Molecular Therapy
Nucleic Acids
Original Article

ac4C-AFL: A high-precision identification of human mRNA N4-acetylcytidine sites based on adaptive feature representation learning

Nhat Truong Pham,[1] Annie Terrina Terrance,[1] Young-Jun Jeon,[1] Rajan Rakkiyappan,[2] and Balachandran Manavalan[1]

[1]Department of Integrative Biotechnology, College of Biotechnology and Bioengineering, Sungkyunkwan University, Suwon, Gyeonggi-do 16419, Republic of Korea; [2]Department of Mathematics, Bharathiar University, Coimbatore, Tamil Nadu 641046, India

RNA N4-acetylcytidine (ac4C) is a highly conserved RNA modification that plays a crucial role in controlling mRNA stability, processing, and translation. Consequently, accurate identification of ac4C sites across the genome is critical for understanding gene expression regulation mechanisms. In this study, we have developed ac4C-AFL, a bioinformatics tool that precisely identifies ac4C sites from primary RNA sequences. In ac4C-AFL, we identified the optimal sequence length for model building and implemented an adaptive feature representation strategy that is capable of extracting the most representative features from RNA. To identify the most relevant features, we proposed a novel ensemble feature importance scoring strategy to rank features effectively. We then used this information to conduct the sequential forward search, which individually determine the optimal feature set from the 16 sequence-derived feature descriptors. Utilizing these optimal feature descriptors, we constructed 176 baseline models using 11 popular classifiers. The most efficient baseline models were identified using the two-step feature selection approach, whose predicted scores were integrated and trained with the appropriate classifier to develop the final prediction model. Our rigorous cross-validations and independent tests demonstrate that ac4C-AFL surpasses contemporary tools in predicting ac4C sites. Moreover, we have developed a publicly accessible web server at https://balalab-skku.org/ac4C-AFL/.

INTRODUCTION

Post-transcriptional modifications are a crucial step in rRNA processing that promote mRNA stability and decoding fidelity of protein synthesis.[1] N4-acetylcytidine (ac4C) is a post-transcriptional RNA modification catalyzed by the enzyme N-acetyltransferase 10, which involves the addition of an acetyl group to the fourth nitrogen atom of the cytidine base. These modified ribonucleotides are highly conserved across all domains of life[2] and are enriched within the coding sequences in the transcriptome.[1] ac4C was first identified at the wobble position 34 of the bacterial elongator tRNAMet,[3] and was later identified in eukaryotic tRNAs and 18S rRNA.[4] These modifications at the wobble sites of RNA enhance translation efficiency by facilitating correct codon recognition during protein synthesis.[3] Despite the importance of ac4C in fine-tuning the recognition and binding of codons, its biogenesis and biochemical role in the translation process are far from discovery.

The identification of ac4C modification sites is of critical importance to both biological studies and computational research. Experimental methods like two-dimensional thin-layer chromatography, dot blot, high-performance liquid chromatography combined with mass spectrometry,[5,6] and ac4C-specific RNA immunoprecipitation (acRIP)[1] help to pinpoint the locations of these modification sites. However, the majority of these methods have a limitation in that they cannot offer single-base resolution; they can determine the possible region of ac4C. A recent study[1] identified extensive ac4C distribution throughout the human transcriptome, with most sites situated within coding sequences. Moreover, mRNAs modified by ac4C have an extended half-life and exhibit enhanced translation efficiency. Given the limitations and the time-consuming nature of experimental methods, there is a need for computational methods capable of identifying ac4C sites both accurately and reliably.

In recent years, four computational tools have been developed for predicting ac4C sites in mRNA: PACES,[7] XG-ac4C,[8] DeepAc4C,[9] and iRNA-ac4C.[10] PACES is an ac4C site predictor that utilizes random forest (RF) classifiers trained on position-specific dinucleotide sequence profile and K-nucleotide frequencies as features. XG-ac4C is another predictor that uses eXtreme Gradient Boosting Trees (XGBT) coupled with electron-ion interaction pseudopotentials (EIIPs) of nucleotides as features. DeepAc4C is a web server based on deep learning (DL), which integrates a convolutional neural network (CNN) trained on physicochemical patterns and distributed representation of nucleic acids (NAs). Recently, Su et al.[10] developed a novel computational

Received 7 September 2023; accepted 18 April 2024;
https://doi.org/10.1016/j.omtn.2024.102192.

Correspondence: Rajan Rakkiyapan, Department of Mathematics, Bharathiar University, Coimbatore, Tamil Nadu 641046, India.
E-mail: rakkiyappan.maths@buc.edu.in
Correspondence: Balachandran Manavalan, Department of Integrative Biotechnology, College of Biotechnology and Bioengineering, Sungkyunkwan University, Suwon, Gyeonggi-do 16419, Republic of Korea.
E-mail: bala2022@skku.edu

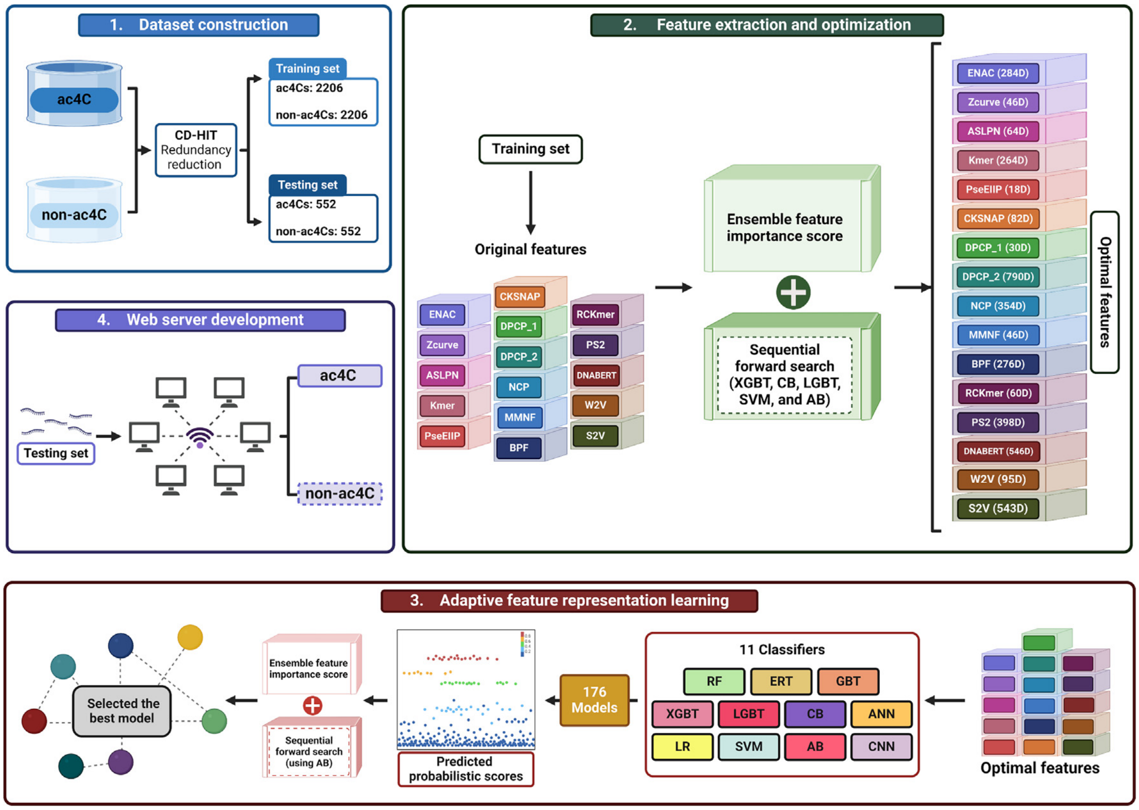

Figure 1. The workflow of constructing ac4C-AFL tool for identifying ac4C sites in human mRNA

It includes dataset construction, feature extraction and optimization, adaptive feature representation learning, and web server development.

model called iRNA-ac4C, which utilizes sequence information, such as nucleotide composition, nucleotide chemical (Ch) property, and accumulated nucleotide frequency to train a model using Gradient Boosting Decision Trees (GBT). Of these four methods, iRNA-ac4C is the most promising because it uses a larger training dataset, which resulted in better performance than the other three methods. However, iRNA-ac4C's predictive accuracy (ACC) still has room for improvement. There is potential to further enhance iRNA-ac4C's predictive capabilities by incorporating novel computational approaches.

In this study, we have implemented an adaptive feature representation strategy to develop a novel predictor, named ac4C-AFL, which accurately identifies ac4C sites from primary RNA sequences (Figure 1). First, we optimized the sequence length to capture the most relevant information around the modification sites and then employed 16 different feature encoding algorithms encapsulating composition details, position-specific information, physicochemical properties, pre-trained models, and natural language processing (NLP). Second, we have introduced a novel ensemble feature importance scoring (EFIS) strategy to rank features effectively and carried out sequential forward search to identify the optimal feature set individually from each of the 16 different features. The feature descriptors include enhanced NA composition (ENAC), position-specific of two nucleotides (PS2), composition of *k*-spaced NA pair (CKSNAP), electron-ion interaction pseudo potentials (EIIPs) of trinucleotide (PseEIIP), the Z curve parameters for frequencies of phase-specific trinucleotides (Zcurve), Kmer, reverse complement Kmer (RCKmer), dinucleotide physicochemical properties type 1 (DPCP_1), DPCP type 2 (DPCP_2), nucleotide Ch property (NCP), binary profile feature (BPF), a combination of multivariate mutual information and accumulated nucleotide frequency (MMNF), and a combination of adaptive skip dinucleotide composition and local position-specific dinucleotide frequency (ASLPN), word-to-vector (W2V), sequence-to-vector (S2V), and DNA language model-based feature (DNABERT). Third, the optimal features were trained with 11 different machine learning (ML) and DL classifiers and generated 176 baseline models. Notably, the 11 distinct ML and DL classifiers include RF, extremely randomized tree (ERT), artificial neural network (ANN), logistic regression (LR), GBT, XGBT, light GBT (LGBT), AdaBoost (AB), support vector machine (SVM), CNN, and catboost (CB). Finally, the most efficient baseline models were identified using the two-step feature selection approach, whose predicted scores were integrated and trained with the SVM classifier to develop the final prediction model. Our rigorous cross-validations (CVs) and independent tests demonstrate that ac4C-AFL surpasses contemporary tools in predicting ac4C sites. Moreover, we have established a freely available web server for ac4C-AFL at https://balalab-skku.org/ac4C-AFL/. We anticipate that ac4C-AFL will serve

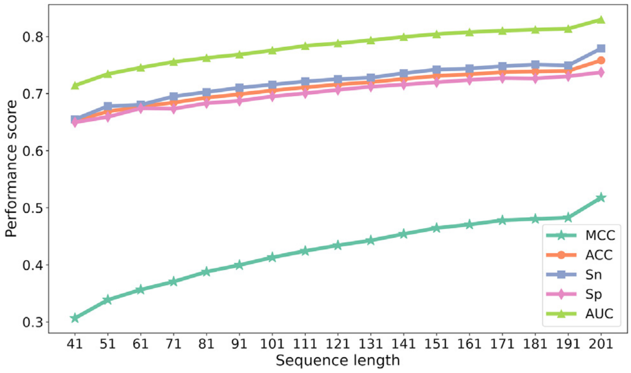

Figure 2. Performance comparison between different sequence lengths based on 16 different feature descriptors and 11 different classifiers

The average performance in terms of Matthews correlation coefficient (MCC), accuracy (ACC), sensitivity (Sn), specificity (Sp), and area under the receiver operating characteristic (ROC) curve (AUC) values with respect to different sequence lengths.

as a valuable tool in expediting the discovery of ac4C sites and aiding in the elucidation of their roles in post-transcriptional regulation.

RESULTS

Identifying the optimal sequence length for accurate ac4C prediction

Generally, researchers have used fixed sequence lengths of 41 bp or 201 bp to train models for predicting post-transcriptional modification sites.[11,12] However, it is important to explore different sequence lengths or fragments to find the optimal length that captures the most relevant information around both positive and negative samples. This is because different sequence lengths may contain different amounts of information about the modification site, and the optimal length may vary depending on the specific modification site being studied. In this regard, we generated 17 different fragments, starting with 41 bp and increasing by 10 bp (5 bp at either side), up to 201 bp. For each fragment, we generated 16 different feature descriptors and explored 11 different classifiers, whose performances were averaged that provide a straightforward global evaluation metric. This approach enabled us to compare the performance of different fragments in a fair manner. Specifically, 176 models were generated for each fragment, and their performances were averaged. In total, we generated 2,992 models. Figure 2 demonstrates a consistent increase in performance as the sequence length enlarges. It hits peak performance with a sequence length of 201 bp, suggesting that larger segments upstream and downstream of the central cytosine residues carry valuable discriminative information. This has been correctly captured by feature encoding algorithms. Our analysis of the results has led us to conclude that 201 bp is the optimal sequence length for this study. We use this length for all subsequent analyses, which are described in detail in the following sections.

Performance analysis of different feature descriptors on ML classifiers

We utilized 16 distinct feature descriptors, encapsulating composition, position-specific information, and physicochemical properties. These descriptors were then evaluated for their ability to differentiate ac4Cs from non-ac4Cs using a set of 11 different classifiers. The performance of these classifiers, represented in global metrics like ACC, Matthews correlation coefficient (MCC), and area under the receiver operating characteristic curve (AUC), can be seen in Figure 3, while the comprehensive metrics of all 176 baseline models are presented in Table S1. Results show that nine descriptors, specifically ENAC, Zcurve, ASLPN, Kmer, PseEIIP, CKSNAP, W2V, S2V, and DPCP_1, which are primarily compositional descriptors, NLP-based embeddings, and a small proportion of position-specific information, possess a higher discriminative capacity compared with the remaining descriptors (indicated by darker shades according to the color bar in Figure 3). Interestingly, we found that each feature descriptor achieved varying levels of performance when trained with different classifiers. For example, the S2V descriptor, when trained with the SVM classifier, yielded the highest performance, with an MCC of 0.619 and an ACC of 0.808. However, when used with the ANN classifier, the performance was significantly lower, with an MCC of 0.546 and an ACC of 0.772. This underlines the importance of exploring different classifiers for the same dataset to maximize performance.

To assess the overall performance of each classifier, we calculated the mean performance across all 16 descriptors. As indicated in Figure 3D, the AB classifier outperformed others, with an MCC of 0.538, ACC of 0.769, sensitivity (Sn) of 0.788, specificity (Sp) of 0.749, and an AUC of 0.842. Six other classifiers showed similar performance, with MCCs ranging from 0.520 to 0.534, slightly trailing the top-performing classifiers. While ANN was ranked last, it still demonstrated reasonable performance. Overall, all descriptors utilized in this study demonstrated robust discriminative abilities, with ACCs exceeding 72%. However, these descriptors varied widely in their dimensions, ranging from 64D to 4378D. Not all of these dimensions are of equal importance and may contain redundant or irrelevant information.[13] The exclusion of such information could potentially result in improved performance.

Optimizing each feature descriptor utilizing the two-step feature selection method

Two-step feature selection is a process of ranking features and then using sequential forward search to select the best subset of features.[14,15] In this study, a novel EFIS strategy was used to rank the features. Specifically, optimal parameters for six tree-based classifiers (RF, ERT, GBT, XGBT, LGBT, and CB) were obtained for each descriptor and used to generate respective classifier feature importance scores (FISs). Each classifier assigned FIS to a given descriptor, which was then normalized on a 0 to 1 scale, and the average FIS from all six classifiers was calculated for each feature. This ensemble score was utilized to rank the features and create multiple sets ranging from 10D to their maximum dimension with an increment of 2D, resulting in extensive feature sets. However, inputting all of these feature sets into different classifiers would require a significant amount of computation. Consequently, only AB, which demonstrated the best overall performance (as mentioned above), was selected, and its performance was evaluated using 10-fold CV.

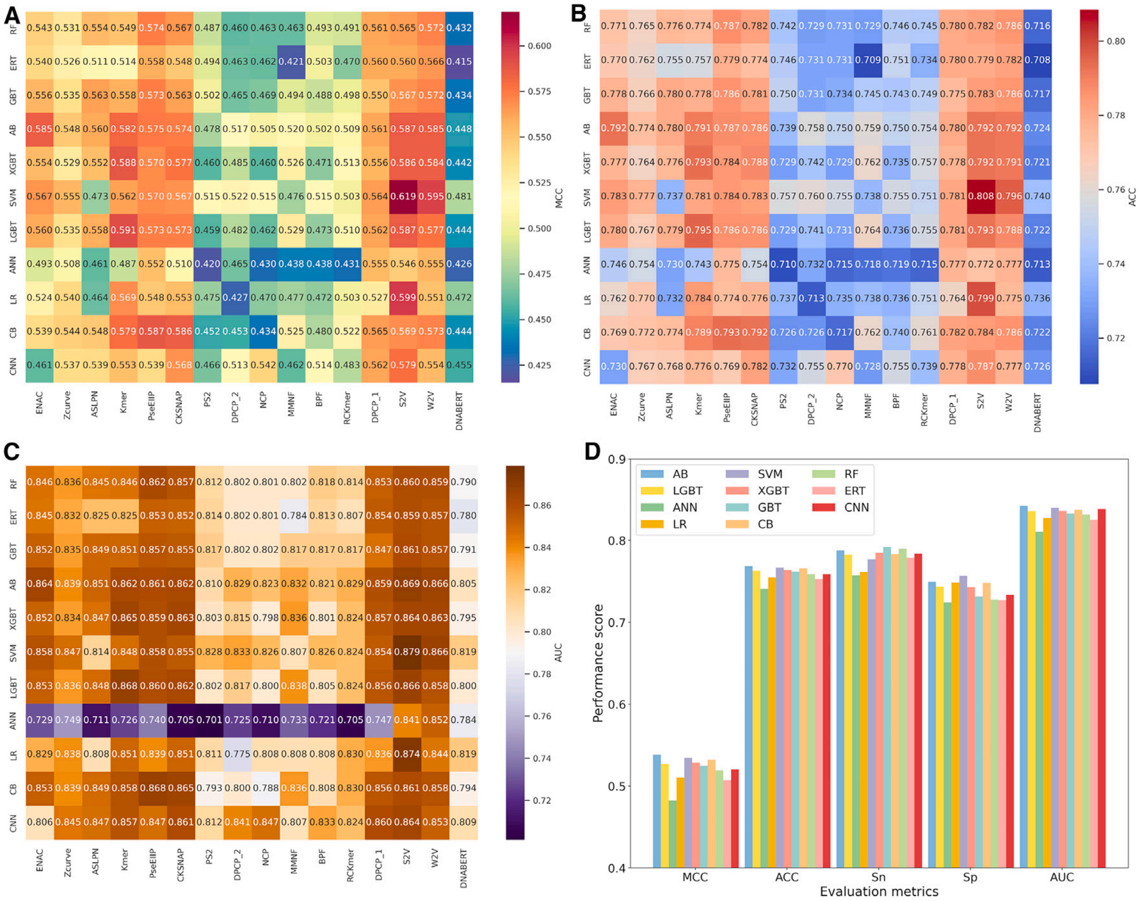

Figure 3. Performance comparison between different classifiers based on the original feature sets

(A) MCC values, (B) ACC values, (C) AUC values, and (D) average performance comparison between 11 different classifiers based on 16 feature descriptors.

Figure 4A displays that as features are continuously added, performance progressively improves until it reaches its peak and then stabilizes at that plateau. Specifically, ASLPN, BPF, CKSNAP, DPCP_1, DPCP_2, ENAC, Kmer, MMNF, NCP, PS2, PseEIIP, RCKmer, W2V, S2V, DNABERT, and Zcure achieved their optimal performance, with MCC values of 0.579, 0.525, 0.582, 0.559, 0.542, 0.594, 0.607, 0.535, 0.520, 0.526, 0.588, 0.527, 0.597, 0.596, 0.464, and 0.548, respectively (Figure 4B), utilizing optimal feature dimensions of 64D, 276D, 82D, 30D, 790D, 284D, 264D, 46D, 354D, 398D, 18D, 60D, 95D, 543D, 546D, and 46D, respectively (Figure 4C). The optimal feature achieves a similar performance to the control using Zcurve and DPCP_1, indicating that all the features are equally important for the final performance. However, for the remaining encodings, the optimal features improve the MCC by 1%–4.78%. To be precise, we have obtained the optimal feature set dimensions for ENAC, Zcurve, ASLPN, Kmer, PseEIIPP, CKSNAP, PS2, DPCP_2, NCP, MMNF, BPF, RCKmer, W2V, S2V, DNABERT, and DPCP_1, and they are 37.17%, 31.94%, 15.42%, 19.35%, 28.13%, 46.59%, 12.50%, 18.04%, 59.05%, 20%, 34.5%, 11.76%, 18.55%, 53.03%, 71.09%, and 8.52%, respectively. Overall, our approach has succeeded in significantly diminishing the feature dimension and has consequently enhanced the prediction performance in comparison with the controls.

Construction of baseline models using the optimal features

In the process of constructing the baseline models, we utilized the previously mentioned optimal feature sets from each of the 16 descriptors. These sets were used as inputs into 11 distinct classifiers that were then individually trained using a 10-fold CV technique. As illustrated in Figure 5, the performance trends of these optimized models are similar to those based on the original feature dimensions. To demonstrate the degree of performance improvement relative to the original feature dimensions, we computed the average performance of 11 classifiers for each descriptor. This was done based on the optimal feature sets, and the results were compared with those derived from the original feature sets. As demonstrated in Figure 5, it is evident that the optimal features of PseEIIP, CKSNAP, Kmer, ENAC, Zcurve, ASLPN, RCKmer, MMNF, BPF, DPCP_2, NCP, W2V, S2V, DNABERT, and PS2 contribute to

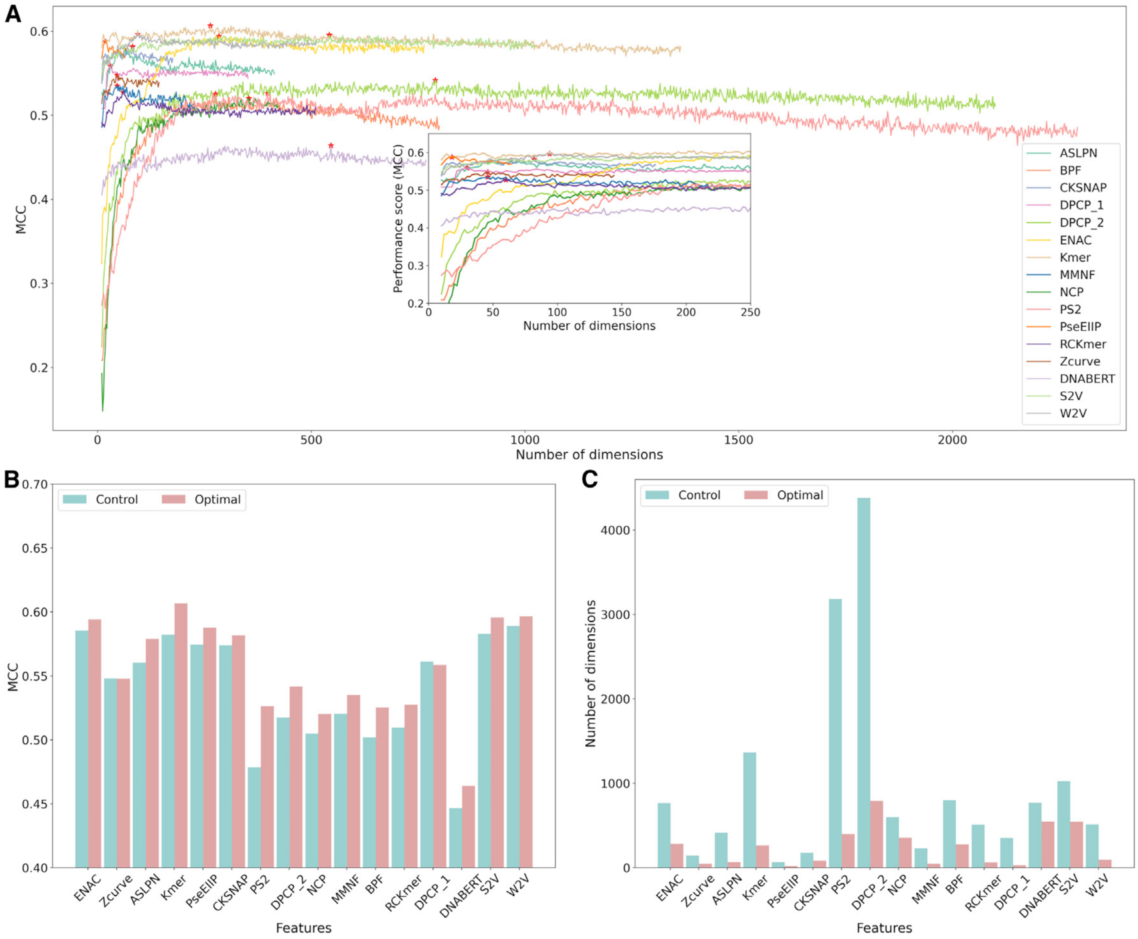

Figure 4. The performance of the EFIS-based sequential forward search algorithm, implemented with the AB classifier, was evaluated for 16 unique feature descriptors

(A) The performance was measured in terms of the MCC, and the optimal feature set that achieved the peak performance was marked with a red asterisk. (B) The MCC values between the original (control) feature set and the optimal feature set. (C) Comparison of feature dimension for each feature descriptor between the original and the optimal feature set.

notable improvements in terms of ACC, with gains of 0.65%, 0.53%, 1.11%, 0.48%, 0.46%, 1.87%, 0.96%, 1.67%, 1.28%, 1.26%, 0.92%, 0.39%, 0.34%, 0.49%, and 2.83%, respectively. However, the performance of DPCP_1 with the optimal feature set remained comparable with that with the original dimension. Crucially, the adoption of the optimal feature dimensions led to an overall enhancement of prediction performance. These 176 optimized baseline models were then leveraged for the further development of ac4C-AFL. This not only indicates the efficacy of feature optimization, but also underscores its contribution to the improved performance of ac4C-AFL.

Overall, the optimal features significantly improved the prediction performance. However, individual classifier with respect to optimal feature descriptor is improved compared with the original feature dimension. For instance, the optimal PS2 trained with RF, ERT, SVM, and LR achieved the MCC values of 0.556, 0.563, 0.567, and 0.553, which is significantly improved from the corresponding performance using the original feature dimension, 0.487, 0.494, 0.515, and 0.475. Overall, the average performance of each classifier is over the MCC of 0.510, indication that our systematic feature optimization approach significantly improves the prediction performance.

Construction of ac4C-AFL

The probabilistic scores of all 176 baseline models were combined to generate 176D features, thereby significantly strengthening the eventual predictive model. Subsequently, these features were ranked using an EFIS, and a sequential forward search was performed using 11 classifiers. The results showed that as features were added, the performance of all classifiers gradually increased and then stabilized (Figure 6). SVM achieved its maximum ACC with 110D features, while CNN reached its highest ACC with 170D features. The remaining classifiers attained their maximum ACC within the range

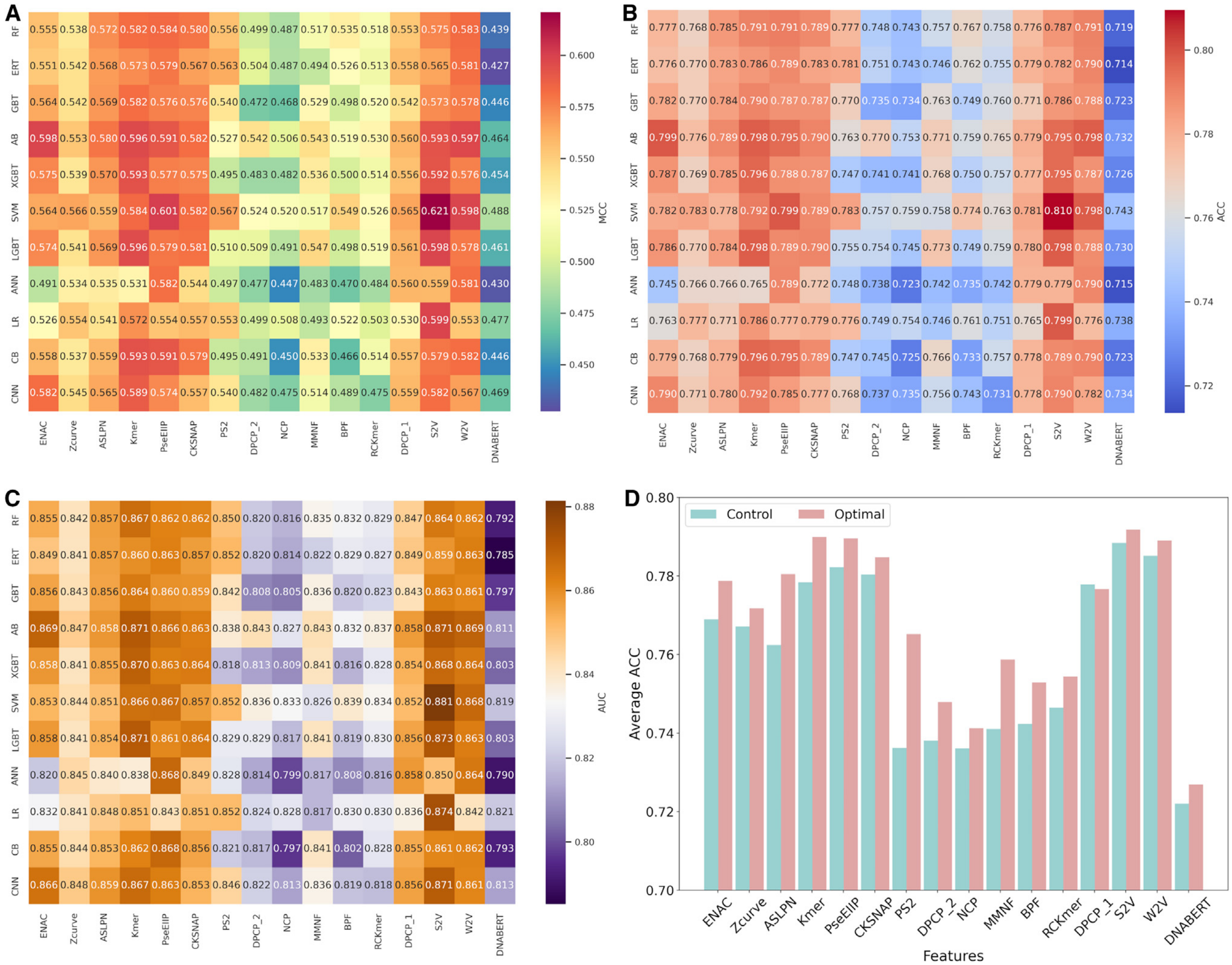

Figure 5. Performance comparison in terms of MCC, ACC, and AUC values between different classifiers based on the optimal feature sets
(A) MCC values, (B) ACC values, (C) AUC values, and (D) Comparison in terms of average ACC values between the control and the optimal feature sets.

of 50D to 100D features, except for GBT with 140D feature and LR with 120D feature.

Among the 11 classifiers, the SVM-based model utilizing the 110D probabilistic features achieved excellent performance, with MCC, ACC, Sn, Sp, and AUC values of 0.668, 0.833, 0.857, 0.810, and 0.903, respectively, as depicted in Figure 6. These results indicated that the predicted values from all baseline models were equally crucial in achieving the best performance, not only for the SVM classifier, but also for other classifiers. Consequently, we named the SVM-based prediction model as ac4C-AFL. Interestingly, the proposed approach significantly improved prediction performance compared with the existing predictor, iRNA-ac4C, when evaluated on the same training dataset (Table 1). Specifically, ac4C-AFL demonstrated a 6.70% increase in MCC, a 3.30% improvement in ACC, and a 2.80% enhancement in AUC, underscoring how the adaptive feature representation approach substantially contributed to the overall performance improvement.

Performance comparison of ac4C-AFL with the existing predictors on the independent dataset

We evaluated four different methods for predicting ac4C: iRNA-ac4C, PACES, XG-ac4C, and DeepAc4C (Table 2). While iRNA-ac4C was trained on the same dataset as ac4C-AFL, the other three methods used different datasets. Among these methods, two methods—ac4C-AFL and iRNA-ac4C—showed excellent performance. Particularly, ac4C-AFL achieved the best results, with an MCC of 0.647, an ACC of 0.823, and an AUC value of 0.895. It is noteworthy that both MCC and ACC of ac4C-AFL were 5%–50% and 2.50%–29.30% higher than the other methods compared in this study. In contrast, the remaining three methods performed well in identifying non-ac4C instances, but showed limited effectiveness in detecting ac4Cs. These three tools used a dataset that included specific motif sequences, leading to the exclusion of certain positive samples. Consequently, this selective approach might have caused an underrepresentation of positive instances in the dataset. Moreover, PACES and

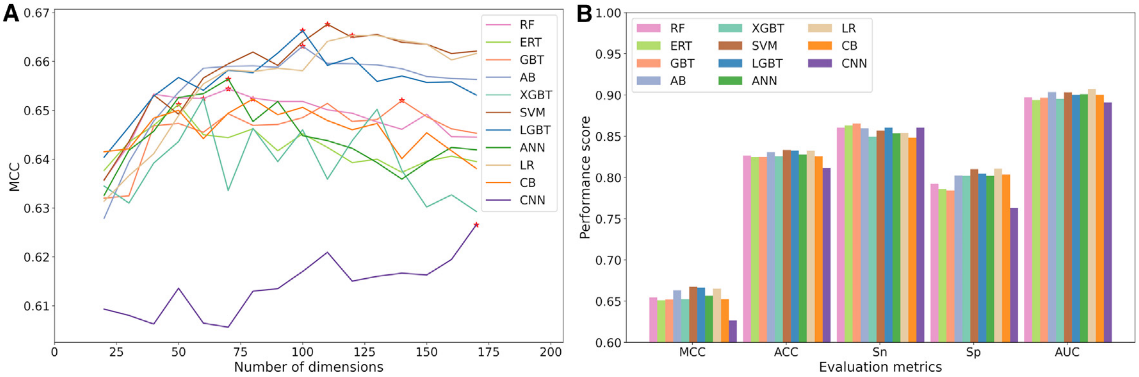

Figure 6. EFIS-based sequential forward search performance in terms of MCC graph showcasing 11 algorithms

(A) Performance comparison in terms of MCC values with respect to different feature sets for 11 different classifiers. The best performing model is highlighted in red asterisk, and (B) Comparative performance of the best models across 11 different classifiers.

XG-ac4C were trained on imbalanced datasets, with a skewed ratio of 1:10 between positive and negative samples. As a consequence, the models were more tended to learn from the abundant negative samples and potentially overlooking important features of positive instances. This imbalance likely contributed to reduced Sn and high Sp in the models' performance, rendering these less suitable for genome-wide ac4C detection.

Overall, ac4C-AFL consistently outperformed iRNA-ac4C on both the training dataset and the independent dataset, suggesting that ac4C-AFL exhibits greater stability and generalizability. This promising performance makes ac4C-AFL a strong contender for genome-wide ac4C detection, showing potential for identifying novel ac4Cs accurately.

Feature contribution analysis

To assess the effectiveness of our features, we used t-distributed stochastic neighbor embedding (t-SNE) to visualize the distribution of positive and negative samples in the 110D probabilistic feature vector. We also compared these results with the t-SNE plots of the top five individual feature descriptors, namely CKSNAP, DPCP_1, Kmer, PseEIIP, and ENAC. The t-SNE plots of the individual feature descriptors showed distinct distributions of positive and negative samples (Figures S1A–S1E). However, the t-SNE plot of the 110D vector showed a significant separation between positive and negative samples, with only a few instances overlapping (Figure S1F). These findings suggest that the 110D vector generated by our adaptive feature representation learning is better at differentiating between ac4C and non-ac4C samples than the other feature spaces. This means that our approach can significantly improve the performance of discriminating between these two classes. As a result, this approach can be applied to identify other post-transcriptional modification sites.

Table 1. Performance comparison of ac4C-AFL with the existing predictor on the training dataset

Tools	MCC	ACC	Sn	Sp	AUC
ac4C-AFL	0.668	0.833	0.857	0.810	0.903
iRNA-ac4C	0.601	0.800	0.770	0.830	0.875

Web server development

We have developed and launched a user-friendly and easy-to-use web server at https://balalab-skku.org/ac4C-AFL/to ensure wide accessibility and widespread adoption of ac4C-AFL. More specifically, we employed the Scikit-learn package to train the models, utilizing the Python programming language. Additionally, we utilized the Django framework to deploy the models, enabling seamless communication between the PostgreSQL database (back-end) and the web user interface (HTML, JavaScript, CSS – front-end). As a result, we created a simple yet efficient web server interface that allows users to obtain predicted results, interact with them, download the results in CSV format, and retrieve past job searches through the PostgreSQL database by providing the 'Job ID' and utilizing the 'Find Job' feature. Moreover, users can access the 'Help' page at https://balalab-skku.org/ac4C-AFL/help/to learn how to utilize the web server effectively. We hope that the availability of ac4C-AFL through this web server will contribute to advancements in RNA modification research, fostering collaboration and innovation within the scientific community. By facilitating the exploration of ac4C sites in human mRNA, we aspire to make meaningful contributions to the broader understanding of gene expression and its regulatory mechanisms.

DISCUSSION

RNA ac4C modification in mRNA, a significant factor in gene expression regulation, plays an essential role in post-transcriptional alterations. It provides crucial insights into transcriptional regulation mechanisms and biological processes. Consequently, accurately identifying ac4C sites in the genome using computational techniques is of paramount importance. Not only are these methods cost effective, but they also save time. Up to now, only four predictors have been designed for detecting ac4C sites in human mRNA, and there is room for enhancing prediction ACC. In this study, we propose a novel predictor, ac4C-AFL, which optimizes the length of sequences that

Table 2. Performance comparison of ac4C-AFL with the existing predictors on independent dataset

Tools	MCC	ACC	Sn	Sp	AUC
ac4C-AFL	0.647	0.823	0.844	0.803	0.895
iRNA-ac4C	0.597	0.798	0.767	0.829	0.880
PACES	0.176	0.530	0.060	1.000	NA
XG-ac4c	0.207	0.592	0.359	0.824	NA
DeepAc4C	0.147	0.536	0.010	0.971	0.803

contain important information near the modification sites. Following this, we employed a comprehensive feature optimization and adaptive feature representation learning approach. In brief, we leveraged 16 feature descriptors and identified their optimal features using a unique EFIS and AB algorithm. These optimal features were then inputted into 11 different classifiers, resulting in 176 baseline models. Subsequently, a two-step feature selection approach was employed to select the most crucial model for the final model construction. Notably, this is the first time that such a large-scale feature encoding and classifier has been used in ac4C prediction. The comparative performance of ac4C-AFL on the training and independent datasets demonstrates a significant enhancement in prediction performance compared with the most competent existing predictor, iRNA-ac4C. The benchmark dataset and the ac4C-AFL web server are accessible at https://balalab-skku.org/ac4C-AFL/.

Nevertheless, our study has limitations. Our study utilized both conventional and NLP-based feature extraction methods. However, developing novel sequence-based features through comparative analysis is crucial for further improvement. Additionally, exploring alternative computational frameworks[11,16–20] beyond the proposed one is necessary to evaluate their potential in enhancing prediction performance. Moreover, due to benchmark data limitations, ac4C-AFL is not efficient in identifying ac4Cs in other RNA types like rRNA or tRNA. Future work will involve gathering comprehensive data on a broader range of RNA types and exploring the potential of different computational frameworks that incorporate novel or additional feature extraction methods.[21,22] In conclusion, ac4C-AFL serves as a powerful tool for identifying ac4C sites in mRNA and is expected to play a pivotal role in deciphering the functional mechanisms of ac4C sites.

MATERIALS AND METHODS

Dataset construction

Recently, Su et al.[10] have employed acRIP-seq data[1] to create a reliable benchmarking dataset, which served as a basis for developing iRNA-ac4C predictor. The authors carefully curated the dataset by selecting cytidines in the close proximity to ac4C peaks as potential modification sites. Using these modification sites as central points, 100 nucleotides were garnered from either sides and labeled them as positive samples. To create negative samples, the authors randomly selected sequences from non-peak regions, each comprising 201 nucleotides with a cytidine at the center, mirroring positive samples. To ensure predictive ACC, redundant sequences with more than 80% similarity were removed using the CD-HIT.[23] To achieve a balanced dataset, an equal number of sequences were randomly selected from the negative samples, matching the count of positive samples. This compiled data was then randomly divided into training and independent testing datasets at a ratio of 80:20. After these processes, the final training dataset composed of 2,206 positive and 2,206 negative samples. The independent dataset consisted of 552 positive and 552 negative samples. It is important to note that the same dataset has been employed in this study, as it is the most recent one available. Notably, models developed based on the same data will allow for a fair comparison with existing predictors.

Framework of ac4C-AFL

In ac4C-AFL, we build a total of 176 predictive models by utilizing 16 optimal feature descriptors and 11 different classifiers. The first step in this process involves subjecting the input RNA sequences to an adaptive feature representation learning scheme. In this phase, each mRNA sequence is transformed into an n-dimensional feature vector. Subsequently, each of these feature descriptors optimized using two-step feature selection approach and identified its corresponding optimal feature set. The next step involves using 16 optimal feature sets as inputs for 11 different classifiers. Training process using 10-fold CV results in the creation of a robust, well-trained predictive model for each classifier, with each model being capable of providing a predictive score for a given mRNA sequence. This score, which ranges from 0 to 1, provides a quantitative measure of the likelihood of a sequence being an ac4C or non-ac4C.

Adaptive feature representation learning scheme

This involves two main steps: (1) feature encoding and optimization, and (2) feature representation learning and optimization, which are described in detail below.

Step 1: Feature encoding and optimization

To incorporate sufficient information in our model, we used 18 feature encoding algorithms. Note that some of the feature encodings were linearly combined resulting in 16 feature descriptors. However, the original feature set contained redundant or noisy information.[24] Not every feature calculated to characterize RNA sequence will be relevant for effective discrimination of ac4Cs. To minimize feature redundancy and computational complexity, we utilized two-step feature selection approach to select the most informative features. Generally, tree-based FIS or statistical scoring functions like F-score or minimum redundancy maximum relevance are used to rank the features.[25–27] Here, we have introduced EFIS strategy based on the FISs of six different tree-based classifiers. It is important to note that we used the optimal parameters obtained for each classifier to compute the FIS. Notably, EFIS is computed as follows.

Given an initial input feature vector $F \in \{f_1, f_2, \ldots, f_d, \ldots, f_D\}$, where $d = 1, 2, \ldots, D$, D is the number of dimensions of the input feature. Assume a scoring function $S_c(d) \in [0, 1]$ is applied to rank the importance of each dimension d, where $c = 1, 2, \ldots, C$, C is the number of scoring functions. As a result, an EFIS function can be defined as follows:

$$EFIS = \frac{1}{C}\sum_{c=1}^{C} S_c(F_d). \quad \text{(Equation 1)}$$

Employing EFIS, we ranked the features from the highest to lowest and generated various feature subsets, starting with 10D and increasing by increments of 2D up to the actual original dimension. Each of these subsets was inputted into AB classifier and trained to its respective model. The feature subset that achieved the highest performance (based on MCC) was considered the optimal feature set. The details of the feature descriptors are briefly introduced below, with their summarization presented in Table S2.

ENAC. ENAC is an improved version that builds upon NA composition (NAC) by employing a continuously sliding fixed-length window throughout the input RNA sequence, spanning from the N-terminus to the C-terminus. This is utilized to calculate the frequency of each specific NA type within the designated window. Given a sliding window (SW), the frequency (f) can be computed as:

$$f(NA_i) = \frac{\sum(NA_i)}{SW}, \quad \text{(Equation 2)}$$

where $NA_i \in \{A, C, G, U\}$ and $\Sigma(NA_i)$ is the total number of NA type NA_i. Notably, the cumulative frequencies of NAs within a sliding window SW must add up to 1. For instance, assuming the sliding window ($SW = 5$) encompasses [A, G, C, U, A] as an RNA subsequence, the resulting array should display [0.4, 0.2, 0.2, 0.2], representing the frequency distribution of NA types A, C, G, and U within the SW, respectively.

PS2. PS2 refers to the pairs of adjacent nucleotides in a pairwise manner, namely AA, AC, AG, AU, CA,…UU, resulting to a total of 16 pairs.[28] The 16 pairs are encoded into 16 binary bits, either 0 or 1. For instance, AC is represented by (0100000000000000), and GU is represented by (0000000000010000). As a result, an RNA sequence ACGU is encoded as (010000000000000000000010000000000000 000000010000).

CKSNAP. CKSNAP integrates the principles of NAC and PS2 concepts in the context of NA pair. Notably, it systematically computes the frequencies of 16 NA pairs by any k-spaced NA, where k can vary from 0 to 5. The 16 pairs are the same as those in PS2, namely AA, AC, AG, AU, CA, …, UU; however, the pairs are separated by a k-spaced NA. For instance, if $k = 0$, the 16 pairs in the CKSNAP are constructed in the same manner as those in the PS2. Differing from PS2, these 16 pairs are then calculated using NAC rather than binary bits for encoding. In practice, the feature vector of each k-spaced case for a given RNA sequence of length L can be defined as follows:

$$CKSNAP = \left[f\left(NA_{xy}\right)^{(i)}, f\left(NA_{xy}\right)^{(i+1)}, \ldots, f\left(NA_{xy}\right)^{(P)}\right], \quad \text{(Equation 3)}$$

where $f(NA_{xy})$ is the frequency of the paired NA xy and it can be calculated as:

$$f\left(NA_{xy}\right) = \frac{\sum\left(NA_{xy}\right)}{L - k+1}, \quad \text{(Equation 4)}$$

$NA_{xy} \in \{AA, AC, AG, AU, CA, \ldots, UU\}$, $i = 1, 2, 3, \ldots, P$ with $P = 16$ is the number of pairs, and $\Sigma(NA_{xy})$ is the total number of the paired NA xy in the sequence

PseEIIP. The EIIP values were used to encode DNA or protein sequences by calculating the energy of delocalized electrons in nucleotide or amino acid sequences. In the context of an RNA sequence, the EIIP values of nucleotides are encoded as follows: EIIPs of A is 0.1260, C is 0.1340, G is 0.0806, and U (which is analogous to T) is 0.1335, respectively[29], as reported in.[30]

Based on the EIIP encoding, the PseEIIP encoding vector is calculated by combining the concepts of the EIIP and the ENAC encodings as:

$$PseEIIP = \left[\left(EIIP_{xyz} \times f\left(NA_{xyz}\right)\right)^{(i)}, \ldots, \left(EIIP_{xyz} \times f\left(NA_{xyz}\right)\right)^{(T)}\right], \quad \text{(Equation 5)}$$

where $EIIP_{xyz} = EIIP_x + EIIP_y + EIIP_z$, $f(NA_{xyz})$ is the normalized frequency of the trinucleotide, $NA_{xyz} \in \{AAA, AAC, AAG, AAU, \ldots, UUU\}$, and $i = 1, 2, 3, \ldots, T$ with T as the number of trinucleotides. It is worth noting that $f(NA_{xyz})$ is calculated by applying NAC for the trinucleotide.

Zcurve. Zcurve contains the information about frequencies of phase-specific tri-nucleotides. A detailed description has been provided in previous study,[31] and it can be computed as follows:

$$\begin{cases} i_{RS}^{z} = \left(p^z(RSA) + p^z(RSG)\right) - \left(p^z(RSC) + p^z(RSU)\right) \\ i_{RS}^{z} = \left(p^z(RSA) + p^z(RSC)\right) - \left(p^z(RSU) + p^z(RSG)\right), \\ i_{RS}^{z} = \left(p^z(RSA) + p^z(RSU)\right) - \left(p^z(RSC) + p^z(RSG)\right) \end{cases} \quad \text{(Equation 6)}$$

where $R \; or \; S \in \{A, U, G, C\}; z = 1, 2, 3.$

Kmer. Kmer encoding algorithm is based on NAC in the context of k neighboring NAs. In practice, the Kmer is computed with $k \in \{1, 2, 3, 4, 5\}$ that represents mononucleotide, dinucleotide, trinucleotide, tetranucleotide, and pentanucleotide, respectively, resulting in an output of 1364D feature vector ($4^1 + 4^2 + 4^3 + 4^4 + 4^5$). Given an RNA sequence length of L, the frequencies of $k = 2$ can be computed as:

$$f(NA_{km}) = \frac{\sum(NA_{km})}{L}, \quad \text{(Equation 7)}$$

where $NA_{km} \in \{AA, AC, AG, AU, CA, \ldots, UU\}$ and $\Sigma(NA_{km})$ is the number of types of k neighboring NAs in the given sequence.

RCKmer. RCKmer is a specific type of Kmer constructed by removing the reverse complement kmers for each type of k neighboring NAs in the given sequence.[32,33] For instance, with $k = 2$, the following kmers should be removed, namely 'CU,' 'GG,' 'GU, 'UC,' 'UG,' and 'UU,' resulting in a10 discriminatively remaining kmers in the RCKmer.

DPCP_1. There are 22 physicochemical properties for dinucleotides for RNA sequence.[28,34] Most of them can be extracted for any dinucleotides, such as shift, slide, stacking energy, rise, enthalpy (Ch), enthalpy (physical [Ph]), entropy (Ch), entropy (Ph), hydrophilicity (Ch), hydrophilicity (Ph), tilt, roll, free energy (Ch), free energy (Ph), and twist. Whereas the others can only be extracted for specific dinucleotides, namely keto (GU), adenine content, guanine content, GC content, cytosine content, purine (AG) content, and thymine content. It is worth noting that the default value is zero for any dinucleotides, except the specific ones mentioned.

The DPCP_1 can be defined as:

$$DPCP\_1 = \left[\left(PCP_{xy}^{(i)} \times f(xy)\right)^{(j)}, \ldots, \left(PCP_{xy}^{(i+1)} \times f(xy)\right)^{(j+1)}, \ldots, \left(PCP_{xy}^{(N_{PCP})} \times f(xy)\right)^{(N_D)}\right], \quad \text{(Equation 8)}$$

where $x, y \in \{A, C, G, U\}$, $i = 1, 2, 3, \ldots, N_{PCP}$ with N_{PCP} is the total number of physicochemical properties, $j = 1, 2, 3, \ldots, N_D$ with N_D is the number of dinucleotides, $f(xy)$ is the normalized frequency of the dinucleotide, and $PCP^{(i)}$ denotes the i-th physicochemical property. As a result, the output of DPCP_1 should be 352D vector (16×22) corresponding with the number of dinucleotides and the number of physicochemical properties, respectively.

DPCP_2. The DPCP_2 is an enhanced version of the DPCP_1. Given an RNA sequence length of L, the DPCP_2 can be defined as:

$$DPCP\_2 = \left[PCP^{(i)}\left(NA_x^{(m)} NA_y^{(n)}\right), \ldots, PCP^{(i+1)}\left(NA_x^{(m+1)} NA_y^{(n+1)}\right), \ldots, PCP^{(N_{PCP})}\left(NA_x^{(L-1)} NA_y^{(L)}\right)\right], \quad \text{(Equation 9)}$$

where $NA_x, NA_y \in \{AA, AC, AG, AU, CA, \ldots, UU\}$, $m = 1, 2, \ldots, (L - 1)$, $n = (m+1), (m+2), \ldots, L$, and $PCP^{(i)}(NA_x^{(m)} NA_y^{(n)})$ is the i-th physicochemical property of the dinucleotide, $NA_x^{(m)} NA_y^{(n)}$.

NCP. For each of the four natural nucleotide types (A, C, G, and U), there are three types of Ch properties: ring structure (purine: A, G, and pyrimidine: C, U), functional group (amino: A, C, and keto: G, U), and hydrogen bond (strong: C, G, and weak: A, U). Each property has two classes, representing different nucleotide types with distinct Ch characteristics. Based on Ch properties [2], each NA will be encoded into three-coordinates (X, Y, Z) as:

$$X_{NA_i} = \begin{cases} 0 & if \quad NA_i \in \{U, C\} \\ 1 & if \quad NA_i \in \{G, A\} \end{cases}; \quad Y_{NA_i} = \begin{cases} 0 & if \quad NA_i \in \{U, G\} \\ 1 & if \quad NA_i \in \{C, A\} \end{cases}; \quad Z_{NA_i} = \begin{cases} 0 & if \quad NA_i \in \{G, C\} \\ 1 & if \quad NA_i \in \{U, A\} \end{cases}. \quad \text{(Equation 10)}$$

As a result, A, C, G, and U can be encoded by 1, 1, 1, 0, 1, 0, 1, 0, 0, and 0, 0, 1, respectively.

BPF. The BPF is also known as one-hot encoding. It is widely used for encoding DNA, RNA, peptide, and protein sequences. In the context of an RNA sequence, each NA is encoded by a 4D binary vector, namely A (1000), C (0100), G (0010), and U (0001), respectively. As a result, given an RNA sequence length of L, the output vector should be flattened with the dimension of $4 \times L$.

ASLPN. The ASLPN is utilized by combining both adaptive skipped dinucleotide composition (ASDC) and local position-specific dinucleotide frequency (LPSDF). The detail of each of them can be briefly presented as below.

The ASDC encoding is an enhanced version of the dinucleotide composition by integrating the k-skip-n-gram model. Specially, it uses k-skipped nucleotides while computing the n-gram model so that both distance and composition information are integrated. Due to the dimension of the output, feature vector can be exponentially increased by $n - gram$, only the case of $n = 2$ (dinucleotide) is analyzed for this kind of feature encoding algorithm. Given an RNA sequence length of L, the output feature vector of the ASDC can be denoted as:

$$ASDC = \left[f_1(NA_{xy}), f_2(NA_{xy}), \ldots, f_i(NA_{xy}), \ldots, f_{16}(NA_{xy})\right], \quad \text{(Equation 11)}$$

where $i = 1, 2, \ldots, 16$, $NA_{xy} \in \{AA, AU, AC, AG, \ldots, UU\}$, and the occurrence frequency of all possible dinucleotide based on k-skipped nucleotides can be computed by:

$$f_i(NA_{xy}) = \frac{\sum_{k=1}^{L-1} NA_{xy}^{(k)}}{\sum_{i=1}^{16} \sum_{k=1}^{L-1} NA_{xy}^{(k)}}. \quad \text{(Equation 12)}$$

The LPSDF is another dinucleotide composition that calculates the frequency of the dinucleotide constructed by the nucleotide at the specific position and the previous position within an RNA sequence. The occurrence frequencies of such a dinucleotide at position i-th can be computed by:

$$f_i = \frac{\sum\left(NA_x^{(i-1)} NA_y^{(i)}\right)}{SW}, \qquad \text{(Equation 13)}$$

where $SW = L - i$, SW is the length of the subsequence $\{NA_x^{(1)}, NA_x^{(2)}, \ldots, NA_x^{(i)}\}$, L is the length of the given RNA sequence, and $NA_x, NA_y \in \{A, C, G, U\}$.

MMNF. The MMNF is constructed by incorporating multivariate mutual information (MMI) with accumulated nucleotide frequency (ANF). The process of extracting the MMI and ANF is described as below.

To obtain the MMI encoding,[35] the frequencies of k-mer with $k \in \{2, 3\}$ are utilized. Consequently, the corresponding mutual information can be computed as below.

For $k = 2$,

$$MI_2\left(NA_{xy}\right) = f\left(NA_{xy}\right)\ln\left(\frac{f\left(NA_{xy}\right)}{f(NA_x)f\left(NA_y\right)}\right). \qquad \text{(Equation 14)}$$

And for $k = 3$,

$$MI_3\left(NA_{xyz}\right) = f\left(NA_{xy}\right)\ln\left(\frac{f\left(NA_{xy}\right)}{f(NA_x)f\left(NA_y\right)}\right) + \frac{f(NA_{xz})}{f(NA_z)}\ln\left(\frac{f(NA_{xz})}{f(NA_z)}\right) - \frac{f\left(NA_{xyz}\right)}{f\left(NA_{yz}\right)}\ln\left(\frac{f\left(NA_{xyz}\right)}{f\left(NA_{yz}\right)}\right). \qquad \text{(Equation 15)}$$

Here, $NA_x, NA_y, NA_z \in \{A, C, G, U\}$ and $f(NA_x), f(NA_y), f(NA_z)$ are their frequencies in the RNA sequence; $f(NA_{xy}), f(NA_{xz})$ are the frequencies of the '2-mer' NA_{xy} (e.g., AA, CC, GG, UU, etc.); and $f(NA_{xyz})$ is the frequency of the '3-mer' NA_{xyz} (e.g., AAA, CCC, GGG, UUU, etc.).

The ANF is the combination of the accumulated NAC with the NCP. The accumulated frequency can be defined as below:

$$\rho = \frac{1}{N_{NA_x}}\sum_{i}^{L} f(NA_x), \qquad \text{(Equation 16)}$$

where L is the length of the RNA sequence, N_{NA_x} is the occurrence number of nucleotides NA_x ($NA_x \in \{A, C, G, U\}$) in the prefix sequence $[1, 2, \ldots, i]$, and the $f(NA_x)$ will be encoded by binary bits $\in \{0, 1\}$. For instance, given an RNA sequence 'ACGUUGCA,' the output NCP encoding should be {(1, 1, 1), (0, 1, 0), (1, 0, 0), (0, 0, 1), (0, 0, 1), (1, 0, 0), (1, 1, 1), (0, 1, 0)}. Meanwhile, the accumulated frequency ρ values of 'A' are 1 and 0.25 at positions 1 and 8, respectively; the accumulated frequency ρ values of 'C' are 0.50 and 0.29 at positions 2 and 7, respectively; the accumulated frequency ρ values of 'G' are 0.33 and 0.33 at positions 3 and 6, respectively; and the accumulated frequency ρ values of 'U' are 0.25 and 0.40 at positions 4 and 5, respectively. Consequently, the final output ANF encoding after incorporating these values should be {(1, 1, 1, 1), (0, 1, 0, 0.50), (1, 0, 0, 0.33), (0, 0, 1, 0.25), (0, 0, 1, 0.40), (1, 0, 0, 0.33), (0, 1, 0, 0.29), (1, 1, 1, 0.25)}. In this way, ANF enhances the NCP encoding by incorporating the long-range sequential order information.

DNABERT. DNABERT[21] is a pre-trained bidirectional encoder representation of DNA sequences that captures sequence information based on contextual relationships between k-mers in the input DNA sequence. The DNABERT model leverages a Transformer architecture to effectively analyze the intricate connections between nucleotides in both forward and reverse directions. DNABERT utilizes a kmer-based tokenizer which generates distinct tokens called 'k-mers' from the genomic DNA, which are then processed through 12 Transformer blocks. The transformer blocks analyze the relationship between k-mers and generate in-depth representations of the DNA sequence, which captures the contextual information within the genetic data. Notably, DNABERT has shown applicability to RNA, as RNA differs from DNA only by a single base and preserves almost identical genetic information. This allows processing of RNA sequences simply by replacing nucleotide 'U' with nucleotide 'T.' When provided with an RNA sequence, DNABERT generates a feature vector of 768D.

W2V. W2V[36] is a neural network-based model that efficiently captures semantic and syntactic word relationships by considering the context in which words appear in a large corpus of text. The Word2Vec model encompasses both the Continuous Bag-of-Words (CBOW) and Skip-gram architectures. CBOW predicts a target word based on its context words, whereas Skip-gram predicts context words for a given target word. Both models are employed independently within the Word2Vec framework, and they are trained on large collections of textual data. This enables conversion of words into vector representations that convey lexicon-wide semantic links. In this context, using W2V, a feature vector of 512D was extracted from the RNA sequence.

S2V. S2V[37] model transforms sequences to vector representations, leveraging the Embeddings from Language Models (ELMos) framework. ELMo integrates a character-aware CNN layer with two bidirectional long short-term memory (Bi-LSTM) layers. The addition of the CNN layer facilitates the extraction of information from individual characters, thus molding the representations of tokens. The generated tokens are further improved by incorporating two consecutive Bi-LSTM layers for generation of embeddings. All layers in the ELMo architecture are seamlessly integrated by a top layer, ultimately resulting in comprehensive representations. When applied to RNA sequence, it generates a feature vector of 1024D.

Step 2: Feature representation learning and optimization

Subsequently, we used 10 different classifiers to train and evaluate all these optimal feature descriptors. Each trained model assigns a predicted probability score of ac4Cs for each training or testing sample. We subsequently concatenated all 176 predictions generated by all models into a new feature vector. Thus, for a given RNA sequence, it is ultimately represented by a 176D vector. To enhance the feature representation ability, we further optimized the feature representations using the two-step feature selection method as mentioned in step 1. Briefly, EFIS is calculated again to rank the features and generate feature subsets ranging from 20D to 170D, with an interval of 10D. All these feature subsets are inputted into 11 different classifiers. The performances of all these models were compared and the best model was selected.

CV and performance evaluation

We utilized a total of 11 distinct ML and DL classifiers for this study. To optimize all the hyperparameters associated with these ML and DL algorithms, we implemented a 10-fold CV technique. A comprehensive explanation of the 10-fold CV process, as well as the search ranges for the ML and DL hyperparameters, can be found in our previous studies.[38–40]

Several performance metrics[41,42] are commonly employed to assess the effectiveness of each model, including ACC, Sn, Sp, MCC, and AUC. The mathematical equations of ACC, Sn, Sp, and MCC, are given below:

$$ACC = \frac{TP+TN}{TP+TN+FP+FN}, \qquad \text{(Equation 17)}$$

$$MCC = \frac{TP \times TN - FP \times FN}{\sqrt{(TP+FP) \times (TP+FN) \times (TN+FP) \times (TN+FN)}}, \qquad \text{(Equation 18)}$$

$$Sn = \frac{TP}{TP+FN}, \qquad \text{(Equation 19)}$$

$$Sp = \frac{TN}{TN+FP}. \qquad \text{(Equation 20)}$$

The number of true positives, true negatives, false positives, and false negatives, respectively, is represented by TP, TN, FP, and FN.

DATA AND CODE AVAILABILITY

Web server can be accessed via https://balalab-skku.org/ac4C-AFL/ and all the processed data used in this study can be downloaded from the web server.

SUPPLEMENTAL INFORMATION

Supplemental information can be found online at https://doi.org/10.1016/j.omtn.2024.102192.

ACKNOWLEDGMENTS

This work was supported by the National Research Foundation of Korea (NRF) grant funded by the Korean government (MSIT) (RS-2023-00217881). This research was supported by the Department of Integrative Biotechnology, Sungkyunkwan University (SKKU) and the BK21 FOUR Project.

AUTHOR CONTRIBUTIONS

N.T.P.: Methodology, software, validation, data curation, writing - original draft; A.T.T.: methodology, validation, writing - original draft; Y.-J.J.: validation, writing - review and editing; R.R.: methodology, validation, writing - review and editing; B.M.: conceptualization, methodology, validation, writing - original draft, writing - review and editing, supervision.

DECLARATION OF INTERESTS

The authors declare no competing interests.

REFERENCES

1. Arango, D., Sturgill, D., Alhusaini, N., Dillman, A.A., Sweet, T.J., Hanson, G., Hosogane, M., Sinclair, W.R., Nanan, K.K., Mandler, M.D., et al. (2018). Acetylation of Cytidine in mRNA Promotes Translation Efficiency. Cell *175*, 1872–1886.e24. https://doi.org/10.1016/j.cell.2018.10.030.

2. Thomas, J.M., Briney, C.A., Nance, K.D., Lopez, J.E., Thorpe, A.L., Fox, S.D., Bortolin-Cavaille, M.L., Sas-Chen, A., Arango, D., Oberdoerffer, S., et al. (2018). A Chemical Signature for Cytidine Acetylation in RNA. J. Am. Chem. Soc. *140*, 12667–12670. https://doi.org/10.1021/jacs.8b06636.

3. Stern, L., and Schulman, L.H. (1978). The role of the minor base N4-acetylcytidine in the function of the Escherichia coli noninitiator methionine transfer RNA. J. Biol. Chem. *253*, 6132–6139. https://doi.org/10.1016/S0021-9258(17)34590-8.

4. Boccaletto, P., Stefaniak, F., Ray, A., Cappannini, A., Mukherjee, S., Purta, E., Kurkowska, M., Shirvanizadeh, N., Destefanis, E., Groza, P., et al. (2022). MODOMICS: a database of RNA modification pathways. 2021 update. Nucleic Acids Res. *50*, D231–D235. https://doi.org/10.1093/nar/gkab1083.

5. Jin, G., Xu, M., Zou, M., and Duan, S. (2020). The Processing, Gene Regulation, Biological Functions, and Clinical Relevance of N4-Acetylcytidine on RNA: A Systematic Review. Mol. Ther. Nucleic Acids *20*, 13–24. https://doi.org/10.1016/j.omtn.2020.01.037.

6. Zhang, Y., Lu, L., and Li, X. (2022). Detection technologies for RNA modifications. Exp. Mol. Med. *54*, 1601–1616. https://doi.org/10.1038/s12276-022-00821-0.

7. Zhao, W., Zhou, Y., Cui, Q., and Zhou, Y. (2019). PACES: prediction of N4-acetylcytidine (ac4C) modification sites in mRNA. Sci. Rep. *9*, 11112. https://doi.org/10.1038/s41598-019-47594-7.

8. Alam, W., Tayara, H., and Chong, K.T. (2020). XG-ac4C: identification of N4-acetylcytidine (ac4C) in mRNA using eXtreme gradient boosting with electron-ion interaction pseudopotentials. Sci. Rep. *10*, 20942. https://doi.org/10.1038/s41598-020-77824-2.

9. Wang, C., Ju, Y., Zou, Q., and Lin, C. (2021). DeepAc4C: a convolutional neural network model with hybrid features composed of physicochemical patterns and distributed representation information for identification of N4-acetylcytidine in mRNA. Bioinformatics *38*, 52–57. https://doi.org/10.1093/bioinformatics/btab611.

10. Su, W., Xie, X.Q., Liu, X.W., Gao, D., Ma, C.Y., Zulfiqar, H., Yang, H., Lin, H., Yu, X.L., and Li, Y.W. (2023). iRNA-ac4C: A novel computational method for effectively detecting N4-acetylcytidine sites in human mRNA. Int. J. Biol. Macromol. *227*, 1174–1181. https://doi.org/10.1016/j.ijbiomac.2022.11.299.

11. Chen, R., Li, F., Guo, X., Bi, Y., Li, C., Pan, S., Coin, L.J.M., and Song, J. (2023). ATTIC is an integrated approach for predicting A-to-I RNA editing sites in three species. Brief. Bioinform. *24*, bbad170. https://doi.org/10.1093/bib/bbad170.

12. Shoombuatong, W., Basith, S., Pitti, T., Lee, G., and Manavalan, B. (2022). THRONE: A New Approach for Accurate Prediction of Human RNA N7-Methylguanosine Sites. J. Mol. Biol. *434*, 167549. https://doi.org/10.1016/j.jmb.2022.167549.

13. Boopathi, V., Subramaniyam, S., Malik, A., Lee, G., Manavalan, B., and Yang, D.C. (2019). mACPpred: A Support Vector Machine-Based Meta-Predictor for Identification of Anticancer Peptides. Int. J. Mol. Sci. *20*, 1964. https://doi.org/10.3390/ijms20081964.

14. Ao, C., Ye, X., Sakurai, T., Zou, Q., and Yu, L. (2023). m5U-SVM: identification of RNA 5-methyluridine modification sites based on multi-view features of physicochemical features and distributed representation. BMC Biol. *21*, 93. https://doi.org/10.1186/s12915-023-01596-0.

15. Yuan, S.S., Gao, D., Xie, X.Q., Ma, C.Y., Su, W., Zhang, Z.Y., Zheng, Y., and Ding, H. (2022). IBPred: A sequence-based predictor for identifying ion binding protein in phage. Comput. Struct. Biotechnol. J. *20*, 4942–4951. https://doi.org/10.1016/j.csbj.2022.08.053.

16. Wang, R., Jiang, Y., Jin, J., Yin, C., Yu, H., Wang, F., Feng, J., Su, R., Nakai, K., Zou, Q., and Wei, L. (2023). DeepBIO: an automated and interpretable deep-learning platform for high-throughput biological sequence prediction, functional annotation and visualization analysis. Nucleic Acids Res. *51*, 3017–3029. https://doi.org/10.1093/nar/gkad055.

17. Wang, C., and Zou, Q. (2023). Prediction of protein solubility based on sequence physicochemical patterns and distributed representation information with DeepSoluE. BMC Biol. *21*, 12. https://doi.org/10.1186/s12915-023-01510-8.

18. Abbas, Z., Rehman, M.U., Tayara, H., Zou, Q., and Chong, K.T. (2023). XGBoost framework with feature selection for the prediction of RNA N5-methylcytosine sites. Mol. Ther. *31*, 2543–2551. https://doi.org/10.1016/j.ymthe.2023.05.016.

19. Pham, N.T., Rakkiyapan, R., Park, J., Malik, A., and Manavalan, B. (2023). H2Opred: a robust and efficient hybrid deep learning model for predicting 2'-O-methylation sites in human RNA. Brief. Bioinform. *25*, bbad476. https://doi.org/10.1093/bib/bbad476.

20. Basith, S., Pham, N.T., Song, M., Lee, G., and Manavalan, B. (2023). ADP-Fuse: A novel two-layer machine learning predictor to identify antidiabetic peptides and diabetes types using multiview information. Comput. Biol. Med. *165*, 107386. https://doi.org/10.1016/j.compbiomed.2023.107386.

21. Ji, Y., Zhou, Z., Liu, H., and Davuluri, R.V. (2021). DNABERT: pre-trained Bidirectional Encoder Representations from Transformers model for DNA-language in genome. Bioinformatics *37*, 2112–2120. https://doi.org/10.1093/bioinformatics/btab083.

22. Akiyama, M., and Sakakibara, Y. (2022). Informative RNA base embedding for RNA structural alignment and clustering by deep representation learning. NAR Genom. Bioinform. *4*, lqac012. https://doi.org/10.1093/nargab/lqac012.

23. Huang, Y., Niu, B., Gao, Y., Fu, L., and Li, W. (2010). CD-HIT Suite: a web server for clustering and comparing biological sequences. Bioinformatics *26*, 680–682. https://doi.org/10.1093/bioinformatics/btq003.

24. Basith, S., Manavalan, B., Hwan Shin, T., and Lee, G. (2020). Machine intelligence in peptide therapeutics: A next-generation tool for rapid disease screening. Med. Res. Rev. *40*, 1276–1314. https://doi.org/10.1002/med.21658.

25. Liu, X.W., Shi, T.Y., Gao, D., Ma, C.Y., Lin, H., Yan, D., and Deng, K.J. (2023). iPADD: A Computational Tool for Predicting Potential Antidiabetic Drugs Using Machine Learning Algorithms. J. Chem. Inf. Model. *63*, 4960–4969. https://doi.org/10.1021/acs.jcim.3c00564.

26. Yang, Y.H., Ma, C.Y., Gao, D., Liu, X.W., Yuan, S.S., and Ding, H. (2023). i2OM: Toward a better prediction of 2'-O-methylation in human RNA. Int. J. Biol. Macromol. *239*, 124247. https://doi.org/10.1016/j.ijbiomac.2023.124247.

27. Lv, H., Zhang, Y., Wang, J.S., Yuan, S.S., Sun, Z.J., Dao, F.Y., Guan, Z.X., Lin, H., and Deng, K.J. (2022). iRice-MS: An integrated XGBoost model for detecting multitype post-translational modification sites in rice. Brief. Bioinform. *23*, bbab486. https://doi.org/10.1093/bib/bbab486.

28. Liu, B., Gao, X., and Zhang, H. (2019). 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. *47*, e127. https://doi.org/10.1093/nar/gkz740.

29. Lalović, D., and Veljković, V. (1990). The global average DNA base composition of coding regions may be determined by the electron-ion interaction potential. Biosystems *23*, 311–316. https://doi.org/10.1016/0303-2647(90)90013-q.

30. Nair, A.S., and Sreenadhan, S.P. (2006). A coding measure scheme employing electron-ion interaction pseudopotential (EIIP). Bioinformation *1*, 197–202.

31. Gao, F., and Zhang, C.-T. (2004). Comparison of various algorithms for recognizing short coding sequences of human genes. Bioinformatics *20*, 673–681. https://doi.org/10.1093/bioinformatics/btg467.

32. Gupta, S., Dennis, J., Thurman, R.E., Kingston, R., Stamatoyannopoulos, J.A., and Noble, W.S. (2008). Predicting human nucleosome occupancy from primary sequence. PLoS Comput. Biol. *4*, e1000134. https://doi.org/10.1371/journal.pcbi.1000134.

33. Noble, W.S., Kuehn, S., Thurman, R., Yu, M., and Stamatoyannopoulos, J. (2005). Predicting the in vivo signature of human gene regulatory sequences. Bioinformatics *21* (*Suppl 1*), i338–i343. https://doi.org/10.1093/bioinformatics/bti1047.

34. Chen, Z., Liu, X., Zhao, P., Li, C., Wang, Y., Li, F., Akutsu, T., Bain, C., Gasser, R.B., Li, J., et al. (2022). iFeatureOmega: an integrative platform for engineering, visualization and analysis of features from molecular sequences, structural and ligand data sets. Nucleic Acids Res. *50*, W434–W447. https://doi.org/10.1093/nar/gkac351.

35. Wei, L., Su, R., Luan, S., Liao, Z., Manavalan, B., Zou, Q., and Shi, X. (2019). Iterative feature representations improve N4-methylcytosine site prediction. Bioinformatics *35*, 4930–4937. https://doi.org/10.1093/bioinformatics/btz408.

36. Mikolov, T., Chen, K., Corrado, G., and Dean, J. (2013). Efficient Estimation of Word Representations in Vector Space. Preprint at arXiv *1*. https://doi.org/10.48550/arXiv.1301.3781.

37. Heinzinger, M., Elnaggar, A., Wang, Y., Dallago, C., Nechaev, D., Matthes, F., and Rost, B. (2019). Modeling aspects of the language of life through transfer-learning protein sequences. BMC Bioinf. *20*, 723–817. https://doi.org/10.1186/s12859-019-3220-8.

38. Bupi, N., Sangaraju, V.K., Phan, L.T., Lal, A., Vo, T.T.B., Ho, P.T., Qureshi, M.A., Tabassum, M., Lee, S., and Manavalan, B. (2023). An Effective Integrated Machine Learning Framework for Identifying Severity of Tomato Yellow Leaf Curl Virus and Their Experimental Validation. Research (Wash D C) *6*, 0016. https://doi.org/10.34133/research.0016.

39. Charoenkwan, P., Schaduangrat, N., Pham, N.T., Manavalan, B., and Shoombuatong, W. (2023). Pretoria: An effective computational approach for accurate and high-throughput identification of CD8(+) t-cell epitopes of eukaryotic pathogens. Int. J. Biol. Macromol. *238*, 124228. https://doi.org/10.1016/j.ijbiomac.2023.124228.

40. Hasan, M.M., Tsukiyama, S., Cho, J.Y., Kurata, H., Alam, M.A., Liu, X., Manavalan, B., and Deng, H.W. (2022). Deepm5C: A deep-learning-based hybrid framework for identifying human RNA N5-methylcytosine sites using a stacking strategy. Mol. Ther. *30*, 2856–2867. https://doi.org/10.1016/j.ymthe.2022.05.001.

41. Malik, A., Subramaniyam, S., Kim, C.B., and Manavalan, B. (2022). SortPred: The first machine learning based predictor to identify bacterial sortases and their classes using sequence-derived information. Comput. Struct. Biotechnol. J. *20*, 165–174. https://doi.org/10.1016/j.csbj.2021.12.014.

42. Dao, F.Y., Lv, H., Su, W., Sun, Z.J., Huang, Q.L., and Lin, H. (2021). iDHS-Deep: an integrated tool for predicting DNase I hypersensitive sites by deep neural network. Brief. Bioinform. *22*, bbab047. https://doi.org/10.1093/bib/bbab047.