Analytical Biochemistry 593 (2020) 113592

Contents lists available at ScienceDirect

Analytical Biochemistry

journal homepage: www.elsevier.com/locate/yabio

# Inspector: a lysine succinylation predictor based on edited nearest-neighbor undersampling and adaptive synthetic oversampling☆

Yan Zhu[a], Cangzhi Jia[a], Fuyi Li[b,c,**], Jiangning Song[b,c,*]

[a] School of Science, Dalian Maritime University, Dalian, 116026, China
[b] Monash Biomedicine Discovery Institute and Department of Biochemistry and Molecular Biology, Monash University, Melbourne, VIC, 3800, Australia
[c] Monash Centre for Data Science, Faculty of Information Technology, Monash University, Melbourne, VIC, 3800, Australia

ARTICLE INFO

*Keywords:*
Edited nearest-neighbor undersampling
Adaptive synthetic oversampling
Random forest

ABSTRACT

Lysine succinylation is an important type of protein post-translational modification and plays a key role in regulating protein function and structural changes. The mechanism and function of succinylation have not been clarified. The key to better understanding the precise mechanism and functional role of succinylation is the identification of lysine succinylation sites. However, conventional experimental methods for succinylation identification are often expensive, time-consuming, and labor-intensive. Therefore, the new development of computational approaches to effectively identify lysine succinylation sites from sequence data is much needed. In this study, we proposed a novel predictor for lysine succinylation identification, Inspector, which was developed by using the random forest algorithm combined with a variety of sequence-based feature-encoding schemes. Edited nearest-neighbor undersampling method and adaptive synthetic oversampling approach were employed to solve dataset imbalance, and a two-step feature-selection strategy was applied to optimize the feature set for training the accuracy of the prediction model. Empirical studies on performance comparison with existing tools showed that Inspector was able to achieve competitive predictive performance for distinguishing lysine succinylation sites.

## 1. Introduction

Post-translational modifications (PTMs) refer to the covalent addition and enzymatic modifications of proteins during or after protein biosynthesis, and they often play important roles in modifying protein functions and regulating gene expression [1]. There are multiple types of PTMs that can occur at the lysine residues of proteins, including acetylation, glycation, ubiquitination, and succinylation [2], with protein succinylation ubiquitous in both eukaryotes and prokaryotes. The modified proteins are distributed throughout the cell membrane, cytoplasmic matrix, different organelles, and various parts of the nucleus [3–5], involved in various metabolic reactions, including glucose metabolism, the tricarboxylic acid (TCA) cycle, and fatty acid metabolism, and are closely related to the activities of living organisms [6,7]. Succinylation can occur in the active site of homoserine trans-succinylase, with a succinyl group transferred from succinyl-CoA to homoserine during the intermediate reaction. Succinylation is evolutionarily conserved and commonly found in response to various physiological conditions [8]. Additionally, lysine succinylation sites can potentially regulate cellular enzymes and metabolism, such as the TCA cycle, amino acid degradation, nitrogen metabolism, and fatty acid oxidation [9]. Histone succinylation, particularly, contributes to the regulation of protein structure and function [10].

The key to understanding the mechanisms of lysine succinylation is the identification of lysine succinylation sites [11]. Conventional experimental techniques, such as site-directed mutagenesis, mass spectrometry, and creation of a peptide library, are costly and labor-intensive [12]. Enzyme optimization, another experimental method, is also too difficult to conduct [13]. Therefore, computational methods if capable of accurately predicting lysine succinylation sites can provide a great complementary approach to aid succinylation-site identification [14]. Recently, many computational predictors have been developed and published for this reason. Xu et al. (2015) proposed iSuc-PseAAC

☆ This work was supported by Fundamental Research Funds for the Central Universities (3132019175, 3132019323), the National Natural Science Foundation of Liaoning Province (20180550307).

** Corresponding author. Monash Biomedicine Discovery Institute and Department of Biochemistry and Molecular Biology, Monash University, Melbourne, VIC, 3800, Australia.

* Corresponding author. Monash Biomedicine Discovery Institute and Department of Biochemistry and Molecular Biology, Monash University, Melbourne, VIC, 3800, Australia.

*E-mail addresses:* Fuyi.Li1@monash.edu (F. Li), Jiangning.Song@monash.edu (J. Song).

https://doi.org/10.1016/j.ab.2020.113592
Received 19 October 2019; Received in revised form 14 January 2020; Accepted 17 January 2020
Available online 20 January 2020
0003-2697/ © 2020 Published by Elsevier Inc.

[15] based on pseudo-amino acid composition encoding and support vector machine (SVM). Jia et al. [16] developed iSuc-PseOpt based on pseudo-amino acid composition encoding and *k*-nearest neighbor (KNN) algorithm. Hasan et al. trained two random forest (RF) classifiers [SuccinSite [17], and SuccinSite2.0 [18]] via merging multiple features. Abdollah et al. (2018) combined secondary structure features and a position-specific scoring matrix to train the predictor SSEvol-Suc [19]. López et al. [20] identified the evolutionary information around lysine succinylation sites and combined these with the predicted structural features to train an SVM-based predictor called Success. Ning et al. [21] developed an ensemble SVM-based method called PSuccE that employed multiple features with an information-gain feature-selection method. More recently, Hasan and Kurata [22] developed an RF and logistic regression-based tool called GPSuc by integrating multiple sequence features. Admittedly, considerable progress has been made in the development of succinylation site prediction methods. However, due to the low percentage of PTMs sites in annotations, these predicting PTMs still have an imbalance problem that existing methods have not adequately addressed and moreover attenuates the performance of the current tools. Therefore, to develop more accurate predictors using algorithms capable of dealing with unbalanced datasets is inevitable.

In this study, we proposed a novel computational approach called Inspector to tackle these problems and improve the predictive performance of lysine succinylation sites by integrating six sequence features. Inspector used edited nearest-neighbor (ENN) undersampling and adaptive synthetic sampling approach (ADASYN) oversampling techniques to balance the training dataset. Inspector also employed the RF algorithm that combined with a two-step feature-selection strategy to boost predictive performance. A 10-fold cross-validation test demonstrated Inspector's predictive capacity of with 90% prediction accuracy, and the AUC in ROC analysis was 0.96.

## 2. Materials and methods

The overall framework of Inspector is illustrated in Fig. 1 in four major steps: data collection and preprocessing, feature extraction and dataset balancing, feature selection and model construction, and model evaluation and validation. In the first step, we collected the training and test datasets from UniProtKB/Swiss-Prot [21] and NCBI (http://www.ncbi.nlm.nih.gov/protein/) databases. Then in the second step, we applied six different types of sequence-encoding schemes to extract sequence-based features, and two dataset balancing techniques were used to balance the training dataset. A two-step feature-selection algorithm combined with the RF algorithm was employed in the third step to train the optimal prediction model. The optimized RF model was subsequently used for a 10-fold cross-validation. In the fourth step, the trained RF model was validated using an independent test dataset, and the predictive performance was compared with the existing tools.

### 2.1. Datasets

We used datasets constructed by Ning et al. [21] to train and validate the Inspector. The Succinylated proteins verified by the experiments were collected from the UniProtKB/Swiss-Prot and NCBI protein-sequence databases, and redundant sequences were removed by using CD-HIT [23] with a sequence-identity threshold < 30%. This process left 2322 succinylated proteins, with 5009 experimentally verified lysine succinylation sites. Lysine sites (K) not annotated as succinylation sites were used as negative samples. We then randomly selected 2198 succinylated proteins as a training dataset, containing 4755 succinylation sites (as positive training samples) and 50,547 non-succinylation sites (as negative training samples). The remaining 124 succinylated proteins with 254 succinylation sites (as positive test samples) and 2977 non-succinylation sites (as negative test samples) were used as an independent test dataset. Each sample, $S_L(K)$, can be generally represented as:

$$S_L(K) = R_{-L}R_{-(L-1)}...R_{-2}R_{-1}KR_{+1}R_{+2}...R_{+(L-1)}R_{+L} \quad (1)$$

where $R_{-L}$ denotes the $L$-th downstream amino acid residue from the center site, $K$ (lysine), $R_{+L}$ denotes the $L$-th upstream amino acid residue from the center $K$ site, and the center $K$ site represents the lysine residue:

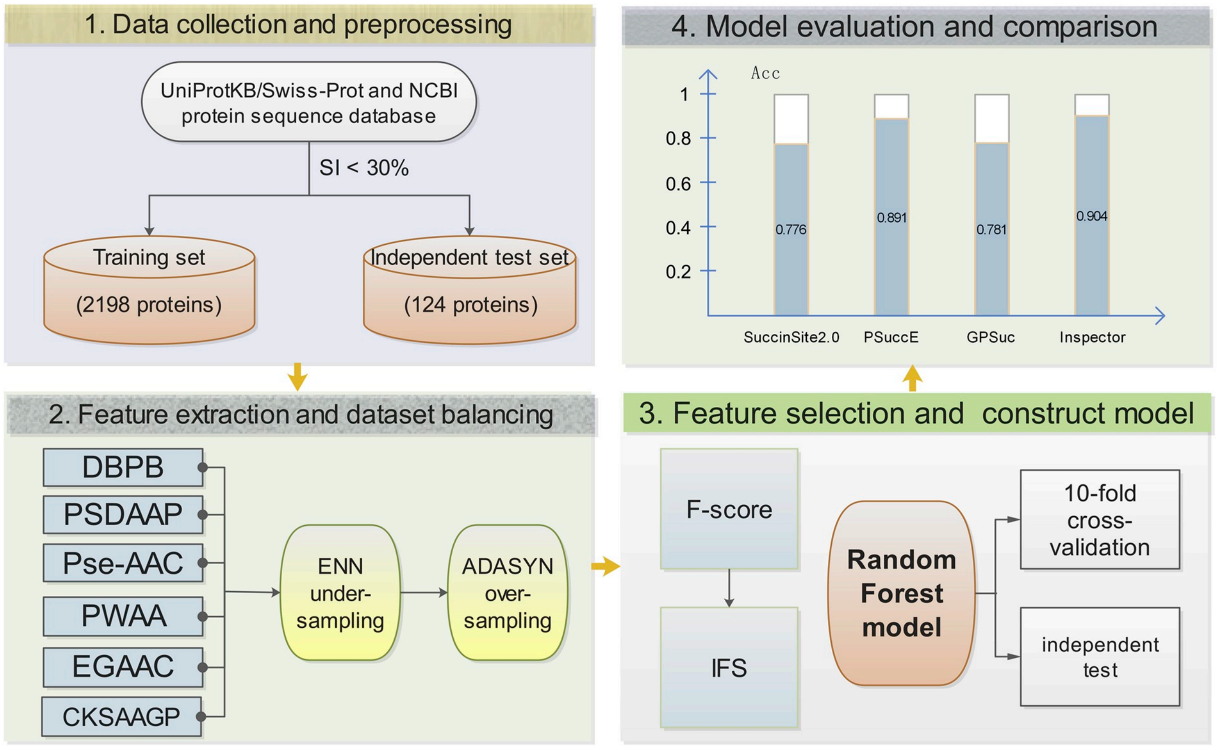

**Fig. 1.** Overall framework of Inspector.

$$S_L(K) \in \begin{cases} S_L^{+}(K) & \textit{if K is the succinylation site} \\ S_L^{-}(K) & \textit{otherwise} \end{cases} \tag{2}$$

$S_L^{+}(K)$ and $S_L^{-}(K)$ represent the positive and negative samples, respectively. And we selected $L = 17$ for this study. Thus, each sample was represented as a sequence fragment with 35 amino acid residues, and we used pseudo amino acid "X" when the number of flanking residues was < L.

## 2.2. Feature-extraction strategy

### 2.2.1. Bi-profile Bayes (BPB)

The BPB feature-extraction method has been profoundly useful in many fields of bioinformatics, including PTM sites prediction [24,25], promoters [26], and protease substrates and cleavage sites [27]. Especially, it considers position-specific information of each amino acid in both positive and negative training samples simultaneously. We defined the amino acid position-specific tendency matrices $P$ and $N$ as:

$$P = \begin{pmatrix} p_{1,1} & \cdots & p_{1,Z} \\ \vdots & \ddots & \vdots \\ p_{21,1} & \cdots & p_{21,Z} \end{pmatrix} \tag{3}$$

$$N = \begin{pmatrix} n_{1,1} & \cdots & n_{1,Z} \\ \vdots & \ddots & \vdots \\ n_{21,1} & \cdots & n_{21,Z} \end{pmatrix} \tag{4}$$

In these matrices, $Z$ indicates the length of peptide fragment after omitting amino acid K in the middle position. $p_{i,j}$ represents the frequency of the $i$-th amino acid in the $j$-th position of the positive training samples. And $n_{i,j}$ represents the frequency of the $i$-th amino acid in the $j$-th position of the negative training samples. Each sample can be encoded as

$$V_{BPB} = (v_1,\ v_2,\ \ldots,\ v_Z,\ v_{Z+1},\ v_{Z+2},\ \ldots,\ v_{2Z}) \tag{5}$$

In this equation, $v_1,\ v_2,\ \ldots,\ v_Z$ represents the posterior probability of the present amino acid at the $i$-th position in all positive training samples, and $v_{Z+1},\ v_{Z+2},\ \ldots,\ v_{2Z}$ denotes the posterior probability of the present amino acid at the $i$-th position in all negative training samples. The posterior probability can be estimated by the occurrence of each amino acid at each position in the training datasets (*i.e.,* matrices $P$ and $N$) [28].

### 2.2.2. Double BPB (DBPB)

Inspired by the superiority of BPB, DBPB [29] considers the frequency of every di-amino acids in each position in both positive and negative samples. Given one protein peptide sequence, $S$, the DBPB feature vector can be defined as:

$$V_{DBPB} = (c_1,\ c_2,\ \ldots,\ c_{Z-1},\ c_Z,\ c_{Z+1},\ \ldots,\ c_{2(Z-1)}) \tag{6}$$

where $Z$ is the length of peptide fragment after omitting amino acid K in the middle position. $c_1,\ c_2,\ \ldots,\ c_{Z-1}$ denotes the posterior probability of di-amino acids present at the $i$-th position in all positive samples. And $c_Z,\ c_{Z+1},\ \ldots,\ c_{2(Z-1)}$ denotes the posterior probability of di-amino acids present at the $i$-th position in all negative samples.

### 2.2.3. Position-specific di-amino acid propensity (PSDAAP)

The posterior probability of every two adjacent amino acids at each position in the positive peptide sequence datasets is subtracted from the posterior probability in the negative peptide sequence datasets [30]. The feature vector of PSDAAP is defined as follows:

$$V_{PSDAAP} = \begin{pmatrix} p_{1,1} - n_{1,1} & \ldots & p_{1,Z-1} - n_{1,Z-1} \\ \vdots & \ddots & \vdots \\ p_{21^2,1} - n_{21^2,1} & \ldots & p_{21^2,Z-1} - n_{21^2,Z-1} \end{pmatrix} = \begin{pmatrix} f_{1,1} & \cdots & f_{1,Z-1} \\ \vdots & \ddots & \vdots \\ f_{21^2,1} & \cdots & f_{21^2,Z-1} \end{pmatrix} \tag{7}$$

where $p_{i,j}$ represents the posterior probability of each di-amino acid at each position in the positive training dataset, $n_{i,j}$ represents the posterior probability of each di-amino acid at each position in the negative training dataset, and $Z$ is the length of the peptide fragment ($i.e. Z = 35$).

### 2.2.4. Pseudo-amino acid composition (Pse-AAC)

In order to take both local and global sequence-order information of a protein sequence into accounts, Pse-AAC [31] was developed and widely used to represent protein sequences [32,33]. Pse-AAC indicates the protein sequence as a $20 + \lambda$-dimensional vector, where the first 20 dimensions include the amino acid composition information, and the latter λ-dimensional elements represent a series of physical and chemical factors. This formula can effectively avoid the information loss in both amino acid order as well as physical and chemical information loss in protein sequence. The Pse-AAC feature vector for a protein sequence is given by:

$$V_{Pse-AAC} = (w_1,\ w_2,\ \ldots,\ w_{20},\ w_{20+1},\ w_{20+2},\ \ldots,\ w_{20+\lambda}) \tag{8}$$

where

$$w_x = \begin{cases} \dfrac{f_x}{\sum_{x=1}^{20} f_x + w\sum_{k=1}^{\lambda}\theta_k} & (1 \le x \le 20) \\ \dfrac{w\theta_{x-20}}{\sum_{x=1}^{20} f_x + w\sum_{k=1}^{\lambda}\theta_k} & (20 + 1 \le x \le 20 + \lambda) \end{cases} \tag{9}$$

$f_x (x = 1,\ 2,\ \ldots,\ 20)$ represents the normalized occurrence frequency of the 20 amino acids in the protein fragment. Parameter λ is an integer representing the highest counted rank (or tier) of the correlation along a protein sequence, $w$ is the weight factor used to improve accuracy (range: 0 to 1), and $\theta_k (k = 1,\ 2,\ \ldots,\ \lambda)$ is the $j$-tier correlation factor reflecting the sequence-order correlation between all the $j$-th most contiguous residues along a protein chai. $\theta_k$ is defined as:

$$\theta_k = \frac{\sum_{i=1}^{L-k} \theta(R_i,\ R_{i+k})}{L - k} \quad (k \le \lambda) \tag{10}$$

the correlation function $\theta(R_i,\ R_{i+k})$ is given by:

$$\theta(R_i,\ R_{i+k}) = \frac{1}{\mu}\sum_{j=1}^{\mu} (I_j(R_i) - I_j(R_{i+k})) \tag{11}$$

where $\mu$ is the number of physicochemical indices considered, and $I_j(R_i)$ is the $j$-th physicochemical index value of amino acid $R_i$. It should be noted that before substituting the physicochemical indices values into Eq. (11), they were subject to a standard conversion described by the following equation:

$$I_j(A_i) = \frac{I'(A_i) - \sum_{m=1}^{20}\frac{I'(A_i)}{20}}{\sqrt{\dfrac{\sum_{k=1}^{20} I_j'(x_k) - \sum_{m=1}^{20}\frac{I'(x_m)}{20}}{20}}} \tag{12}$$

where $I_j'(A_i)$ is the $j$-th original physicochemical value of the $i$-th amino acid $A_i$, and $x_k$ and $x_m$ are the 20 amino acids (m, k = 1,2, …,20). Here, amino acid "X" is omitted.

### 2.2.5. Position-weight amino acid (PWAA) composition

PWAA [34] is used to avoid losing the sequence-order information of amino acid residues around certain sites. The position information of the amino acid in the sliding window can be calculated by the following formula:

$$v_i = \frac{1}{G(G+1)} \sum_{j=-G}^{G} x_{i,j}\left(j + \frac{|j|}{G}\right) \tag{13}$$

where $G$ denotes the number of upstream or downstream residues from the center site in the sliding window, and $x_{i,j} = 1$ if $a_i$ is the $j$-th residue in the sliding window, otherwise $x_{i,j} = 0$.

### 2.2.6. Enhanced grouped amino acid composition (EGAAC)

EGAAC feature extraction [35] uses five physicochemical properties of amino acids (*e.g.*, hydrophobicity, charge, and molecular size), and categorizes the 20 amino acid types into five classes: aliphatic group (g1): {G, A, V, L, M, I}; aromatic group (g2): {F, Y, W}; positive charge group (g3): {K, R, H}; negative charge group (g4): {D, E}; and uncharged group (g5): {S, T, C, P, N, Q}. EGAAC is designed to calculate the frequency of each amino acid group defined as follows:

$$V_{EGAAC}(g,\ w) = \frac{N(g, w)}{N(w)},\ \ g \in \{g1,\ g2,\ g3,\ g4,\ g5\},$$
$$w \in \{w1,\ w2,\ ...,\ w31\} \tag{14}$$

where $N(g, w)$ is the number of amino acids in group $g$ within the sliding window $w$, and $N(w)$ is the size of the sliding window $w$. We applied the *iLearn* package [36] to calculate the EGACC features.

### 2.2.7. Composition of k-spaced amino acid group pairs (CKSAAGP)

CKSAAGP [35] calculates the frequency of amino acid group pairs separated by any $k$ residues. Taking $k = 0$ as an example, there are 25 0-spaced group pairs (*i.e.*, {g1g1, g1g2, g1g3, ..., g5g5}). Thus, the feature vector of CKSAAGP can be encoded as follows:

$$V_{CKSAAGP} = \left(\frac{N_{g1g1}}{N_{total}},\ \frac{N_{g1g2}}{N_{total}},\ \frac{N_{g1g3}}{N_{total}},\ ...,\ \frac{N_{g5g5}}{N_{total}}\right) \tag{15}$$

The value of each descriptor denotes the composition of the corresponding residue group pair in a protein or peptide sequence. For example, if residue group pair $g1g1$ appears $m$ times in the protein fragment, the composition of residue pair $g1g1$ is equal to $m$ divided by the total number of 0-spaced residue pairs, $N_{total}$, in the protein. For this study, we used $k = 0, 1, 2, 3, 4,$ and 5 and values for $N_{total} = L-1,\ L-2,\ L-3,\ L-4,\ L-5,$ and $L-6$, respectively, for a protein of length $L$.

## 2.3. Dataset balancing

The collected benchmark dataset contains 4755 succinylated sites as positive samples and 50,549 non-succinylated sites as negative samples. Both samples are highly unbalanced (the number of negative samples is 10-fold that of the positive samples). This class-imbalance training dataset will lead the trained classifiers to achieve better predictive performance for the majority class while weakening the performance of the rare class [37]. However, in this study, the rare class (positive sample) is more important. We employed two sampling algorithms (ENN undersampling and ADASYN oversampling) to effectively address the class-imbalance problem, which is the key to training accurate models.

### 2.3.1. ENN undersampling

Random selection of a certain number of negative samples is a viable approach for dealing with the imbalance problem. However, the random undersampling method does not consider the distribution of samples and might lose important information concerning all of the majority samples. To address such problems, Wilson et al. [38] proposed an ENN rule, where if the KNN samples of a sample differ from their own category, they are deleted. This study used experimentally verified data to confirm that if more than two of the 37 neighbors of the negative sample are positive samples, the negative sample will be deleted.

### 2.3.2. ADASYN oversampling

ADASYN [39] can adaptively synthesize part of the minority class samples according to the distribution and automatically decides how many synthetic samples need to be produced for each minority class sample. Compared with the traditional SMOTE [40,41] sampling technique, ADASYN uses the definition of a density distribution which was deemed as a criterion to automatically decide the number of synthetic samples for each minority data example. For specific definition, please refer to Ref. [39].

In this study, we implemented the balanced sample algorithm with MATLAB. **Algorithm 1** describes the detailed procedure of our optimization approach for the balanced training set. The input of **Algorithm 1** is the initial positive set $P$, and the initial negative set $N$. In the first step, the $getNeighbors()$ function selects $k_1$ neighbors ($k_1 = 37$ in this study) from both $P$ and $N$ for each negative sample in $N$, and these $k_1 \times |N|$ neighbors are saved in $neighbors_N$, where $|N|$ denotes the number of features in the negative feature set $N$. In the following, let $|*|$ denotes the number of elements in set *. Steps 2 through 7 are conducted to filter negative samples in the initial negative set $N$ based on the number of positive samples among $k_1$ neighbors in each negative sample. In step 3, the $posNum()$ function is used to calculate the number of positive samples, $numPos(i)$, among the $k_1$ neighbors in the $i$-th negative sample. If $numPos(i) > e$ ($e = 1$ in this study), which means the $i$-th negative sample has more than $e$ neighbors that are positive samples, the $i$-th negative sample will be deleted from the initial negative sample set (step 5). After completing steps 2 through 7, we obtain a new negative dataset $newN$. In step 9, we first calculate the number of samples, $C$, that needs to be oversampled. The $getNeighbors()$ function then selects $k_2$ neighbors ($k_2 = 5$ in this study) from both $P$ and $newN$ for each positive sample in $P$, and these $k_2 \times |P|$ neighbors are saved in $neighbors_P$ in step 10. Steps 11 through 14 are conducted to calculate the $\rho(j)$ value for each sample in $P$, where $\rho(j)$ represents the proportion of negative samples in $k_2$ neighbors of the $j$-th positive sample. In step 12, the $negNum()$ function is used to calculate the number of negative samples (appearing in $newN$), $numNeg(j)$, among the $k_2$ neighbors in the $j$-th positive sample. Steps 15 through 23 calculate the number of positive samples to be synthesized and generate the synthetic samples. In step 16, the $standardize()$ function is used to normalize the $\rho$ value of each positive sample. The number of synthetic samples, $c_s$, that need to be generated for positive samples are then calculated in step 17. Steps 18 through 22 generate the synthetic samples. In step 19, the $getRandom()$ function is applied to randomly select sample $x_{ts}$ from the $k_2$ neighbors of the positive sample $x_{ts}$. The synthetic sample $PP(t)$ is then generated in step 20, where $\beta$ is a random number, $\beta \in [0,1]$. In step 21, the synthetic sample is added to the positive sample set, $P$, and *newP* and *newN* are returned as the outputs of **Algorithm 1**.

**Algorithm 1.** Framework of the sampling process.

**Algorithm 1** Framework of the sampling process.

```
Input:
        Initial positive set, $P$ ;
        Initial negative set, $N$ ;
Output:
        Positive feature set after balancing, $newP$
        Negative feature set after balancing, $newN$
1:  $getNeighbors_N = getNeighbors(P, N, k_1)$ ;
2:  for each $i \in [1, |N|]$ do
3:      $numPos(i) = posNum(neighbors_N(i))$ ;
4:      if $numPos(i) > e$
5:          $N = removeSample(i, N)$ ;
6:      end if;
7:  end for;
8:  $newN = N$ ;
9:  $C = |newN| - |P|$ ;
10: $neighbors_P = getNeighbors(P, newN, k_2)$ ;
11: for each $j \in [1, |P|]$ do
12:     $numNeg(j) = negNum(neighbors_P(j), newN)$ ;
13:     $\rho(j) = numNeg(j) / k_2$ ;
14: end for;
15: for each $s \in [1, |P|]$ do
16:     $\overline{\rho}(s) = s\tan dardize(\rho(s))$ ;
17:     $c_s = \overline{\rho}(s) \times C$ ;
18:     for each $t \in [1, c_s]$ do
19:         $x_{ts} = getRandom(neighbors_p, P(x_s))$ ;
20:         $PP(t) = x_s + (x_{ts} - x_s) \times \beta$ ;
21:         $P = [P, PP(t)]$;
22:     end for;
23: end for;
24: $newP = P$ ;
25: return $newP$, $newN$ ;
```

as follows:

$$F-score(j)=\frac{\left(\overline{x}_j^{(+)}-\overline{x}_j\right)^2+\left(\overline{x}_j^{(-)}-\overline{x}_j\right)^2}{\frac{1}{m^+-1}\sum_{k=1}^{m^+}\left(\overline{x}_{k,j}^{(+)}-\overline{x}_j^{(+)}\right)^2+\frac{1}{m^--1}\sum_{k=1}^{m^-}\left(\overline{x}_{k,j}^{(-)}-\overline{x}_j^{(-)}\right)^2} \tag{16}$$

## 2.4. Feature optimization

The extracted features exhibit high dimensionality with a total of 452 characteristics. The initial feature set can contain some redundant and noisy features that possibly lead to computationally intensive training process and overfitting. Therefore, it is a common practice to employ feature-selection techniques to reduce the dimensionality of the initial feature set by eliminating features without contributing to the prediction. In this study, we applied a two-step feature-selection algorithm [42,43] (based on F-score [44]) and incremental feature selection (IFS) to select the optimal feature subset for training more effective models. With experiments, the performance of the predictive model was significantly improved by reducing the spatial dimensions of the feature space.

In the first step, we used the F-score feature-selection algorithm to rank all of 452 characteristics. The F-score of the $j$-th feature is defined

where $m^+$ denotes the number of positive samples, $m^-$ denotes the number of negative samples. $\overline{x}_j^{(+)}$ represents the average value of the $j$-th characteristic in the positive training dataset. $\overline{x}_j^{(-)}$ represents the average value of the $j$-th characteristic in the negative training dataset. $\overline{x}_j$ represents the average values of the $j$-th feature in the combined positive and negative datasets. $\overline{x}_{k,j}^{(+)}$ denotes the $j$-th feature of the $k$-th positive sample, and $\overline{x}_{k,j}^{(-)}$ denotes the $j$-th feature of the $k$-th negative sample.

A larger F-score value indicates that the feature is distinguishing well between positive and negative samples and is more useful for classification. By equation (16), the top 432 characteristics were ranked and considered as the optimal feature candidates (abbreviated as OFC) in this study.

In the second step, to determine the optimal feature subset in the

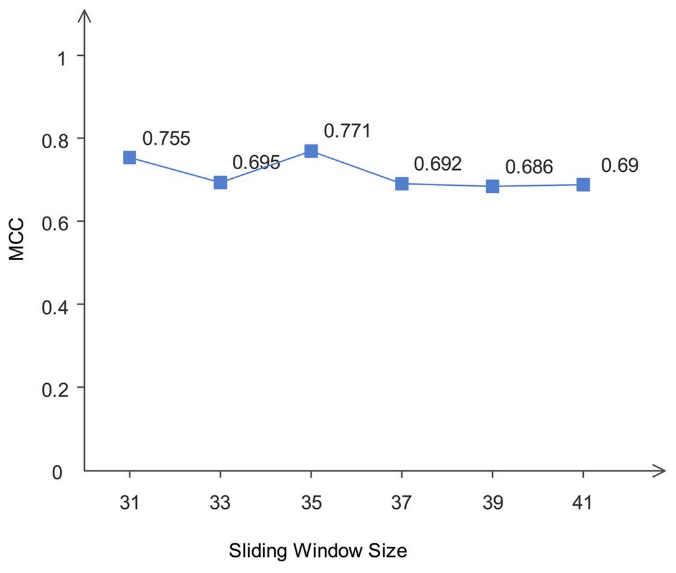

**Fig. 2.** Predicted MCC of five-fold cross-validation based on use of the training dataset and a sliding-window size ranging from 31 to 41.

OFC, we applied the IFS strategy based on the RF classifier and five-fold cross-validation test to find the optimal feature subset included in the OFC. Briefly, the IFS constructs *n* feature subsets by adding one feature at a time from the OFC to the candidate feature subset, ***F***. The *i*-th feature subset is defined as:

$$F = \{f|f_1, f_2, ..., f_i\} \tag{17}$$

where $f_i$ is the *i*-th feature from the OFC. Each feature set, *F*, was tested using the RF classifier and evaluated based on five-fold cross-validation test to avoid overfitting. This process was repeated for five rounds. The resulting feature set with the highest AUC value among the 432 AUC values was selected as the optimal feature set.

## 2.5. Model training

RF [45] is an effective ensemble-learning method for classification and has been widely used in protein bioinformatics [46–50]. A RF classifier comprises a collection of decision-tree classifiers, where each tree is trained with a randomly selected subset of samples. The decision tree is grown as follows. If *N* samples are randomly selected with replacement from the *F* features, the best split node is selected from among the *F* features, and the decision tree is then grown as large as possible without pruning. The construction of the forest is generalized based on the most votes given by all of the individual trees. RF is relatively robust to noise and outliers. Here, we used the RF algorithm to classify the succinylation and non-succinylation sites, as well as for the second-step feature selection and optimization of window sizes.

## 2.6. Performance assessment

To evaluate the prediction model, we performed *k*-fold cross-validation and independent dataset tests [13], and calculated four commonly used performance measurements [51–61]: sensitivity (Sn), specificity (Sp), accuracy (Acc), and Matthew's correlation coefficient (MCC). They are defined as:

$$\begin{cases} Sn = 1 - \frac{N_-^+}{N^+} & 0 \le Sn \le 1 \\ Sp = 1 - \frac{N_+^-}{N^-} & 0 \le Sp \le 1 \\ Acc = 1 - \frac{N_-^+ + N_+^-}{N^+ + N^-} & 0 \le Acc \le 1 \\ MCC = \frac{1 - \left(\frac{N_-^+}{N^+} + \frac{N_+^-}{N^-}\right)}{\sqrt{\left(1 + \frac{N_+^- - N_-^+}{N^+}\right)\left(1 + \frac{N_-^+ - N_+^-}{N^-}\right)}} & -1 \le MCC \le 1 \end{cases} \tag{18}$$

where $N^+$, $N_-^+$, $N^-$, and $N_+^-$ represent the numbers of positives, false negatives, negatives, and false positives, respectively.

Additionally, we plotted the ROC curves and calculated the area under curve (AUC) values in order to evaluate our model performance.

## 3. Results and discussion

### 3.1. Determination of the sliding-window sizes

Because different sliding-window size can affect predictive performance of identifying the lysine succinylated sites, it is necessary to select the optimal window size. Therefore, we used the same processing technique for the training samples for all odd-numbered window sizes between 31 and 41 (*i.e.,* 31, 33, 35, 37, 39, and 41). We then obtained the feature matrix for a training sample of six window sizes using a consistent feature-extraction method (the combination of the six types of features introduced in Section 2.2). Then we used RF classifier to build the prediction model that was tested on five-fold cross-validation. Each window size was evaluated according to the five indicators (Sn, Sp, Acc, MCC, and AUC). The results are listed in Table S1. Ultimately, we determined the optimal window size based on the MCC value that reflects the reliability and comprehensive predictive ability of the model (the closer to 1, the more reliable the result) in this study. Fig. 2 shows a line graph of the MCC values according to six window sizes. A window size of 35 returned the highest value of 0.771. Moreover, we analize the compositional biases around succinylation sites as compared with non-succinylation sites using two-sample logo and showed the statistical results in Fig. 3.

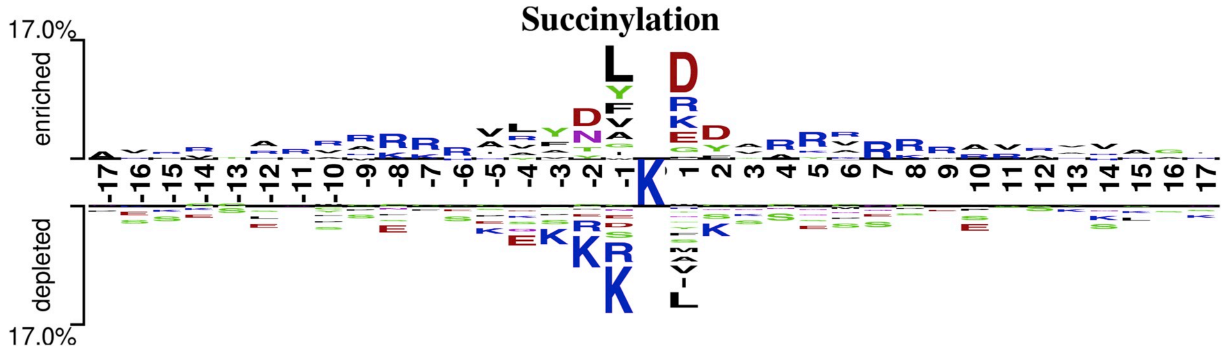

**Fig. 3.** Two-sample Logo (http://www.twosamplelogo.org/) of the compositional biases around succinylation sites as compared with non-succinylation sites ($P < 0.05$; *t*-test). The position between the compositional amino acids of the succinylated and non-succinylated peptides differed greatly for those located in positions from −17 to −1 and + 1 to +17.

**Table 1**
Performance of different sampling methods according to 5-fold cross validation.

| Sampling method | Sn | Sp | Acc | MCC | AUC |
|---|---|---|---|---|---|
| No sampling | 1.000 | 0.008 | 0.915 | 0.082 | 0.674 |
| RD | 0.777 | 0.754 | 0.765 | 0.531 | 0.845 |
| ENN | 0.864 | 0.865 | 0.864 | 0.728 | 0.942 |
| RD + SMOTE | 0.626 | 0.888 | 0.801 | 0.538 | 0.863 |
| RD + ADASYN | 0.608 | 0.875 | 0.786 | 0.504 | 0.850 |
| ENN + SMOTE | 0.781 | 0.925 | 0.880 | 0.717 | 0.941 |
| ENN + ADASYN | 0.868 | 0.914 | 0.900 | 0.771 | 0.961 |

Note: RD-random undersampling; ENN- ENN undersampling; SMOTE-SMOTE oversampling.

## 3.2. Analysis of two-step feature selection

After the determination of the window size at 35, we obtained a 452-dimension feature vector for each sample. To effectively select a candidate feature subset, we ranked the 452 components of the feature vector according to F-score and gradually reduced the dimensions of the characteristics in steps of 20 (*e.g.,* 452, 432, …, 272). The 10 feature subsets then underwent five rounds of 5-fold cross-validation, respectively. The average results were given in Table S2. At 432 dimensions, with the highest MCC value obtained of 0.773. Then, we reserved the 432-dimension feature subset for further training and testing models. The changes of each type of feature dimension is described in Table S3, which shows that the original PSDAAP had 33-dimensional features after two-step feature selection. We can conclude that this type of characteristic is beneficial for succinylation classification.

## 3.3. Analysis of the validity of ENN and ADASYN resampling methods

During the resampling process, the number of samples remaining after balancing can be calculated as follows [41]:

$$N = round\,[(k_1 \times n_0) - (k_0 \times n_1)] \quad (19)$$

where $k_0$ and $k_1$ are the percentages of negative and positive samples in the training dataset, and $n_0$ and $n_1$ are the number of negative and positive samples in the training dataset. With experiment on 5-fold cross-validation, $k_0 = 0.72$ and $k_1 = 0.28$ achieved the best MCC with $N = 10{,}729$. Due to the limitation of $k$ in the KNN used in the undersampling method, the balanced sample number of 10,729 can't be accurately obtained. However, while selecting the window size, we made the sample size per window size as close to $N$ as possible. The number of training samples after balancing for each window size is shown in Table S4.

In addition, we used five rounds of 5-fold cross-validation to test the validity of ENN and ADASYN resampling methods based on a comparison with other sample processing techniques. Table 1 shows the averages results of 5-fold cross-validation, including no sampling, only random undersampling, only ENN undersampling, random undersampling combined with SMOTE oversampling, random undersampling combined with ADASYN oversampling, ENN undersampling combined with SMOTE oversampling, and ENN undersampling combined with ADASYN oversampling, respectively. For random undersampling and ENN undersampling, we selected the same size of negative training samples as the original positive training samples to achieve the best balance. As expected, Table 1 showed that the MCC of no sampling were the worst, 0.082; the random and ENN undersampling, both improved prediction results, with MCC of 0.531 and 0.728, respectively. By comparison, ENN undersampling returned better results than random undersampling. Given that the negative training dataset (50,549) is more than 10-fold larger than the positive training data (4,755), only oversampling is inadequate. And the newly generated data will overwhelm the original data. Therefore, we used combinations of SMOTE oversampling and ENN undersampling, resulting in the highest Sp values of 0.925, whereas other values were worse than those obtained using ENN undersampling and ADASYN oversampling.

**Table 2**
Performance comparisons of different classification algorithms using the training dataset according to 10-fold cross-validation.

| Algorithm | Sn | Sp | Acc | MCC |
|---|---|---|---|---|
| SVM | 0.863 | 0.765 | 0.796 | 0.588 |
| NB | 0.868 | 0.592 | 0.678 | 0.429 |
| KNN | **0.953** | 0.612 | 0.718 | 0.527 |
| DA | 0.907 | 0.829 | 0.853 | 0.696 |
| RF | 0.876 | **0.917** | **0.904** | **0.781** |

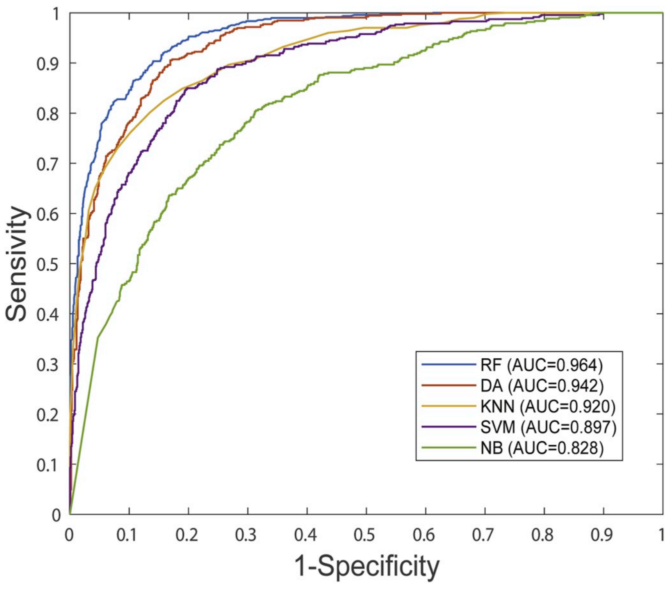

**Fig. 4.** ROC curves for the five classifiers according to 10-fold cross-validation.

## 3.4. Analysis of the effectiveness of RF classifiers

To evaluate the effectiveness of RF classification for predicting succinylation sites, four commonly used classifiers: SVM, naive Bayes (NB), KNN and discriminant analysis (DA) were adopted to predict lysine succinylation sites. To impartially assess predictive performance and robustness, we performed 2-, 4-, 6-, 8-, and 10-cross-validations. The results of the 10-fold cross-validation are summarized in Table 2 and Fig. 4, while the other results are provided in Table S5. In Table 2, KNN achieved the highest Sn value of 0.953, whereas RF only 0.612. KNN returned the highest overall results in AUC, Sp, Acc, and MCC.

## 3.5. Comparison of inspector with existing methods

We used the same training dataset to perform a 10-fold cross-validation to objectively compare Inspector with three other predictors: SuccinSite2.0 [18], PSuccE [21], and GPSuc [22]. Fig. 5 shows that PSuccE and Inspector displayed similar predictive performance, whose Sn, Sp, Acc, and MCC values are better than those of SuccinSite2.0 and GPSuc. However, Inspector returned the best Sn and Acc of four predictors, therefore, it is more suitable for recognizing Lysine succinylation.

### 3.5.1. Comparison of inspector with existing methods using an independent dataset

To further evaluate the capability of Inspector, we compared its performance with existing methods SuccinSite [17], SuccinSite2.0 [18], Success [20], PSuccE [21], and GPSuc [22] by using an independent

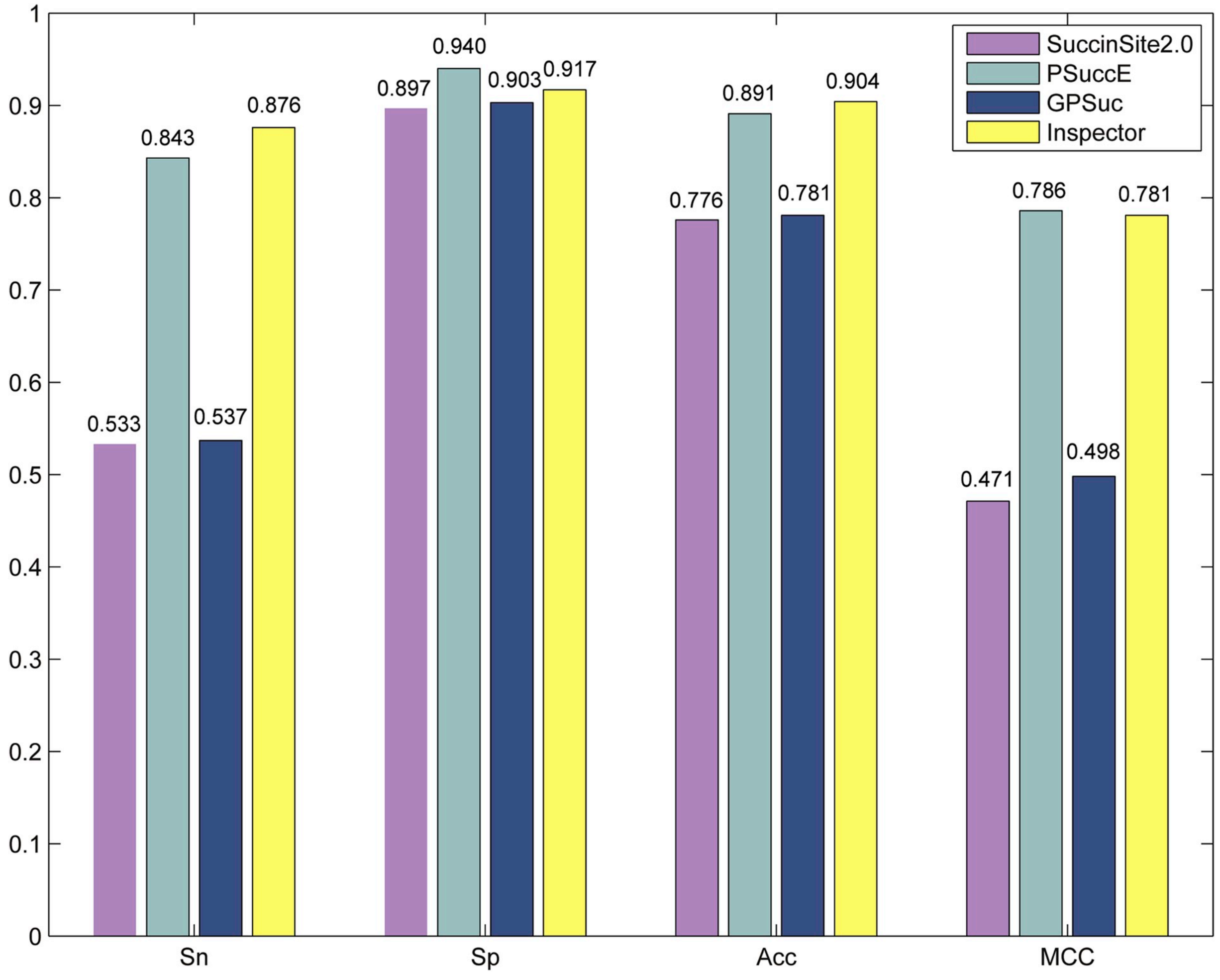

**Fig. 5.** 10-fold cross-validation using the same training dataset.

**Table 3**
Performance of exiting predictors using an independent test dataset.

| Predictor | Sn | Sp | Acc | MCC |
|---|---|---|---|---|
| SuccinSite | 0.371 | 0.882 | 0.842 | 0.199 |
| SuccinSite2.0 | 0.454 | 0.882 | 0.848 | 0.261 |
| Success | 0.142 | 0.868 | 0.811 | 0.070 |
| PsuccE | 0.375 | **0.886** | 0.845 | 0.204 |
| GPSuc | 0.499 | 0.883 | **0.853** | **0.296** |
| Inspector | **0.693** | 0.717 | 0.715 | 0.238 |

dataset. The resulting Sn, Sp, Acc, and MCC values for each method are shown in Table 3.

Inspector achieved the best performance in terms of Sn value (0.693), however, with a poor Sp value of 0.717 compared with other predictors. In contrary, PsuccE returned the best Sp value of 0.886, however, with the worst Sn value of 0.375 compared with other predictors. The GPSuc demonstrated the same trend as PsuccE, with a Sn value of 0.499 and a Sp value of 0.883. Generally speaking, Sn represents the percentage of all positive cases that are predicted as positive examples, thereby measures the ability of a classifier to identify positive examples. By contrast, Sp measures the capacity of a classifier to identify negative examples. Therefore, a low Sn value is associated with “missing diagnosis”. However, Inspector achieved Sp value of 0.717, which has a balanced Sn value. In the biological field, the goal is to find as many lysine succinylation sites as possible, which means the predictor with higher Sn is more suitable for experimental verification.

## 4. Conclusions

In this study, we proposed a novel RF-based method, Inspector, for more accurate prediction of lysine succinylation sites. Inspector incorporated a variety of sequence-based features and used an efficient two-step feature-selection method that successfully combines F-score determination and incremental feature-selection algorithm. Additionally, we applied two dataset-balancing algorithms (ENN undersampling and ADASYN oversampling) to reduce variance in the training sample. Extensive benchmarking tests based on a 10-fold cross-validation and an independent test dataset demonstrated greater effectiveness and especially sensitivity of the predictive performance of Inspector compared to other classifiers. But, the predictive performance of independent dataset still needs to be improved, possibly due to that the test dataset and training dataset do not have the same probability distribution. Therefore, more efficient feature extraction techniques should be studied in the future.

## CRediT authorship contribution statement

**Yan Zhu:** Data curation, Software, Writing - original draft. **Cangzhi Jia:** Conceptualization, Writing - review & editing. **Fuyi Li:** Methodology, Writing - original draft. **Jiangning Song:** Writing - review & editing, Supervision.

## Acknowledgement

This work was supported by Fundamental Research Funds for the

Central Universities (3132019175, 3132019323, 3132018280), the National Natural Science Foundation of Liaoning Province (20180550307, 20170540092) and the Natural Science Foundation of China (61803065). J.S. and F.L.' works were supported by grants from the National Health and Medical Research Council of Australia (NHMRC) (APP490989, APP1127948 and APP1144652), the Australian Research Council (ARC) (LP110200333 and DP120104460), the National Institute of Allergy and Infectious Diseases of the National Institutes of Health (R01 AI111965), and a Major Inter-Disciplinary Research (IDR) project awarded by Monash University.

## Appendix A. Supplementary data

Supplementary data to this article can be found online at https://doi.org/10.1016/j.ab.2020.113592.

## References

[1] A.R. Farley, A.J. Link, Chapter 40 identification and quantification of protein posttranslational modifications, Methods Enzymol. 463 (2009) 725–763.
[2] Y. Zhou, N. Zhang, B.Q. Li, T. Huang, Y.D. Cai, X.Y. Kong, A method to distinguish between lysine acetylation and lysine ubiquitination with feature selection and analysis, J. Biomol. Struct. Dyn. 33 (2015) 2479–2490.
[3] C. Wu, X. Liu, Y. Zheng, W. He, G. Yang, P. Wu, C. Cai, Fluorescence activation imaging of localization, distribution, and level of miRNA in various organelles inside cells, Talanta 186 (2018) 406–412.
[4] Z. Liao, Z. Quan, D. Li, L. Li, X. Wang, Cancer diagnosis from isomiR expression with machine learning method, Curr. Bioinform. 11 (2018) 1-1.
[5] K.Y. Qu, F. Guo, X.R. Liu, Y. Lin, Q. Zou, Application of machine learning in microbiology, Front. Microbiol. 10 (2019).
[6] M.A. Castro, F.A. Beltrán, B. Sebastián, I.I. Concha, A metabolic switch in brain: glucose and lactate metabolism modulation by ascorbic acid, J. Neurochem. 110 (2010) 423–440.
[7] X.X. Zeng, W. Wang, G.S. Deng, J.X. Bing, Q. Zou, Prediction of potential disease-associated MicroRNAs by using neural networks, Mol. Ther. Nucleic Acids 16 (2019) 566–575.
[8] I.G. Udina, A.V. Baranova, A.A. Kompaniytsev, G.E. Sulimova, The evolutionarily conserved CKAP2 gene is located in human region 13q14.3 often rearranged in various tumors, Russ. J. Genet. 37 (2001) 105–107.
[9] W. Bao, L. Zhu, D.S. Huang, ILSES: identification lysine succinylation-sites with ensemble classification, IEEE International Conference on Bioinformatics & Biomedicine, 2017.
[10] X. Xu, T. Liu, J. Yang, L. Chen, B. Liu, C. Wei, L. Wang, Q. Jin, The first succinylome profile of Trichophyton rubrum reveals lysine succinylation on proteins involved in various key cellular processes, BMC Genomics 18 (2017) 577.
[11] F. Li, C. Fan, T.T. Marquez-Lago, A. Leier, J. Revote, C. Jia, Y. Zhu, A.I. Smith, G.I. Webb, Q. Liu, L. Wei, J. Li, J. Song, PRISMOID: a Comprehensive 3D Structure Database for Post-translational Modifications and Mutations with Functional Impact, Brief Bioinform, 2019, https://doi.org/10.1093/bib/bbz050.
[12] Q. Zou, P. Xing, L. Wei, B. Liu, Gene2vec: Gene Subsequence Embedding for Prediction of Mammalian N 6-methyladenosine Sites from mRNA, RNA, New York, N.Y.), 2019.
[13] Z. Chen, X. Liu, F. Li, C. Li, T. Marquez-Lago, A. Leier, T. Akutsu, G.I. Webb, D. Xu, A.I. Smith, L. Li, K.C. Chou, J. Song, Large-scale Comparative Assessment of Computational Predictors for Lysine Post-translational Modification Sites, Brief Bioinform, 2018, https://doi.org/10.1093/bib/bby089.
[14] Y. Li, M. Niu, Q. Zou, ELM-MHC: an improved MHC identification method with extreme learning machine algorithm, J. Proteome Res. 18 (2019) 1392–1401.
[15] Y. Xu, Y.X. Ding, J. Ding, Y.H. Lei, L.Y. Wu, N.Y. Deng, iSuc-PseAAC: predicting lysine succinylation in proteins by incorporating peptide position-specific propensity, Sci. Rep. UK 5 (2015).
[16] J.H. Jia, Z. Liu, X. Xiao, B.X. Liu, K.C. Chou, iSuc-PseOpt: identifying lysine succinylation sites in proteins by incorporating sequence-coupling effects into pseudo components and optimizing imbalanced training dataset, Anal. Biochem. 497 (2016) 48–56.
[17] M.M. Hasan, S. Yang, Y. Zhou, M.N. Mollah, SuccinSite: a computational tool for the prediction of protein succinylation sites by exploiting the amino acid patterns and properties, Mol. Biosyst. 12 (2016) 786–795.
[18] M.M. Hasan, M.S. Khatun, M.N.H. Mollah, C. Yong, D.J. Guo, A systematic identification of species-specific protein succinylation sites using joint element features information, Int. J. Nanomed. 12 (2017) 1–13.
[19] D. Abdollah, Y. Lopez, S.P. Lal, G. Taherzadeh, A. Sattar, T. Tsunoda, A. Sharma, Improving succinylation prediction accuracy by incorporating the secondary structure via helix, strand and coil, and evolutionary information from profile bigrams, PLoS One 13 (2018).
[20] Y. Lopez, A. Sharma, A. Dehzangi, S.P. Lal, G. Taherzadeh, A. Sattar, T. Tsunoda, Success: evolutionary and structural properties of amino acids prove effective for succinylation site prediction, BMC Genomics 19 (2018).
[21] Q. Ning, X.S. Zhao, L.L. Bao, Z.Q. Ma, X.W. Zhao, Detecting Succinylation sites from protein sequences using ensemble support vector machine, BMC Bioinf. 19 (2018).
[22] M.M. Hasan, H. Kurata, GPSuc: global prediction of generic and species-specific succinylation sites by aggregating multiple sequence features, PLoS One 13 (2018).
[23] H. Ying, N. Beifang, G. Ying, F. Limin, L. Weizhong, CD-HIT Suite: a web server for clustering and comparing biological sequences, Bioinformatics 26 (2010) 680–682.
[24] C. Jia, H. Gong, Y. Zhu, Y. Shi, The computational prediction methods for linear B-cell epitopes, Curr. Bioinform. 14 (2019) 226–233.
[25] J.L. Shao, D. Xu, S.N. Tsai, Y.F. Wang, S.M. Ngai, Computational identification of protein methylation sites through Bi-profile Bayes feature extraction, PLoS One 4 (2009).
[26] M. Zhang, F. Li, T.T. Marquez-Lago, A. Leier, C. Fan, C.K. Kwoh, K.C. Chou, J. Song, C. Jia, MULTiPly: a novel multi-layer predictor for discovering general and specific types of promoters, Bioinformatics 35 (2019) 2957–2965.
[27] J.N. Song, H. Tan, H.B. Shen, K. Mahmood, S.E. Boyd, G.I. Webb, T. Akutsu, J.C. Whisstock, Cascleave: towards more accurate prediction of caspase substrate cleavage sites, Bioinformatics 26 (2010) 752–760.
[28] C. Jia, W. H, EnhancerPred: a predictor for discovering enhancers based on the combination and selection of multiple features, Sci. Rep. 6 (2016) 38741.
[29] B.Q. Li, L.L. Hu, S. Niu, Y.D. Cai, K.C. Chou, Predict and analyze S-nitrosylation modification sites with the mRMR and IFS approaches, J. Proteom. 75 (2012) 1654–1665.
[30] Y. Xu, X. Wen, X.J. Shao, N.Y. Deng, K.C. Chou, iHyd-PseAAC: predicting hydroxyproline and hydroxylysine in proteins by incorporating dipeptide position-specific propensity into pseudo amino acid composition, Int. J. Mol. Sci. 15 (2014) 7594–7610.
[31] H.B. Shen, K.C. Chou, PseAAC: a flexible web server for generating various kinds of protein pseudo amino acid composition, Anal. Biochem. 373 (2008) 386–388.
[32] T. Li, R. Song, Q. Yin, M. Gao, Y. Chen, Identification of S-nitrosylation sites based on multiple features combination, Sci. Rep. 9 (2019) 3098.
[33] Q. Wuyun, W. Zheng, Y. Zhang, J. Ruan, G. Hu, Improved species-specific lysine acetylation site prediction based on a large variety of features set, PLoS One 11 (2016) e0155370.
[34] S.P. Shi, J.D. Qiu, X.Y. Sun, S.B. Suo, S.Y. Huang, R.P. Liang, PMeS: prediction of methylation sites based on enhanced feature encoding scheme, PLoS One 7 (2012) e38772.
[35] Z. Chen, P. Zhao, F.Y. Li, A. Leier, T.T. Marquez-Lago, Y.N. Wang, G.I. Webb, A.I. Smith, R.J. Daly, K.C. Chou, J.N. Song, iFeature: a Python package and web server for features extraction and selection from protein and peptide sequences, Bioinformatics 34 (2018) 2499–2502.
[36] Z. Chen, P. Zhao, F. Li, T.T. Marquez-Lago, A. Leier, J. Revote, Y. Zhu, D.R. Powell, T. Akutsu, G.I. Webb, K.-C. Chou, A.I. Smith, R.J. Daly, J. Li, J. Song, iLearn: an integrated platform and meta-learner for feature engineering, machine-learning analysis and modeling of DNA, RNA and protein sequence data, Briefings Bioinf. (2019), https://doi.org/10.1093/bib/bbz041.
[37] L.Y. Wei, P.W. Xing, J.C. Zeng, J.X. Chen, R. Su, F. Guo, Improved prediction of protein-protein interactions using novel negative samples, features, and an ensemble classifier, Artif. Intell. Med. 83 (2017) 67–74.
[38] D.L. Wilson, Asymptotic properties of nearest neighbor rules using edited data, IEEE Trans. Syst. Man Cybern., SMC-2 (2007) 408–421.
[39] H.B. He, Y. Bai, E.A. Garcia, S.T. Li, ADASYN: adaptive synthetic sampling approach for imbalanced learning, IEEE Int. Jt. Conf. Neural Netw. 1–8 (2008) 1322–1328 2008.
[40] N.V. Chawla, K.W. Bowyer, L.O. Hall, W.P. Kegelmeyer, SMOTE: synthetic minority over-sampling technique, J. Artif. Intell. Res. 16 (2002) 321–357.
[41] C. Jia, Y. Zuo, S-SulfPred, A sensitive predictor to capture S-sulfenylation sites based on a resampling one-sided selection undersampling-synthetic minority oversampling technique, J. Theor. Biol. 422 (2017) 84–89.
[42] F. Li, C. Li, M. Wang, G.I. Webb, Y. Zhang, J.C. Whisstock, J. Song, GlycoMine: a machine learning-based approach for predicting N-, C- and O-linked glycosylation in the human proteome, Bioinformatics 31 (2015) 1411–1419.
[43] F. Li, C. Li, J. Revote, Y. Zhang, G.I. Webb, J. Li, J. Song, T. Lithgow, GlycoMine (struct): a new bioinformatics tool for highly accurate mapping of the human N-linked and O-linked glycoproteomes by incorporating structural features, Sci. Rep. 6 (2016) 34595.
[44] V.M. Bui, S.L. Weng, C.T. Lu, T.H. Chang, T.Y. Weng, T.Y. Lee, SOHSite: incorporating evolutionary information and physicochemical properties to identify protein S -sulfenylation sites, BMC Genomics 17 (2016) 9.
[45] L. Breiman, Random forests, Mach. Learn. 45 (2001) 5–32.
[46] C. Jia, Y. Zuo, Q. Zou, J. Hancock, O-GlcNAcPRED-II: an integrated classification algorithm for identifying O-GlcNAcylation sites based on fuzzy undersampling and a K-means PCA oversampling technique, Bioinformatics (2018) 34.
[47] J.H. Jia, Z. Liu, X. Xiao, B.X. Liu, K.C. Chou, pSuc-Lys: predict lysine succinylation sites in proteins with PseAAC and ensemble random forest approach, J. Theor. Biol. 394 (2016) 223–230.
[48] Y. Li, M.J. Wang, H.L. Wang, H. Tan, Z.D. Zhang, G.I. Webb, J.N. Song, Accurate in silico identification of species-specific acetylation sites by integrating protein sequence-derived and functional features, Sci. Rep. UK 4 (2014).
[49] X.Q. Ru, L.H. Li, Q. Zou, Incorporating distance-based top-n-gram and random forest to identify electron transport proteins, J. Proteome Res. 18 (2019) 2931–2939.
[50] L.Y. Wei, S.X. Wan, J.S. Guo, K.K.L. Wong, A novel hierarchical selective ensemble classifier with bioinformatics application, Artif. Intell. Med. 83 (2017) 82–90.
[51] F. Li, C. Li, T.T. Marquez-Lago, A. Leier, T. Akutsu, A.W. Purcell, A. Ian Smith, T. Lithgow, R.J. Daly, J. Song, K.C. Chou, Quokka: a comprehensive tool for rapid and accurate prediction of kinase family-specific phosphorylation sites in the human proteome, Bioinformatics 34 (2018) 4223–4231.
[52] Y. Guo, D. Peng, J. Zhou, S. Lin, C. Wang, W. Ning, H. Xu, W. Deng, Y. Xue, iEKPD

2.0: an update with rich annotations for eukaryotic protein kinases, protein phosphatases and proteins containing phosphoprotein-binding domains, Nucleic Acids Res. 47 (2019) D344–D350.

[53] L.D. Wang, R.J. Zhang, Y.S. Mu, Fu-SulfPred, Identification of protein S-sulfenylation sites by fusing forests via chou's general PseAAC, J. Theor. Biol. 461 (2019) 51–58.

[54] V.N. Nguyen, K.Y. Huang, C.H. Huang, T.H. Chang, N.A. Bretana, K.R. Lai, J.T.Y. Weng, T.Y. Lee, Characterization and identification of ubiquitin conjugation sites with E3 ligase recognition specificities, BMC Bioinf. 16 (2015).

[55] F. Li, Y. Wang, C. Li, T.T. Marquez-Lago, A. Leier, N.D. Rawlings, G. Haffari, J. Revote, T. Akutsu, K.C. Chou, A.W. Purcell, R.N. Pike, G.I. Webb, A. Ian Smith, T. Lithgow, R.J. Daly, J.C. Whisstock, J. Song, Twenty Years of Bioinformatics Research for Protease-specific Substrate and Cleavage Site Prediction: a Comprehensive Revisit and Benchmarking of Existing Methods, Brief Bioinform, 2018, https://doi.org/10.1093/bib/bby077.

[56] F. Li, J. Chen, A. Leier, T. Marquez-Lago, Q. Liu, Y. Wang, J. Revote, A.I. Smith, T. Akutsu, G.I. Webb, L. Kurgan, J. Song, DeepCleave: a deep learning predictor for caspase and matrix metalloprotease substrates and cleavage sites, Bioinformatics (2019), https://doi.org/10.1093/bioinformatics/btz721.

[57] F. Li, Y. Zhang, A.W. Purcell, G.I. Webb, K.C. Chou, T. Lithgow, C. Li, J. Song, Positive-unlabelled learning of glycosylation sites in the human proteome, BMC Bioinf. 20 (2019) 112.

[58] J. Song, F. Li, A. Leier, T.T. Marquez-Lago, T. Akutsu, G. Haffari, K.C. Chou, G.I. Webb, R.N. Pike, J. Hancock, PROSPERous: high-throughput prediction of substrate cleavage sites for 90 proteases with improved accuracy, Bioinformatics 34 (2018) 684–687.

[59] C. Jia, M. Zhang, C. Fan, F. Li, J. Song, Formator: predicting lysine formylation sites based on the most distant undersampling and safe-level synthetic minority oversampling, IEEE ACM Trans. Comput. Biol. Bioinform (2019), https://doi.org/10.1109/TCBB.2019.2957758.

[60] S. Mei, F. Li, A. Leier, T.T. Marquez-Lago, K. Giam, N.P. Croft, T. Akutsu, A.I. Smith, J. Li, J. Rossjohn, A.W. Purcell, J. Song, A Comprehensive Review and Performance Evaluation of Bioinformatics Tools for HLA Class I Peptide-Binding Prediction, Brief Bioinform, 2019, https://doi.org/10.1093/bib/bbz051.

[61] X. Wang, C. Li, F. Li, V.S. Sharma, J. Song, G.I. Webb, SIMLIN: a bioinformatics tool for prediction of S-sulphenylation in the human proteome based on multi-stage ensemble-learning models, BMC Bioinf. 20 (2019) 602.