SCIENCE CHINA
Information Sciences

• RESEARCH PAPER •

November 2024, Vol. 67, Iss. 11, 212106:1–212106:16
https://doi.org/10.1007/s11432-024-4171-9

SBSM-Pro: support bio-sequence machine for proteins

Yizheng WANG[1,2], Yixiao ZHAI[1,2], Yijie DING[2*] & Quan ZOU[1,2*]

[1]*Institute of Fundamental and Frontier Sciences, University of Electronic Science and Technology of China, Chengdu* 611731*, China;*
[2]*Yangtze Delta Region Institute (Quzhou), University of Electronic Science and Technology of China, Quzhou* 324003*, China*

Received 1 March 2024/Revised 12 April 2024/Accepted 20 May 2024/Published online 22 October 2024

Abstract Proteins play a pivotal role in biological systems. The use of machine learning algorithms for protein classification can assist and even guide biological experiments, offering crucial insights for biotechnological applications. We introduce the support bio-sequence machine for proteins (SBSM-Pro), a model purpose-built for the classification of biological sequences. This model starts with raw sequences and groups amino acids based on their physicochemical properties. It incorporates sequence alignment to measure the similarities between proteins and uses a novel multiple kernel learning (MKL) approach to integrate various types of information, utilizing support vector machines for classification prediction. The results indicate that our model demonstrates commendable performance across ten datasets in terms of the identification of protein function and post translational modification. This research not only exemplifies state-of-the-art work in protein classification but also paves avenues for new directions in this domain, representing a beneficial endeavor in the development of platforms tailored for the classification of biological sequences. SBSM-Pro is available for access at http://lab.malab.cn/soft/SBSM-Pro/.

Keywords protein classification, machine learning, multiple kernel learning, sequence alignment

1 Introduction

Bio-sequences, including DNA, RNA, and proteins, are the basis of genetic research. With the accumulation of extensive genomic data, there is an urgent need for computer-assisted function annotation. Due to the complex relationship between genetic sequences and diseases, multiple computer predictions are necessary to assist laboratory research. Bio-sequence classification is a valuable tool that helps researchers understand and analyze biological sequences. It serves as a key driving force for advancing research in the field of bioinformatics.

In the field of bio-sequence classification, machine learning methods are broadly pursued using two strategies: feature extraction combined with traditional classification methods and direct sequence classification via deep learning techniques.

For bio-sequences, relevant features are mainly characterized as frequency, physicochemical, structural, and evolutionary features [1]. Several notable tools for sequence feature extraction include PseKNC-General [2], PyFeat [3], iFeature [4], VisFeature [5], POSSUM [6], Rcpi [7], and Protr [8]. Furthermore, every alphabet in the sequence, whether amino acids or nucleotides, can be numerically represented, thereby contributing to the global feature of the sequence [9–11].

After completing feature extraction, protein structure and function recognition can be achieved through machine learning. Among the algorithms that have shown excellent performance and are widely used in recent research, random forests (RF) [12] and support vector machines (SVM) [13, 14] are notable examples. Given these traditional numerical classification features, classifiers can be integrated to facilitate the classification and discrimination of biological sequences. This led to the emergence of platforms

* Corresponding author (email: wuxi_dyj@163.com, zouquan@nclab.net)

that combine feature extraction and classifiers, such as gkmSVM [15], iLearnPlus [16], Biological Seq-Analysis 2.0 [17], and BioSeq-BLM [18]. Notably, gkmSVM was one of the first to use kernel methods for biological sequence predictions, with the most common frequency feature being k-mer, and yielded promising results in certain scenarios, such as predicting enhancer activity in specific cell types [19] and disease-relevant mutations [20]. However, the performance of gkmSVM frequently falls short due to its exclusive reliance on rudimentary k-mer features and its susceptibility to overfitting. Both iLearnPlus and Biological Seq-Analysis 2.0 offer a rich array of feature extraction and analysis methods, making them more commonly employed in biological sequence classification research compared to traditional tools. However, these tools do not account for sequence structural information.

Deep learning-based methods circumvent the need for feature extraction by directly encoding sequences into neural networks. Through training, the architecture and parameters of the network are fine-tuned, enabling it to classify the training samples effectively. The most renowned application of this approach is AlphaFold2's [21] prediction of protein 3D structures, facilitated by the advent of cryo-electron microscopy, which provides a wealth of 3D structural samples for AI training. Platforms such as Kipoi [22], Pysster [23], Selene [24], and DNA-BERT [25] have been developed for deep learning-based classification of biological sequences. Inspired by biological processes, convolutional neural networks (CNNs), whose connectivity patterns between neurons resemble those of the animal visual cortex, are commonly used for various sequence data to learn the inherent regularities [26–33] and specificities [34] within gene sequences. By reincorporating newly discovered sequence motifs into the neural network model and continuously updating the model's predictive scores, accuracy in predicting sequence specificity can be improved, enabling the analysis of potentially pathogenic genomic variations. Recurrent neural networks (RNNs) accumulate sequence information over time, and the bidirectional LSTM proposed based on it also achieved good results [35]. Hybrid predictive models combining both CNNs and RNNs are currently popular and have been applied in various computational biology domains, including DNA methylation [36], chromatin accessibility [37], and noncoding RNAs [38]. With the application of new technologies from the field of natural language processing to biological sequence analysis, protein pre-trained language models have performed well in tasks related to the recognition of protein structure and function [39].

Classical CNNs and RNNs, as well as more recently celebrated transformer architectures like BERT [40, 41] and GPT, have their origins in domains like image analysis (for tasks such as face recognition or autonomous driving) and natural language processing. However, there is no universal algorithm or framework specifically tailored for biological sequence data. The development of custom algorithms specifically designed for these types of data and problems represents one of the most exciting prospects in bioinformatics. Conventional machine learning approaches, despite their advantage in facilitating the development of user-friendly predictive software platforms through numerical feature extraction, have limitations. Relying solely on word frequency, physicochemical, and evolutionary features fall short in capturing the full spectrum of sequence information, thus invariably imposing a ceiling on prediction accuracy. On the other hand, deep learning-based approaches pose unique challenges. They demand a substantial volume of training data to avoid overfitting, and the complexity of deep learning software packages can detract from the user-friendliness of the sequence classification platform.

In response to the challenges encountered by the aforementioned research approaches, building on the strengths of SVM, especially their prowess in effectively managing small sample problems, we present an innovative methodology termed the SBSM-Pro. In this paper, we make several key contributions to the field, which are given as follows. (i) We propose a novel standard process named physicochemical properties-spectral clustering-dictionaries (PSD) that effectively reduces the amino acid alphabet. This process facilitates sequence alignment and accurately represents the distances between proteins, thereby linking the physicochemical properties of proteins with their sequences. (ii) We introduce two methods for calculating sequence similarity kernels, namely, the Levenshtein (LS) distance and the Smith–Waterman (SW) score. These techniques allow for precise comparisons between protein sequences. (iii) We present a new MKL approach that combines global and local kernels, thus effectively integrating multiple similarity kernels. This distinctive method optimizes the processing and understanding of protein data. (iv) We employ an SVM with a precomputed kernel to receive the fused sequence kernels for protein prediction. This machine learning model ensures efficient and precise prediction. These combined contributions present a comprehensive and innovative approach to the analysis and prediction of protein sequences.

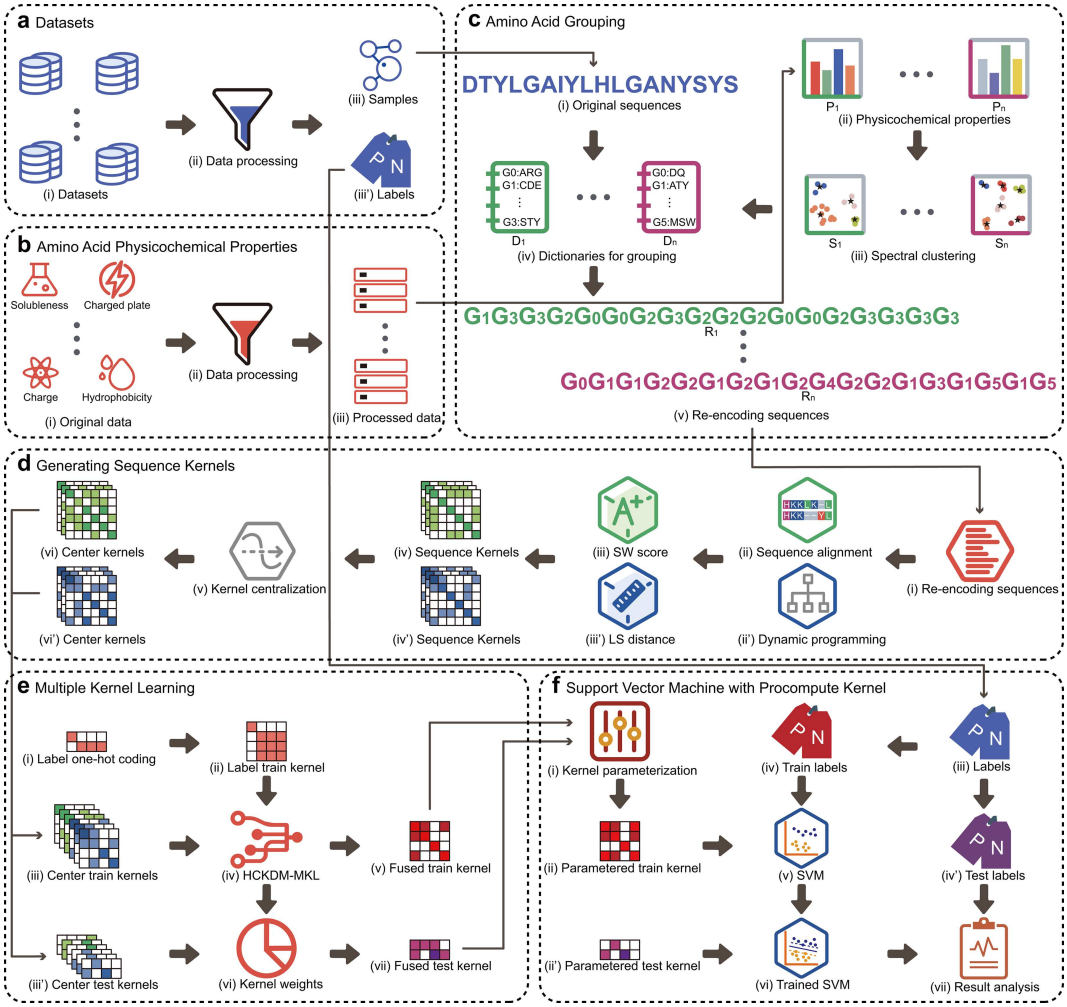

Figure 1 (Color online) Overview of SBSM-Pro. (a) Processing 10 datasets to obtain labels and samples; (b) collecting and processing data on amino acid physicochemical properties; (c) re-encoding sequences using the PSD process; (d) generating sequence kernels using LS distance and SW score; (e) applying HCKDM-MKL to generate fused training and testing kernels; (f) training and testing SVM.

2 Materials and methods

In this section, we will delve into the methodologies associated with SBSM-Pro. Figure 1 provides an overview of SBSM-Pro. The first step involves collecting relevant datasets for protein identification. After collecting these datasets, they underwent processing, resulting in the creation of multiple sets of protein samples with their respective labels. Subsequently, we retrieved physicochemical property data of amino acids from the available literature. These data were also preprocessed. Finally, the amino acid physicochemical properties were subjected to spectral clustering, giving rise to the dictionaries for grouping. The original protein sequences were then transformed into re-encoding sequences by amino acid grouping in accordance with the corresponding dictionary. To gauge the similarity among these reencoding sequences, we employed sequence alignment techniques along with dynamic programming methods. Central kernels were derived by applying suitable kernel processing techniques. We proposed an innovative MKL strategy to fuse these central kernels. The fused central kernel was subsequently fed into SBSM-Pro for classification, ultimately leading to the final classification outcome.

2.1 Datasets

To evaluate SBSM-Pro, we collected 10 different protein classification datasets, including the identification of protein functions and posttranslational modifications (PTMs). The datasets encompass various protein functionalities, such as DNA-binding proteins (DBPs) [42], type III secreted effectors (T3SEs) [43], and

Table 1 Summary of datasets utilized in the research

Dataset	Description	Training set		Testing set	
		Positive	Negative	Positive	Negative
DBP	DNA-binding proteins	525	550	93	93
T3SE	Type III secreted effectors	309	310	42	34
PVP	Phage virion proteins	99	208	30	64
PTSS	Protein tyrosine sulfation sites	200	420	80	80
PSNS	Protein S-nitrosylation sites	731	810	43	121
PLGS	Protein lysine glutarylation sites	400	1703	44	203
PCS1	Protein carbonylation sites 1	300	1949	-	-
PCS2	Protein carbonylation sites 2	126	792	-	-
PCS3	Protein carbonylation sites 3	136	847	-	-
PCS4	Protein carbonylation sites 4	121	732	-	-

phage virion proteins (PVPs) [44], contributing to our understanding of genetic encoding, host–pathogen interactions, and virus–host relationships, respectively. Regarding PTMs, we considered protein tyrosine sulfation sites (PTSS) [45], protein s-nitrosylation sites (PSNS) [46], protein lysine glutarylation sites (PLGS) [47], and protein carbonylation sites (PCS) [48]. These PTMs play significant roles in modifying the behavioral properties of proteins and are implicated in numerous cellular processes, including metabolic regulation, redox reactions, and biological processes linked with various diseases. These datasets allow for a comprehensive evaluation of our SBSM-Pro, providing robust validation of our model through the identification of protein function and PTMs.

We collected a set of commonly used datasets for protein classification, including 3 for protein function identification and 7 for PTM identification. We then analyzed the protein sequence lengths and presented them as a box plot, as detailed in Appendix A. It is worth noting that the protein sequence lengths in each PTM identification dataset are consistent. However, the lengths of sequences for protein function identification vary.

In assessing two machine learning algorithms, it is crucial to employ the same training and testing sets for evaluation. This methodology eliminates variations that may arise from different data partitions, ensuring a reliable and fair comparison between algorithms. In our study, we categorized the collected datasets into two types: Type I and Type II. With regards to Type I datasets, it has been a customary practice in previous research to assess models using pre-partitioned datasets. We adhered to this practice to guarantee fairness in the comparison of the algorithms. Type II datasets, in contrast, do not have a predefined division into training and testing sets. For these datasets, we utilized the method of 10-fold cross-validation for model evaluation, in line with approaches utilized in prior studies.

The fundamental step of cross-validation involves partitioning the entire dataset into K subsets. In each iteration, one subset is designated as the testing set, while the remaining K-1 subsets serve as the training set, yielding model evaluation outcomes. This process is executed K times, ensuring a different testing set for each run. Consequently, K-model evaluation outcomes are obtained, and the final model performance assessment is derived from their average. A summary of datasets is shown in Table 1.

2.2 Amino acid grouping

For a specific amino acid with distinct physicochemical properties, we employ spectral clustering to perform clustering analysis and CHI to evaluate the clustering effect, as detailed in Appendix A. The clustering process results in the formation of k_c clusters, each containing at least one amino acid. We consider each cluster as a distinct group, defined as a set. An illustrative example is provided below:

$$G_1 = \{A, W\}, \tag{1}$$

where A and W respectively represent Alanine and Tryptophan.

Due to the existence of k_c clusters, there are k_c sets, similar to those in (1). These sets satisfy the following equation:

$$\mathrm{AA} = \bigcup_{i=1}^{k} G_i, \tag{2}$$

where AA represents a set of 20 amino acids.

Next, we consider k_c groups as k_c dictionary entries and include them in a set, resulting in the formation of dictionaries for grouping that is defined as follows:

$$D_1 = \{G_1, \cdots, G_K\}. \tag{3}$$

In accordance with a specific dictionary, amino acid residues of the original protein sequence are replaced by the group number in which they are located, resulting in the re-encoding protein sequence.

The physicochemical properties of amino acids are obtained through data processing, the specific details of which can be found in Appendix A, and then the corresponding clustering results are achieved through spectral clustering. According to the clustering results, dictionaries for grouping can be generated. There is a one-to-one correspondence between them. We define this standard process as PSD. The PSD process reduces the alphabet of the original protein sequence. In biology, amino acid residues within proteins may undergo substitutions. However, some of these substitutions may have minimal impact on the protein's function or even no effect at all. By grouping amino acids using PSD, we can reduce such noise, facilitating subsequent sequence alignment. Additionally, this approach can link the original sequence of the protein to its structure, enabling a more accurate representation of the distances between proteins.

2.3 Generating sequence kernels

In the previous section, we re-encoded the protein sequences. In this section, we generate the SW score [49] and LS distance through sequence alignment and dynamic programming, respectively, as detailed in Appendix A. Subsequently, a series of transformations, such as normalization and centralization, are applied to produce the sequence similarity kernel.

The following equation presents the normalized SW score for the protein sequences S_x and S_y:

$$\mathrm{SW}^*(S_x, S_y) = \frac{\mathrm{SW}(S_x, S_y)}{\max(l_x, l_y)}, \tag{4}$$

where $\mathrm{SW}(S_x, S_y)$ and $\mathrm{SW}^*(S_x, S_y)$ represent the original and normalized SW scores, respectively. Meanwhile, l_x and l_y correspond to the lengths of protein sequences S_x and S_y, respectively.

By computing the SW scores between all pairs of sequences in the sample set, we store the results in the symmetric matrix $\mathbf{K_{SW}}$, thus obtaining a protein similarity kernel based on the SW algorithm.

The LS distance, also known as the edit distance, is a widely used metric for measuring the dissimilarity between two strings. It quantifies the minimum number of single-character operations required to transform one string into another. These operations include insertion, deletion, and substitution.

LS distance and the SW algorithm both use dynamic programming to calculate the similarity between two strings. The former measures the minimum number of editing operations needed to transform one string into another, while the latter calculates the highest score of all possible local sequence alignments, rather than the overall edit distance. Compared to the SW algorithm, which can be optimized by adjusting scoring settings to suit specific types of sequence alignment tasks, the LS distance generally does not involve complex scoring mechanisms for character matching. When calculating the LS distance between two protein sequences S_x and S_y, each with lengths l_x and l_y respectively, a matrix $\mathbf{M} \in \mathbf{R}^{l_x \times l_y}$ is initially generated. The matrix serves as a dynamic programming table that stores the intermediate distances between substrings. Initialization of the matrix involves setting the first row and column to the respective indices, representing the cost of transforming an empty string into the corresponding prefix or vice versa. The subsequent entries in the matrix are filled iteratively by comparing characters at each position and determining the minimum cost of transforming the prefixes. The final value in the bottom right corner of the matrix represents the LS distance between the two input strings. The LS distances for all protein sequence pairs were calculated, resulting in the LS distance matrix, $\mathbf{D_{LS}}$.

Given that the LS distance measures the dissimilarity between two protein sequences, a protein similarity kernel based on the LS distance $\mathbf{K_{LS}}$ can be obtained through normalization methods. The equation is as follows:

$$\mathbf{K_{LS}} = \mathbf{1} - \frac{\mathbf{D_{LS}} - \min(\mathbf{D_{LS}})}{\max(\mathbf{D_{LS}}) - \min(\mathbf{D_{LS}})}, \tag{5}$$

where $\max(\mathbf{D_{LS}})$ and $\min(\mathbf{D_{LS}})$ represent the maximum and minimum elements in the matrix $\mathbf{D_{LS}}$, respectively.

2.4 Multiple kernel learning

MKL methods have gained significant attention in the field of machine learning due to their ability to effectively model complex relationships in data [50]. These methods extend the traditional single kernel approach by combining multiple kernels to capture information on different aspects of the data. By employing MKL methods, we are able to leverage complementary information from different feature representations, leading to improved predictive performance. In the MKL model, the fused kernel is derived by determining the kernel weights.

For a dataset comprising N samples, we construct a set of p kernel matrices by building the LS kernel and the SW kernel, as defined below:

$$\mathbf{K} = \{\mathbf{K}_1, \mathbf{K}_2, \cdots, \mathbf{K}_p\}, \mathbf{K}_p \in \mathbf{R}^{N\times N}. \tag{6}$$

The objective of MKL is to determine the kernel weights, denoted as β. Its definition is as follows:

$$\beta = [\beta_1, \beta_2, \cdots, \beta_p]. \tag{7}$$

For a set of N training samples, there are corresponding sample labels represented by $\mathbf{L} \in \mathbf{R}^{N\times 1}$. These labels are transformed into a one-hot encoding, denoted as $\mathbf{Y}_{\text{train}}$. We define the target kernel as $\mathbf{U} \in \mathbf{R}^{N\times N}$, with its equation given as follows:

$$\mathbf{U} = \mathbf{Y}_{\text{train}}\mathbf{Y}_{\text{train}}^T. \tag{8}$$

The Hilbert-Schmidt Independence Criterion (HSIC) provides an efficient, non-parametric method for independence testing [51]. And it has been widely applied in areas such as feature selection, causality testing, and variable independence verification [52], with its introduction found in Appendix B.

Then, we define $\mathbf{I}$ as the identity matrix, and it satisfies $\mathbf{I} \in \mathbf{R}^{N\times N}$. By defining $\mathbf{e} = [1, 1, \cdots, 1]^T \in \mathbf{R}^{1\times N}$, we can obtain

$$\mathbf{H} \equiv \mathbf{I} - \frac{\mathbf{e}\mathbf{e}^T}{N}. \tag{9}$$

We measure the dependence between two matrices using HSIC, with the equation shown below:

$$\text{HSIC}(\mathbf{K}, \mathbf{U}) = \frac{1}{N^2} tr(\mathbf{KHUH}), \tag{10}$$

where $\mathbf{K}, \mathbf{U} \in \mathbf{R}^{N\times N}$ are kernel matrices, $k(\mathbf{x}_i, \mathbf{x}_j)$ and $l(y_i, y_j)$. It's important to note that the value of $\text{HSIC}(\mathbf{K}, \mathbf{U})$ is associated with the dependence between $\mathbf{K}$ and $\mathbf{U}$, and a higher value indicates a stronger dependence between the two. In addition, it should be in the range of 0 to 1. When it is equal to 0, we think that $\mathbf{K}$ and $\mathbf{U}$ are independent or irrelevant.

First, we centralize the acquired kernel matrix to normalize the data, ensuring that the similarity or distance of each data point consistently influences the results. Moreover, by subtracting the mean values of rows and columns, the centered kernel emphasizes the similarity information. All these steps enhance the effectiveness of MKL algorithms. The equation is as follows:

$$\left(\widehat{\mathbf{K}}_p\right)_{i,j} = (\mathbf{K}_p)_{i,j} - \frac{1}{N}\sum_{i=1}^{N}(\mathbf{K}_p)_{i,j} - \frac{1}{N}\sum_{j=1}^{N}(\mathbf{K}_p)_{i,j} + \frac{1}{N^2}\sum_{i,j=1}^{N}(\mathbf{K}_p)_{i,j}, \tag{11}$$

where $\mathbf{K_p}$ represents the original kernel of the p^{th} similarity kernel and $\widehat{\mathbf{K}}_p$ denotes the centered kernel.

In the method of global kernel alignment, every pair of samples are forcibly aligned to the same level of similarity, regardless of the distance between them. This approach overlooks the diversity within classes and may lead to insufficient capture of subtle variations in sample features in practical applications, potentially missing important classification information and affecting the overall performance. The local kernel alignment method, by considering only the nearest neighbors of each sample to compute the alignment of the kernel, focuses more on the local structure of the data, which helps in capturing the subtle differences in the data [53, 54]. Hybrid central kernel dependence maximization multiple kernel learning (HCKDM-MKL) leverages both global and local kernels as they are complementary to each other. The schematic of this method is shown in Figure 2. Before using HSIC to measure all sample local kernels

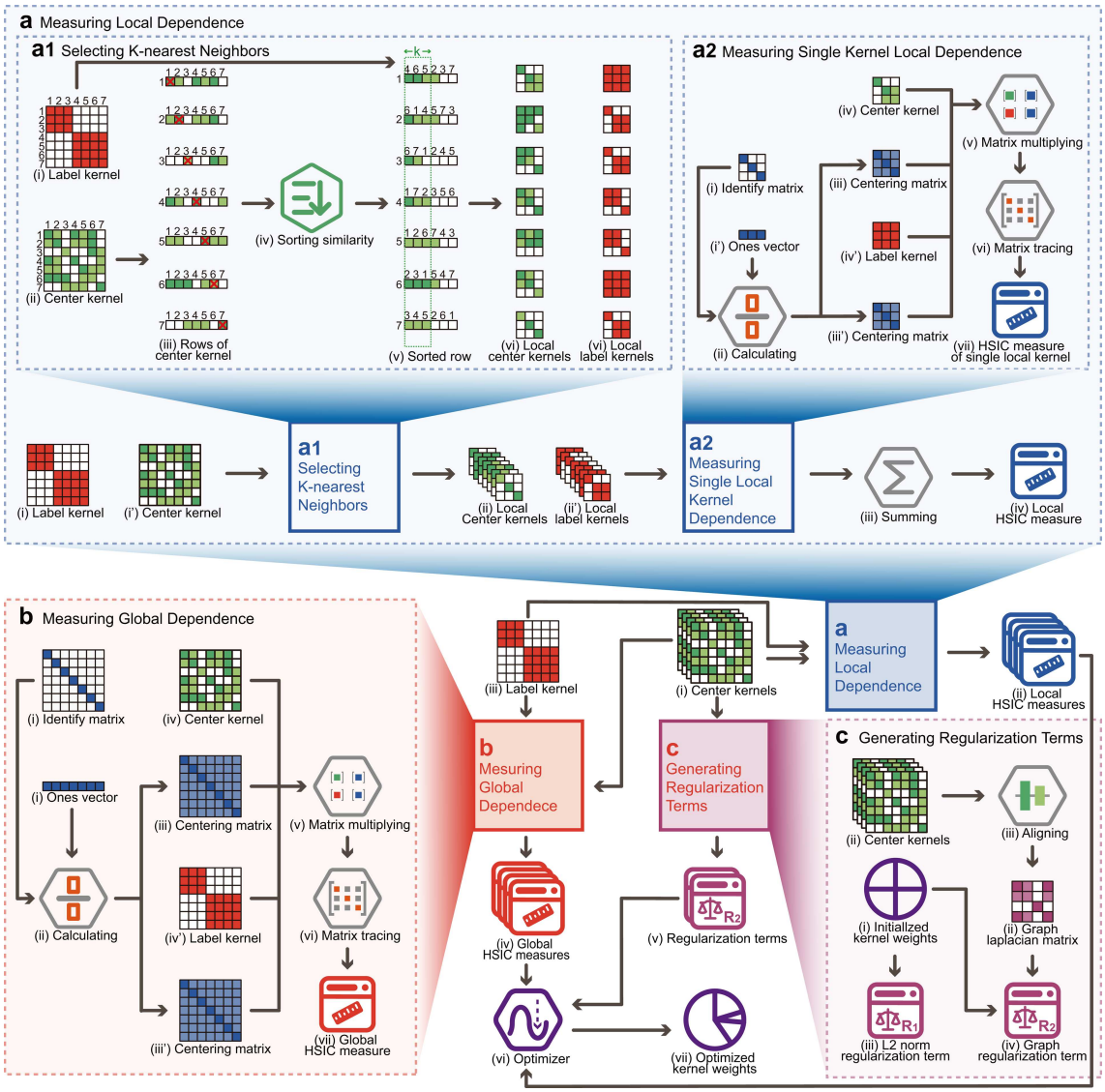

Figure 2 (Color online) Overview of HCKDM-MKL. (a) Measuring local dependence; (b) measuring global dependence; (c) generating regularization terms.

and label kernel, we first define $\mathbf{I}_{local} \in \mathbf{R}^{k\times k}$ as the identity matrix and $\mathbf{e}_{local} = [1, 1, \cdots, 1]^T \in \mathbf{R}^{1\times k}$. We can then calculate the quadratic matrix by the following equation:

$$\mathbf{H}_{local} \equiv \mathbf{I} - \frac{\mathbf{e}_{local}\mathbf{e}_{local}^T}{N} \in \mathbf{R}^{k\times k}. \tag{12}$$

First, calculate the average matrix of all sample matrices. For a given sample i, sort all samples by their similarity to sample i in descending order and select the top k samples with the highest similarity as the nearest neighbors of sample i. Then, extract the corresponding rows and columns from the Center kernel and label kernel matrices to obtain $\hat{\mathbf{K}}^{*(i)}$ and $\mathbf{U}^{(i)}$ respectively. Then, the dependence measures of local and label kernels for all samples M_{local} can be calculated by HSIC, whose equation is shown below:

$$M_{local} = \frac{1}{N}\sum_{i=1}^{N} \mathrm{HSIC}\left(\hat{\mathbf{K}}^{*(i)}, \mathbf{U}^{(i)}\right) = \frac{1}{N}\sum_{i=1}^{N} \frac{1}{k^2} tr\left(\hat{\mathbf{K}}^{*(i)}\mathbf{H}_{local}\mathbf{U}^{(i)}\mathbf{H}_{local}\right). \tag{13}$$

In contrast to local kernels, global kernels are employed to globally represent the characteristics of all kernel functions. To ensure compatibility in matrix dimensions during multiplication, it is necessary to initially define $\mathbf{I}_{global} \in \mathbf{R}^{N\times N} \in \mathbf{R}^{N\times N}$ as the identity and $\mathbf{e}_{global} = [1, 1, \cdots, 1]^T \in \mathbf{R}^{1\times N}$.

Subsequently, we compute the centering matrix according to the equation presented below:

$$\mathbf{H}_{global} \equiv \mathbf{I} - \frac{\mathbf{e}_{global}\mathbf{e}_{global}^{T}}{N} \in \mathbf{R}^{N\times N}, \tag{14}$$

$$M_{global} = \mathrm{HSIC}\left(\hat{\mathbf{K}}^{*}, \mathbf{U}\right) = \frac{1}{N^{2}} tr\left(\hat{\mathbf{K}}^{*}\mathbf{H}_{global}\mathbf{U}\mathbf{H}_{global}\right). \tag{15}$$

R_1 is the L_2 norm regularization, and the equation is as follows:

$$R_1 = \nu_1\|\beta\|^{2}. \tag{16}$$

We define the Frobenius inner as

$$\langle \mathbf{K}, \mathbf{U}\rangle_{F} = tr\left(\mathbf{K}^{T}, \mathbf{U}\right). \tag{17}$$

Then, we defined $\mathbf{W}$ as the cosine similarity matrix between two kernels satisfying $\mathbf{W} \in \mathbf{R}^{P\times P}$, and the equation is as follows:

$$\mathrm{Aligned}\left(\mathbf{K}, \mathbf{U}\right) = \frac{\langle \mathbf{K}, \mathbf{U}\rangle_{F}}{\|\mathbf{K}\|_{F}\|\mathbf{U}\|_{F}}, \tag{18}$$

where $\|\mathbf{K}\|_{F} = \sqrt{\langle \mathbf{K}, \mathbf{K}\rangle_{F}}$ is the Frobenius norm. $\mathbf{D}_{\mathbf{K}}$ is defined as a diagonal matrix that satisfies $\mathbf{W} \in \mathbf{R}^{P\times P}$, and its elements can be calculated as

$$[D_K]_{i,j} = \sum_{j}^{p} [W]_{i,j}. \tag{19}$$

Then, the graph Laplacian matrix $\mathbf{L}_k$ is defined as

$$\mathbf{L}_k = \mathbf{D}_k - \mathbf{W}. \tag{20}$$

We can write the Laplacian regular term as

$$\begin{aligned}\sum_{i,j}^{p}(\beta_i - \beta_j)^{2}W_{ij} &= \sum_{i,j}^{p}\left(\beta_i^{2} + \beta_j^{2} - 2\beta_i\beta_j\right)W_{ij}\\ &= \sum_{i}^{p}\beta_i^{2}D_{ii} + \sum_{j}^{p}\beta_j^{2}D_{jj} - 2\sum_{i,j}^{p}\beta_i\beta_j W_{ij}\\ &= 2\beta^{T}\mathbf{L}_K\beta.\end{aligned} \tag{21}$$

We define R_2 as the graph regularization term, which assists in smoothing the weights. The equation is as follows:

$$R_2 = \beta^{T}\mathbf{L}_K\beta. \tag{22}$$

We combine the regularization terms R_1 and R_2 to obtain the final regularization term R as follows:

$$R = R_1 + R_2 = \nu_1\|\beta\|^{2} + \nu_2\beta^{T}\mathbf{L}_K\beta. \tag{23}$$

Hence, we define a new kernel dependence measuring approach that concurrently considers global and local kernel dependence measurements and uses a parameter $\lambda\,(0 \leqslant \lambda \leqslant 1)$ to establish a trade-off between these two types of kernel alignment. The hybrid dependence measuring between two kernel matrices is as follows:

$$\max \lambda M_{local} + (1-\lambda)M_{global} - R. \tag{24}$$

Then, our method is transformed into an optimization problem, and the optimized fusion kernel can be obtained by maximizing the hybrid dependence, whose equation is shown as follows:

$$\begin{aligned}\max_{\beta,\hat{\mathbf{K}}^{*}} \lambda\tfrac{1}{N}\sum_{i=1}^{N}\tfrac{1}{k^{2}}tr\left(\hat{\mathbf{K}}^{*(i)}\mathbf{H}_{local}\mathbf{U}^{(i)}\mathbf{H}_{local}\right) + (1-\lambda)\tfrac{1}{N^{2}}tr\left(\hat{\mathbf{K}}^{*}\mathbf{H}_{global}\mathbf{U}\mathbf{H}_{global}\right)\\ -\nu_1\|\beta\|^{2} - \nu_2\beta^{T}\mathbf{L}_K\beta,\\ \text{s.t. } \hat{\mathbf{K}}^{*} = \sum_{i=1}^{p}\beta_i\hat{\mathbf{K}}_i,\\ \sum_{i=1}^{p}\beta_i = 1,\\ \beta_i \geqslant 0.\end{aligned} \tag{25}$$

2.5 Support vector machine with precomputed kernel

The similarity kernel obtained from MKL first needs to be parameterized to be compatible with the SVM. Kernel parameterization offers the advantage of enhancing the performance of classifiers or regressors without increasing the computational complexity by mapping the data to a higher-dimensional feature space. Additionally, it can better handle nonlinear relationships among high-dimensional data and samples, thereby expanding the application scope of traditional linear methods. The kernel function of the i-th sequence is defined as.

$$\mathbf{K}\left(\mathbf{x}, \mathbf{x}_i\right) = \exp\left(\alpha \frac{s\left(\mathbf{x}, \mathbf{x}_i\right) - b_i}{b_i}\right), \tag{26}$$

where $s\left(\mathbf{x}, \mathbf{x}_i\right)$ is a pairwise similarity measurement between $\mathbf{x}$ and $\mathbf{x}_i$, b_i is the maximum similarity measurement associated with the i-th support sequence and α is a constant.

The given equation represents the dual form of the SVM optimization problem.

$$\begin{aligned} \max_{\alpha} \sum_{i=1}^{m} \alpha_i - \tfrac{1}{2} \sum_{i=1}^{m} \sum_{j=1}^{m} \alpha_i \alpha_j y_i y_j \hat{K}^*\left(\mathbf{x}_i, \mathbf{x}_j\right), \\ \text{s.t.} \sum_{i=1}^{m} \alpha_i y_i = 0, \\ 0 \leqslant \alpha_i \leqslant C, \ i = 1, 2, \ldots, m. \end{aligned} \tag{27}$$

This function is maximized with respect to α, which is a vector of Lagrange multipliers. The first term of the objective function is the sum of all α_i from 1 to m. The second term is the dot product of the feature vectors $\mathbf{x_i}$ and $\mathbf{x_j}$ scaled by the corresponding α_i, α_j, and the labels y_i, y_j, summed over all pairs of data points. Through HCKDM-MKL, we obtain the fused kernel matrix $\hat{\mathbf{K}}^*$, which corresponds to the kernel function $\hat{K}^*\left(\mathbf{x}_i, \mathbf{x}_j\right)$.

The first constraint ensures that the sum of α_i times the corresponding y_i (the label of each data point) over all data points equals zero. The second constraint bounds each α_i to be nonnegative and no larger than a constant C for all data points. The constant C is a parameter for the SVM that controls the trade-off between maximizing the margin and minimizing the classification error.

3 Results and discussion

3.1 Performance metrics

We employed several widely recognized and indispensable performance metrics for classification models, including accuracy (ACC), true positive rate (TPR) and true negative rate (TNR). TPR, also known as Recall, represents the model's ability to correctly identify positive instances, while TNR indicates the model's capability to accurately identify negative instances. These metrics were chosen to provide a comprehensive evaluation of our model's performance:

$$\mathrm{TPR} = \frac{\mathrm{TP}}{\mathrm{FN} + \mathrm{TP}}, \tag{28}$$

$$\mathrm{TNR} = \frac{\mathrm{TN}}{\mathrm{FP} + \mathrm{TN}}, \tag{29}$$

$$\mathrm{ACC} = \frac{\mathrm{TP} + \mathrm{TN}}{\mathrm{TP} + \mathrm{TN} + \mathrm{FP} + \mathrm{FN}}, \tag{30}$$

where TP, TN, FN, and FP denote the number of true positives, true negatives, false negatives, and false positives, respectively.

3.2 Comparative analysis

To achieve a significant breakthrough with SBSM-Pro, we compared it with the leading contemporary models to evaluate its effectiveness. To demonstrate the robustness of SBSM-Pro, we selected ten commonly used protein classification datasets. The results are shown in Table 2 and Figure 3.

Table 2 Comparison of the proposed method and existing methods

Dataset	SBSM-Pro			Existing methods		
	TPR	TNR	ACC	TPR	TNR	ACC
DBP	0.9140	**0.8710**	**0.8925**	**0.968**	0.538	0.753
T3SE	0.9048	**0.7353**	0.8289	**0.93**	0.71	**0.83**
PVP	**0.7000**	**0.8906**	**0.8298**	0.6670	0.8590	0.7980
PTSS	**0.9375**	**0.8625**	**0.9000**	0.8875	0.8250	0.8563
PSNS	**0.6047**	**0.8017**	**0.7500**	0.6047	0.7769	0.7317
PLGS	**0.7313**	**0.8778**	**0.8381**	0.7273	0.7192	0.7207
PCS1	**0.6983**	0.9038	**0.8737**	0.4518	**0.9925**	0.8443
PCS2	**0.6902**	0.9150	**0.8791**	0.4820	**0.9854**	0.8679
PCS3	**0.7536**	0.8910	**0.8687**	0.4667	**0.9957**	0.8423
PCS4	**0.7204**	0.8976	**0.8699**	0.5068	**0.9858**	0.8617

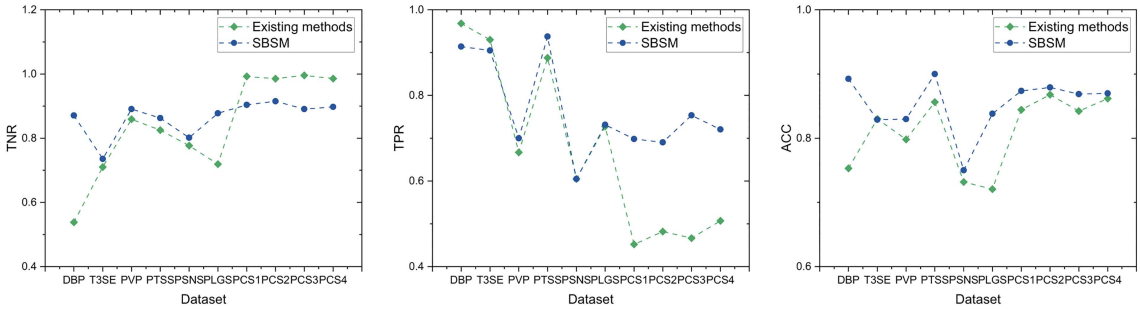

Figure 3 (Color online) Comparison of multiple performance metrics between the proposed method and existing methods.

The previously established method for the DBP dataset, introduced by Lu et al. [42], is a model founded on SVM. This model extracts evolutionary features and concatenates them as input for the model. However, due to its exclusive emphasis on evolutionary features, this model overlooks certain information. The results show that SBSM-Pro surpassed existing methods in TNR, while it was weaker than existing methods in TPR. However, overall, SBSM-Pro achieved an ACC that was 0.1853 higher than the existing methods.

Hui et al. [43] developed the T3SEpp model, which exhibits the best performance with the T3SE dataset. This model integrates both traditional machine learning models, such as SVM and RF, and deep learning models, such as fully connected neural networks and convolutional neural networks. The performance of SBSM-Pro is on par with that of T3SEpp. It is important to note that due to numerical precision differences resulting from retaining significant figures, the performance of SBSM-Pro is not necessarily inferior to that of T3SEpp.

For the PVP and PTSS datasets, the models proposed by Meng et al. [44] and Barukab et al. [45] are currently the best. Both models primarily utilize amino acid composition information, with the former additionally incorporating feature selection algorithms. SBSM-Pro outperforms existing methods across various metrics, with an ACC approximately 3.98% and 5.10% higher than current methods.

Li et al. [46] employed a set of nine features, including the parallel correlation pseudo amino acid composition and adapted normal distribution bi-profile Bayes, to identify PSNS. This model accounts for a rich set of information, subsequently employing the method of information gain for feature vector selection. However, SBSM-Pro, benefiting from the use of original protein sequences which avoids information loss, surpasses existing methods on various indicators, with a 2.50% increase in ACC.

Dou et al. [47] developed iGlu_AdaBoost, a tool designed for the identification of PLGS. This model integrates three feature representation methods: a 188-dimensional feature, the position of K-spaced amino acid pairs, and the enhanced amino acid composition. By applying feature selection, a 37-dimensional optimal feature subset was obtained, and predictions were performed using AdaBoost. SBSM-Pro surpasses existing methods in the three indicators of TPR, TNR, and ACC.

The iCar-PseCp [48] is utilized for the identification of PCSs, employing sequence coupling effects to describe the sequence order with the aim of preserving more information from the original sequence. With the ACC of SBSM-Pro surpassing Existing Methods comprehensively, the TPR of SBSM-Pro is higher than that of existing methods, while the TNR is lower. It is worth noting that all four datasets are for

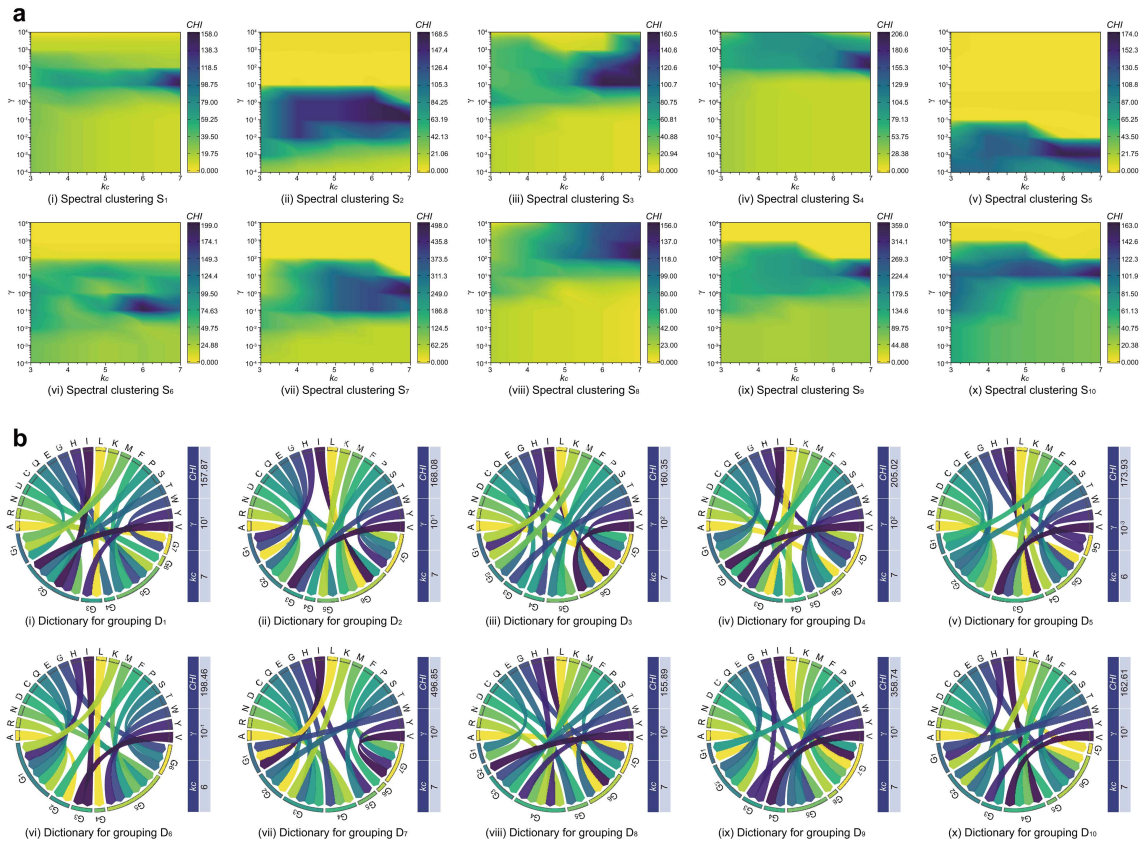

Figure 4 (Color online) Overview of the PSD process. (a) Heatmaps of grid search parameters tuning in spectral clustering; (b) visual representations of dictionaries for grouping.

PCS identification. Therefore, a higher TPR implies the ability to detect more PCS, demonstrating that SBSM-Pro has higher practical value.

In summary, when compared with SBSM-Pro, some models incorporate features from multiple dimensions, encompassing a wealth of information. However, they still inevitably suffer from information loss during feature extraction and selection. On the other hand, other models employ deep learning techniques, but this leads to overfitting. Across 10 commonly used amino acid classification datasets, SBSM-Pro generally outperforms existing methods, effectively demonstrating its superior performance, generalizability, and robustness.

3.3 Creating dictionaries for amino acid grouping by using spectral clustering

The hyperparameters γ in the Gaussian kernel function and k_c in the K-means clustering algorithm need to be specified by the user. For γ, we use a logarithmic scale from 10^{-4} to 10^4 and perform a grid search. For k_c, the number of clusters ranges from 3 to 7 with a step size of 1, also using a grid search approach.

The possible values for the hyperparameters k_c and γ constitute a parameter grid. For each combination of parameters, we trained a spectral clustering model and evaluated the performance of the clustering results. We employed CHI as our evaluation metric and selected the combination of parameters that maximizes this metric as our final configuration for the hyperparameters. For 10 different physicochemical properties of amino acids, we obtained 10 corresponding spectral clustering results. Then, we obtained 10 dictionaries for grouping based on 10 different clustering results, completing the PSD process, as shown in Appendix C. Their visual representations are depicted in Figure 4. According to the clustering results, we found that the number of clusters in dictionaries D_5 and D_6 is six, whereas the remaining eight dictionaries each comprise seven clusters. Each group in the dictionaries contains a maximum of 6 amino acids and a minimum of 1 amino acid.

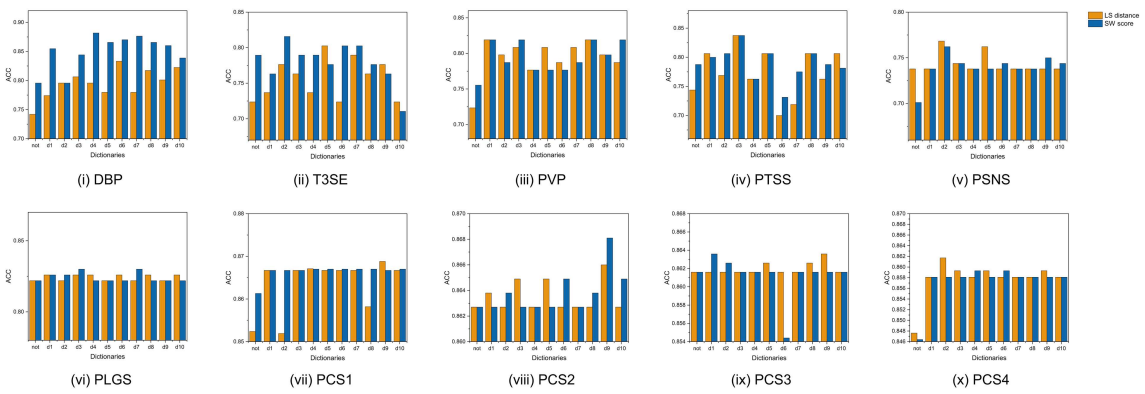

Figure 5 (Color online) Comparison of the effects of different dictionaries for grouping.

3.4 Comparison of the effect of different dictionaries for amino acid grouping

In the previous section, we obtained ten dictionaries for amino acid grouping. Based on one of these dictionaries, the amino acid residues of the original protein sequence were replaced by the identification number of their respective groups, resulting in the re-encoding of the protein.

This process can reduce interference in the sequence alignment, while also linking the original amino acid sequence information to its physicochemical properties. As a result, it effectively enhances the efficiency of LS distance and SW scores in quantifying protein similarity. To substantiate this perspective and highlight the role of the PSD process, we compared the results after substitution with different dictionaries for amino acid grouping to those without any amino acid substitution. Two methods for measuring the amino acid similarity, the LS distance and SW score, were employed to compare the results of dictionaries for grouping. The overall outcomes are illustrated in Figure 5, while the specific results are presented in Appendix C.

The results indicate that models utilizing amino acid grouping generally outperform those that do not incorporate this grouping. These results align with our expectations and demonstrate the intended benefits of amino acid grouping. For specific datasets, such as T3SE, some models exhibited lower performance when using dictionaries compared to not using them.

The results indicate that for the functional protein classification datasets DBP, T3SE, and PVP, the models based on amino acid grouping achieved significant improvements in ACC compared to those without amino acid grouping. For the other seven PTM identification datasets, the effect of amino acid grouping enhancement was not as pronounced. We attribute this result to the relatively shorter protein sequences in the PTM datasets compared to those in the amino acid function identification datasets, which weakens the noise-reducing benefits of amino acid grouping. Additionally, overall, the SW algorithm consistently outperforms the LS algorithm. This advantage is due to the SW algorithm's ability to insert gaps during sequence alignment, resulting in better sequence alignments and, consequently, a more accurate representation of sequence similarity.

The results also r unveiled another noteworthy observation: the performance of different dictionaries varies across datasets. For instance, when using the LS distance, amino acid groupings based on dictionary d_5, which includes protein secondary structure information, achieved the highest performance with the DBP dataset but performed the lowest with T3SE. The reason for this discrepancy is that different PSD processes produce dictionaries corresponding to different physicochemical properties of amino acids, and the contributions of these properties to protein classification vary among datasets.

In conclusion, the use of amino acid grouping can provide substantial performance improvements. However, no single amino acid dictionary exhibits good performance across all datasets. This introduces another challenge: the crucial task of selecting the most suitable dictionary. We innovatively addressed this concern by utilizing MKL to integrate similarity kernels generated from all dictionaries. Different kernels are assigned varying weights, leveraging the potential of each amino acid dictionary. This method will be elucidated in the following section.

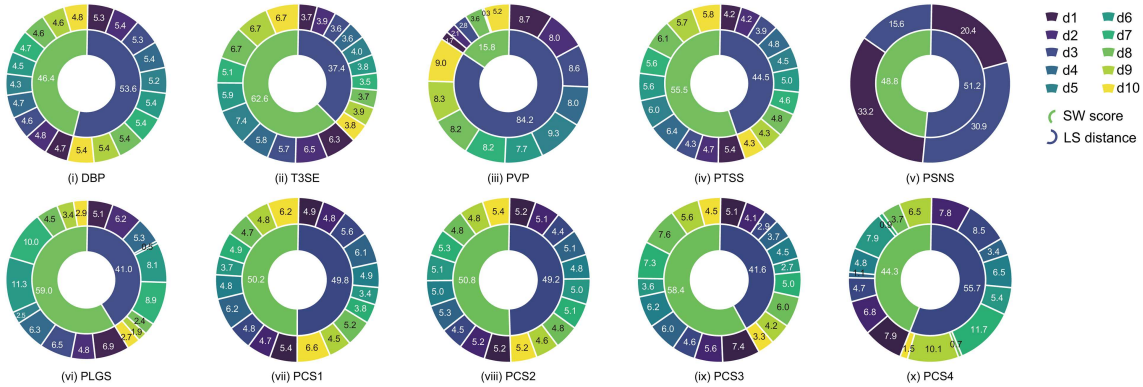

Figure 6 (Color online) Illustration of the proportional kernel weights computed by HCKDM-MKL.

3.5 Multiple kernel learning

In the previous section, we derived dictionaries for amino acid groupings, each corresponding to distinct physicochemical properties. For each dictionary, we selected two distinct sequence similarity measurement methods: the LS distance and SW scores. These two methods offer different perspectives when assessing protein sequence similarity. Consequently, by integrating 10 amino acid substitution dictionaries with the 2 sequence similarity measurement techniques, we obtained 20 protein similarity kernels. These 20 kernels represent a multidimensional evaluation of protein sequence similarity, with each kernel having a unique characterization capability.

In our grid search parameter tuning, we adjusted the parameters k and λ. The parameter k was varied in increments of 10, starting from 10 up to 100. Meanwhile, λ was tested at intervals of 0.1, ranging from 0.1 to 0.9. This systematic approach allowed us to explore a comprehensive grid of parameter combinations to identify the optimal settings for HCKDM-MKL.

Utilizing HCKDM-MKL, we obtained weights for the 20 similarity kernels. These weights signify the contribution of each similarity kernel in the fused kernel. To visually represent the weight of each similarity kernel as well as the proportions of contributions from LS distance and SW scores, we constructed a concentric ring chart, as shown in Figure 6. Examining the kernel weight figures allows us to summarize various typical patterns of kernel weights obtained through the HCKDM-MKL method. The first type, represented by T3SE, effectively utilizes all 20 similarity kernels. These kernels control the importance of different information with weights. The second type utilizes only a few or even a special similarity kernel, exemplified by PSNS. Regardless of the type, they both employ the HCKDM-MKL method to select relevant information, and their effectiveness has been demonstrated in experiments.

To highlight the effectiveness of our newly proposed MKL method, HCKDM-MKL, we also compared it with two common MKL methods: Hilbert-Schmidt independence criterion multiple kernel learning (HSIC-MKL) [51] and hybrid kernel alignment maximization multiple kernel learning (HKAM-MKL) [53]. Furthermore, we included the commonly-used method of average kernel weights for comparison, aiming to evaluate the efficacy of MKL approaches. The results are presented in Table 3 and Figure 7. We found that the performance of HCKDM-MKL consistently surpassed that of HSIC-MKL and HKAM-MKL in terms of the mean weight across all datasets. This underscores the advanced nature and robustness of our method. In Figure 7, the line graph categorizes MKL methods into three regions, A, B, and C, based on performance as delineated by the top and least-performing kernels. Area A represents MKL methods that not only select but also optimally combine various kernels, enhancing overall performance beyond any single kernel. Area B indicates methods that perform adequate kernel selection without significant performance enhancement. In contrast, Area C consists of substandard methods whose performance does not surpass the baseline set by individual kernels. We observe that, apart from the PSNS dataset, HCKDM-MKL consistently falls within area A. This suggests that it effectively accomplishes kernel fusion by appropriately assigning weights to different kernels. This even leads to a notable enhancement in the final results, which aligns perfectly with our expectations.

Table 3 Comparison of different MKL methods

Dataset	HCKDM-MKL	HSIC-MKL	HKAM-MKL	Mean weight
DBP	**0.8925**	0.8817	0.8817	0.8387
T3SE	**0.8289**	0.8026	0.8026	0.8158
PVP	**0.8298**	0.8085	0.8191	0.8298
PTSS	**0.9000**	0.8813	0.8813	0.8813
PSNS	**0.7500**	0.7378	0.7378	0.7378
PLGS	**0.8381**	0.8300	0.8300	0.8340
PCS1	**0.8737**	0.8715	0.8724	0.8724
PCS2	**0.8791**	0.8758	0.8649	0.8780
PCS3	**0.8687**	0.8677	0.8677	0.8677
PCS4	**0.8699**	0.8628	0.8640	0.8640

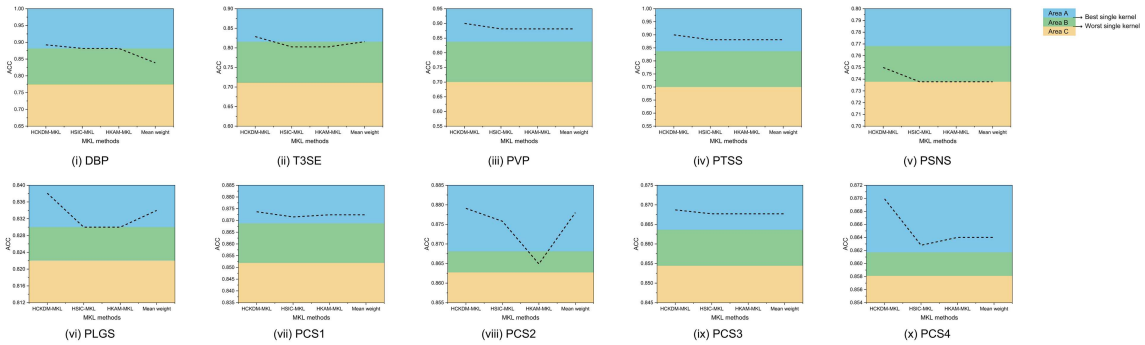

Figure 7 (Color online) Comparison of the effectiveness of different MKL methods.

4 Conclusion

The SBSM-Pro method we proposed has achieved outstanding results across multiple datasets, demonstrating its effectiveness. Through ablation studies, we further showcased the effectiveness and indispensability of each module within SBSM-Pro. Our proposed standard process, PSD, links the physicochemical properties of amino acids with grouping dictionaries, addressing the issue of excessive gaps in protein sequence alignment, and maintaining accurate similarity between protein sequences. This process can be further utilized and developed by more researchers. The SBSM-Pro method computes protein similarities using LS distance and SW score, integrated with an SVM. By extracting multifaceted information from raw protein sequences, it retains more information than traditional feature extraction techniques and achieves a higher accuracy rate. We have also proposed a new MKL method, HCKDM-MKL, for sequence classification, effectively enhancing the potential for computing protein similarity using kernel matrices. The introduction of MKL offers SVMs the possibility to integrate information from different perspectives, which will motivate more researchers to effectively improve the accuracy of sequence classification by designing a variety of sequence similarity metrics.

SBSM-Pro, which constructs sequence kernels from original sequences, has achieved outstanding results. However, numerous avenues for further exploration and in-depth research remain open. First, as part of our future endeavors, we plan to develop a graphical user interface to promote our software and make it more convenient for more biologists to use. Furthermore, we propose to analyze the structural and functional attributes of proteins, thereby assessing the similarity between protein sequences from these two perspectives. Employing deep learning methodologies to predict the three-dimensional structure of proteins, and subsequently evaluating the structural similarities through alignment and comparison tools, presents a feasible strategy for the development of the protein structure kernel in the future. Based on Gene Ontology, we can obtain the annotations of proteins through databases, and then select one or more ontologies from molecular functions, cellular components, and biological processes for analysis. By calculating the similarity between different proteins, we can construct the functional kernel of the proteins. We believe that the construction of structural and functional kernels, coupled with the MKL method we proposed, has the potential to further enhance the performance of SBSM-Pro. In addition, it is important to note that SBSM-Pro was originally designed for bio-sequences. Apart from protein sequences, it should also encompass the classification tasks of DNA and RNA sequences.

In summary, SBSM-Pro, a classification model designed specifically for biological sequences, has achieved outstanding results. This pioneering work, characterized by its scalability, will inspire an increasing number of researchers to delve into related studies. These researchers can explore methods for measuring the similarity between protein sequences from various perspectives, generate similarity kernels, and integrate them into models through MKL methods. Additionally, they can utilize the existing models to assist or even guide biological experiments, probing into the potential information of biological sequences. SBSM-Pro is available for access at http://lab.malab.cn/soft/SBSM-Pro/. The source code and datasets of SBSM-Pro are freely available in the GitHub repository at https://github.com/yzwbio/Support-Biosequence-Machine.

Acknowledgements This work was supported by the National Natural Science Foundation of China (Grant Nos. 62450002, 62425107, 62172076, U22A2038), Zhejiang Provincial Natural Science Foundation of China (Grant No. LY23F020003), and the Municipal Government of Quzhou (Grant No. 2023D036).

Supporting information Appendixes A–C. The supporting information is available online at info.scichina.com and link.springer.com. The supporting materials are published as submitted, without typesetting or editing. The responsibility for scientific accuracy and content remains entirely with the authors.

References

1 Sun A, Li H, Dong G, et al. DBPboost: a method of classification of DNA-binding proteins based on improved differential evolution algorithm and feature extraction. Methods, 2024, 223: 56–64
2 Chen W, Zhang X, Brooker J, et al. PseKNC-General: a cross-platform package for generating various modes of pseudo nucleotide compositions. Bioinformatics, 2014, 31: 119–120
3 Muhammod R, Ahmed S, Md Farid D, et al. PyFeat: a Python-based effective feature generation tool for DNA, RNA and protein sequences. Bioinformatics, 2019, 35: 3831–3833
4 Chen Z, Zhao P, Li F, et al. iFeature: a Python package and web server for features extraction and selection from protein and peptide sequences. Bioinformatics, 2018, 34: 2499–2502
5 Wang J, Du P F, Xue X Y, et al. VisFeature: a stand-alone program for visualizing and analyzing statistical features of biological sequences. Bioinformatics, 2019, 36: 1277–1278
6 Wang J, Yang B, Revote J, et al. POSSUM: a bioinformatics toolkit for generating numerical sequence feature descriptors based on PSSM profiles. Bioinformatics, 2017, 33: 2756–2758
7 Cao D S, Xiao N, Xu Q S, et al. RCPI: R/Bioconductor package to generate various descriptors of proteins, compounds and their interactions. Bioinformatics, 2014, 31: 279–281
8 Xiao N, Cao D S, Zhu M F, et al. Protr/ProtrWeb: R package and web server for generating various numerical representation schemes of protein sequences. Bioinformatics, 2015, 31: 1857–1859
9 Friedel M, Nikolajewa S, Sühnel J, et al. DiProDB: a database for dinucleotide properties. Nucleic Acids Research, 2008, 37: D37–D40
10 Kawashima S, Pokarowski P, Pokarowska M, et al. AAindex: amino acid index database, progress report 2008. Nucleic Acids Research, 2007, 36: D202–D205
11 Li X, Qian Y, Hu Y, et al. MSF-PFP: a novel multisource feature fusion model for protein function prediction. J Chem Inf Model, 2024, 64: 1502–1511
12 Chen D, Li S, Chen Y. ISTRF: Identification of sucrose transporter using random forest. Front Genet, 2022, 13: 1012828
13 Zhang Y, Ni J, Gao Y. RF-SVM: Identification of DNA-binding proteins based on comprehensive feature representation methods and support vector machine. Protein Struct Funct Bioinf, 2022, 90: 395–404
14 Chen Y, Li S, Guo J. A method for identifying moonlighting proteins based on linear discriminant analysis and bagging-SVM. Front Genet, 2022, 13: 963349
15 Ghandi M, Mohammad-Noori M, Ghareghani N, et al. gkmSVM: an R package for gapped-kmer SVM. Bioinformatics, 2016, 32: 2205–2207
16 Chen Z, Zhao P, Li C, et al. iLearnPlus: a comprehensive and automated machine-learning platform for nucleic acid and protein sequence analysis, prediction and visualization. Nucleic Acids Res, 2021, 49: e60
17 Liu B, Gao X, Zhang H. BioSeq-Analysis2.0: an updated platform for analyzing DNA, RNA and protein sequences at sequence level and residue level based on machine learning approaches. Nucleic Acids Res, 2019, 47: e127
18 Li H L, Pang Y H, Liu B. BioSeq-BLM: a platform for analyzing DNA, RNA and protein sequences based on biological language models. Nucleic Acids Res, 2021, 49: e129
19 Ghandi M, Lee D, Mohammad-Noori M, et al. Enhanced regulatory sequence prediction using gapped k-mer features. PLOS Comput Biol, 2014, 10: e1003711
20 Lee D, Gorkin D U, Baker M, et al. A method to predict the impact of regulatory variants from DNA sequence. Nature Genet, 2015, 47: 955–961
21 Jumper J, Evans R, Pritzel A, et al. Highly accurate protein structure prediction with AlphaFold. Nature, 2021, 596: 583–589
22 Avsec Ž, Kreuzhuber R, Israeli J, et al. The Kipoi repository accelerates community exchange and reuse of predictive models for genomics. Nat Biotechnol, 2019, 37: 592–600
23 Budach S, Marsico A. Pysster: classification of biological sequences by learning sequence and structure motifs with convolutional neural networks. Bioinformatics, 2018, 34: 3035–3037
24 Chen K M, Cofer E M, Zhou J, et al. Selene: a PyTorch-based deep learning library for sequence data. Nat Methods, 2019, 16: 315–318
25 Ji Y, Zhou Z, Liu H, et al. DNABERT: pre-trained bidirectional encoder representations from transformers model for DNA-language in genome. Bioinformatics, 2021, 37: 2112–2120
26 Singh R, Lanchantin J, Robins G, et al. DeepChrome: deep-learning for predicting gene expression from histone modifications. Bioinformatics, 2016, 32: i639–i648
27 Zeng H, Edwards M D, Liu G, et al. Convolutional neural network architectures for predicting DNA-protein binding. Bioinformatics, 2016, 32: i121–i127

28 Zeng H, Gifford D K. Predicting the impact of non-coding variants on DNA methylation. Nucleic Acids Res, 2017, 45: e99
29 Xu M, Chen N, Chen T, et al. DeepEnhancer: predicting enhancers by convolutional neural networks. In: Proceedings of IEEE International Conference on Bioinformatics and Biomedicine (BIBM), 2016. 637–644
30 Aoki G, Sakakibara Y. Convolutional neural networks for classification of alignments of non-coding RNA sequences. Bioinformatics, 2018, 34: i237–i244
31 Zhou J, Troyanskaya O G. Predicting effects of noncoding variants with deep learning-based sequence model. Nat Methods, 2015, 12: 931–934
32 Wang X, Ding Z, Wang R, et al. Deepro-Glu: combination of convolutional neural network and Bi-LSTM models using ProtBert and handcrafted features to identify lysine glutarylation sites. Brief Bioinform, 2023, 24: bbac631
33 Dong B, Li M, Jiang B, et al. Antimicrobial peptides prediction method based on sequence multidimensional feature embedding. Front Genet, 2022, 13: 1069558
34 Alipanahi B, Delong A, Weirauch M T, et al. Predicting the sequence specificities of DNA-and RNA-binding proteins by deep learning. Nat Biotechnol, 2015, 33: 831–838
35 Mahmud S M H, Goh K O M, Hosen M F, et al. Deep-WET: a deep learning-based approach for predicting DNA-binding proteins using word embedding techniques with weighted features. Sci Rep, 2024, 14: 2961
36 Angermueller C, Lee H J, Reik W, et al. DeepCpG: accurate prediction of single-cell DNA methylation states using deep learning. Genome Biol, 2017, 18: 1–13
37 Min X, Zeng W, Chen N, et al. Chromatin accessibility prediction via convolutional long short-term memory networks with k-mer embedding. Bioinformatics, 2017, 33: i92–i101
38 Quang D, Xie X. DanQ: a hybrid convolutional and recurrent deep neural network for quantifying the function of DNA sequences. Nucleic Acids Res, 2016, 44: e107
39 Wang X, Han L, Wang R, et al. DaDL-SChlo: protein subchloroplast localization prediction based on generative adversarial networks and pre-trained protein language model. Brief Bioinform, 2023, 24: bbad083
40 Lee H, Lee S, Lee I, et al. AMP-BERT: prediction of antimicrobial peptide function based on a BERT model. Protein Sci, 2023, 32: e4529
41 Liu Y, Liu Y, Wang S, et al. LBCE-XGB: a XGBoost model for predicting linear B-Cell epitopes based on BERT embeddings. Interdiscip Sci, 2023, 15: 293–305
42 Lu W, Song Z, Ding Y, et al. Use Chou's 5-step rule to predict DNA-binding proteins with evolutionary information. BioMed Res Int, 2020, 2020: 6984045
43 Hui X, Chen Z, Lin M, et al. T3SEpp: an integrated prediction pipeline for bacterial type III secreted effectors. mSystems, 2020, 5: e00288-20
44 Meng C, Zhang J, Ye X, et al. Review and comparative analysis of machine learning-based phage virion protein identification methods. Biochim Biophys Acta, 2020, 1868: 140406
45 Barukab O, Khan Y D, Khan S A, et al. iSulfoTyr-PseAAC: identify tyrosine sulfation sites by incorporating statistical moments via Chou's 5-steps rule and pseudo components. Curr Genomics, 2019, 20: 306–320
46 Li T, Song R, Yin Q, et al. Identification of S-nitrosylation sites based on multiple features combination. Sci Rep, 2019, 9: 3098
47 Dou L, Li X, Zhang L, et al. iGlu_AdaBoost: identification of lysine glutarylation using the adaBoost classifier. J Proteome Res, 2021, 20: 191–201
48 Jia J, Liu Z, Xiao X, et al. iCar-PseCp: identify carbonylation sites in proteins by Monte Carlo sampling and incorporating sequence coupled effects into general PseAAC. Oncotarget, 2016, 7: 34558
49 Qu X, Du G, Hu J, et al. Graph-DTI: a new model for drug-target interaction prediction based on heterogenous network graph embedding. Curr Comput Aided Drug Des, 2024, 20: 1013–1024
50 Wang Y, Zhang X, Ju Y, et al. Identification of human microRNA-disease association via low-rank approximation-based link propagation and multiple kernel learning. Front Comput Sci, 2024, 18: 182903
51 Ding Y, Tang J, Guo F. Identification of drug–target interactions via dual laplacian regularized least squares with multiple kernel fusion. Knowledge-Based Syst, 2020, 204: 106254
52 Zhou H, Wang H, Ding Y, et al. Multivariate information fusion for identifying antifungal peptides with Hilbert-Schmidt Independence Criterion. Curr Bioinform, 2022, 17: 89–100
53 Wang Y, Liu X, Dou Y, et al. Multiple kernel learning with hybrid kernel alignment maximization. Patt Recogn, 2017, 70: 104–111
54 Zhao S, Ding Y, Liu X, et al. HKAM-MKM: a hybrid kernel alignment maximization-based multiple kernel model for identifying DNA-binding proteins. Comput Biol Med, 2022, 145: 105395