td>Experimental case 4 (&gt;80)</td>
<td>97.56</td>
<td>96.39</td>
<td>100</td>
<td>100</td>
<td>98.50</td>
</tr>
<tr>
<td>Experimental case 5 (&lt;80)</td>
<td>78.05</td>
<td>43.37</td>
<td>91.76</td>
<td>100</td>
<td>78.44</td>
</tr>
<tr>
<td>Experimental case 6 (&gt;70)</td>
<td>94.73</td>
<td>100</td>
<td>100</td>
<td>100</td>
<td>98.71</td>
</tr>
<tr>
<td>Experimental case 7 (&lt;70)</td>
<td>68.29</td>
<td>12.05</td>
<td>100</td>
<td>100</td>
<td>70.66</td>
</tr>
<tr>
<td>Experimental case 8 (&gt;60)</td>
<td>97.56</td>
<td>96.39</td>
<td>100</td>
<td>100</td>
<td>98.50</td>
</tr>
<tr>
<td>Experimental case 9 (&lt;60)</td>
<td>NA</td>
<td>NA</td>
<td>NA</td>
<td>NA</td>
<td>NA</td>
</tr>
</tbody>
</table>


3.4. Comparison with State-of-the-Art Predictors and the Performance of PSO-LocBact on the Benchmark Dataset S. Note that, in our training and test datasets, we used a threshold of 90% instead of 25% sequence identity because we needed to increase the number of proteins for some classes for which only a limited number of proteins with
reviewed localization statuses were available in the database in order to be able to build a balanced training dataset, which is important for building a consensus predictor. Individual homolog features are not needed to train such a model for consensus prediction, unlike most individual predictor methods, which depend on homolog features for model training and thus need to consider the homology bias of the features. In addition, we included the well-known fair benchmark dataset $S$, which comprises proteins that share less than 25% identity, as our validation dataset to enable performance comparisons with various state-of-the-art methods.

Table 5 shows the performance of PSO-LocBact and various state-of-the-art predictors on dataset $S$, which is a widely used benchmark dataset. This dataset was constructed by the authors of [7, 9] and has been used to test various predictors, including iLoc-Gneg [24], Gram-LocEN [25], Gneg-PLoc [26], Gneg-mPLoc [7], and iLoc-GpLoc [27]. The overall accuracy of PSO-LocBact is 96.15% for Gram-negative bacterial proteins and 99.42% for Gram-positive bacterial proteins, higher than the values for the other state-of-the-art methods. In contrast to the dataset considered in the previous section, which is a balanced dataset, this benchmark consists of imbalanced data. Therefore, PSO-LocBact shows high performance on both balanced and imbalanced datasets.

4. Conclusions

With the growing number of research efforts employing various machine learning approaches to predict the subcellular localization of proteins, these tools can yield incongruent prediction results in some circumstances. In this paper, PSO-LocBact, a method of bacterial protein subcellular localization prediction based on the simple particle swarm optimization (PSO) technique, has been proposed to integrate the prediction results from existing predictors to provide more reliable predictions and increased accuracy under most circumstances. During testing, our proposed method achieved an overall prediction accuracy of over 98%. Hence, this method can provide researchers in the field with more reliable answers for protein localization together with probabilistic scores indicating the confidence of the results.

4.1. Software Package Applications. The PSO-LocBact method is a PSO method for combining the results of multiple classifiers for the prediction of protein subcellular localization in both Gram-negative and Gram-positive bacteria. This method is capable of generating final localization predictions based on protein sequence data. In particular, this method has been developed to address the inconsistency problems encountered in this task. Our recent work has focused on introducing a simple PSO method of optimizing the prediction results obtained from other applications. The software package is designed to be easy to understand and develop. In addition, users are able to use new datasets for training and testing, thus updating this software’s capabilities. By modifying the configuration file, users can reconfigure the software, optimize the weights for each predictor, add more result files to aid in prediction, and even set the basic PSO parameters. These configuration variables are shown in Table 6.
Data Availability

The training and test datasets supporting the analysis in this study are from previously reported studies and datasets, which have been cited. The software is available from the corresponding author upon request. http://www.ncrna-pred.com/psolocbact.htm.

Disclosure

The funders had no role in the design of the study; the collection, analysis, and interpretation of the data; or the writing of the manuscript.

Conflicts of Interest

The authors declare that there are no conflicts of interest regarding the publication of this paper.

Authors’ Contributions

Supatcha Lertampaiporn and Sirapop Nuannimnoi, the first two authors, should be regarded as joint first authors.

Acknowledgments

Sirapop Nuannimnoi was supported by Food Innopolis Grant P-17-50583. The authors acknowledge the use of a computing facility provided by King Mongkut’s University of Technology Thonburi through the “KMUTT 55th Anniversary Commemorative Fund.”

References


<table>
<thead>
<tr>
<th>Configuration variable</th>
<th>Value type</th>
<th>Default value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>w1</td>
<td>Float</td>
<td>0.9</td>
<td>Inertial weight value at the beginning of PSO</td>
</tr>
<tr>
<td>w2</td>
<td>Float</td>
<td>0.4</td>
<td>Inertial weight value at the end of PSO</td>
</tr>
<tr>
<td>c1i</td>
<td>Float</td>
<td>2.5</td>
<td>Cognitive coefficient value at the beginning of PSO</td>
</tr>
<tr>
<td>c1f</td>
<td>Float</td>
<td>0.5</td>
<td>Cognitive coefficient value at the end of PSO</td>
</tr>
<tr>
<td>c2i</td>
<td>Float</td>
<td>0.5</td>
<td>Social coefficient value at the beginning of PSO</td>
</tr>
<tr>
<td>c2f</td>
<td>Float</td>
<td>2.5</td>
<td>Social coefficient value at the end of PSO</td>
</tr>
<tr>
<td>Particle num</td>
<td>Integer</td>
<td>25</td>
<td>Number of particles generated in the swarm</td>
</tr>
<tr>
<td>MAXOBJ</td>
<td>Integer</td>
<td>1,000</td>
<td>Maximum number of allowable objective function calls</td>
</tr>
<tr>
<td>MAXITER</td>
<td>Integer</td>
<td>—</td>
<td>Maximum number of allowable iterations; if this value is set, MAXOBJ will be ignored</td>
</tr>
<tr>
<td>(Program_name)</td>
<td>String</td>
<td>(Gram-negative: CELLO, PSORTb 3.0, CELLO2GO, SOSUI-GramN, SLP-Local, ngLOC, Gneg-mPLoc, PSLpred, LocTree3; Gram-positive: CELLO, PSORTb 3.0, CELLO2GO, ngLOC, Gpos-mPLoc, LocTree3)</td>
<td>A list of names of the programs used to calculate the final result</td>
</tr>
<tr>
<td>(Weight)</td>
<td>Float</td>
<td>—</td>
<td>A list of weights given to represent the reliability of every program included</td>
</tr>
</tbody>
</table>


