*Bioinformatics*, 38(1), 2022, 52–57
doi: 10.1093/bioinformatics/btab611
Advance Access Publication Date: 24 August 2021
Original Paper





Sequence analysis

# DeepAc4C: a convolutional neural network model with hybrid features composed of physicochemical patterns and distributed representation information for identification of N4-acetylcytidine in mRNA

**Chao Wang [1], Ying Ju[2], Quan Zou [1,3,*] and Chen Lin[2,*]**

[1]Institute of Fundamental and Frontier Sciences, University of Electronic Science and Technology of China, Chengdu 610054, China, [2]School of Informatics, Xiamen University, Xiamen 361005, China and [3]Yangtze Delta Region Institute (Quzhou), University of Electronic Science and Technology of China, Quzhou 324000, China

*To whom correspondence should be addressed.

Associate Editor: Anthony Mathelier

Received on May 13, 2021; revised on August 17, 2021; editorial decision on August 18, 2021; accepted on August 20, 2021

## Abstract

**Motivation:** N4-acetylcytidine (ac4C) is the only acetylation modification that has been characterized in eukaryotic RNA, and is correlated with various human diseases. Laboratory identification of ac4C is complicated by factors, such as sample hydrolysis and high cost. Unfortunately, existing computational methods to identify ac4C do not achieve satisfactory performance.

**Results:** We developed a novel tool, DeepAc4C, which identifies ac4C using convolutional neural networks (CNNs) using hybrid features composed of physicochemical patterns and a distributed representation of nucleic acids. Our results show that the proposed model achieved better and more balanced performance than existing predictors. Furthermore, we evaluated the effect that specific features had on the model predictions and their interaction effects. Several interesting sequence motifs specific to ac4C were identified.

**Availability and implementation:** The webserver is freely accessible at https://ac4c.webmalab.cn/, the source code and datasets are accessible at Zenodo with URL https://doi.org/10.5281/zenodo.5138047 and Github with URL https://github.com/wangchao-malab/DeepAc4C.

**Contact:** zouquan@nclab.net or chenlin@xmu.edu.cn

**Supplementary information:** Supplementary data are available at *Bioinformatics* online.

## 1 Introduction

To date, more than 170 modified nucleosides in RNA have been identified (Boccaletto *et al.*, 2018). These post-transcriptional RNA modifications affect intramolecular interactions, structure, and interactions with other molecules. These subtle structural changes introduce functional RNA diversity by regulating translational efficiency, mRNA stability, and RNA-protein interactions, which are all fundamentally important for cell growth and development (Boccaletto *et al.*, 2018; Li, 2020; Thomas *et al.*, 2019). N4-acetylcytidine (ac4C) occurs at cytidine residues in RNA, and this nucleoside substitution has been described in all domains of life (Thomas *et al.*, 2018). Further, ac4C is the only acetylation modification that has been characterized in eukaryotic RNA (Jin *et al.*, 2020). Ac4C is correlated with various human diseases, including inflammation, metabolic diseases, autoimmune diseases, and cancer (Jin *et al.*, 2020).

Ac4c was initially detected in yeast and mammalian transfer RNA (tRNA) (Staehelin *et al.*, 1968; Zachau *et al.*, 1966), followed by observation in bacterial tRNA (Oashi *et al.*, 1972) and eukaryotic ribosomal RNA (Thomas *et al.*, 1978). Recently, Arango *et al.* (2018) revealed that mRNA acetylation, which is catalyzed by N-acetyltransferase 10 (NAT10), is present in the human transcriptome and is especially enriched within coding regions. Further investigation revealed that ac4C in mRNA promotes mRNA stability and translation. The presence of ac4C in mRNA was also observed in yeast, and the modification levels change in response to oxidative stress (Tardu *et al.*, 2019).

Identification of ac4C modification sites is an area of great interest for biological and computational research. In early studies, ac4C identification was performed by experimental methods including high-performance liquid chromatography (HPLC), HPLC-mass

spectrometry (MS), borohydride-based sequencing, and antibody-based methods (refer to Jin *et al.*, 2020 for more details). Among these methods, anti-ac4c antibody (acetylated RNA immunoprecipitation or acRIP)-based high-throughput sequencing, acRIP-seq, revealed the first transcriptome-wide profile of ac4C locations in human mRNA (Arango *et al.*, 2018). However, these experimental approaches have several shortcomings, such as sample hydrolysis, large reagent requirements, poor sensitivity, and high cost (Jin *et al.*, 2020). Consequently, there is an urgent need to develop computational approaches for ac4C identification.

In the last few years, two bioinformatic tools were developed for identifying ac4C sites in mRNA. The first predictor, PACES, was proposed by Zhao *et al.* (2019), in which position-specific dinucleotide sequence profiles and K-nucleotide frequencies were used as features to train a random forest (RF) model. More recently, Alam *et al.* (2020) developed a new predictor called XG-ac4C, which was trained on six feature types (nucleotide chemical properties, nucleotide density, Kmer, one-hot encoding, electron–ion interaction pseudopotentials, and electron–ion interaction pseudopotentials of trinucleotides) using eXtreme Gradient Boosting (XGboost) classifiers. XG-ac4C successfully improved the predictive performance and the area under the precision-recall curve (PRC) by 9.6% compared to PACES in independent tests (Alam *et al.*, 2020). However, the prediction performances of the two predictors are still not ideal. A more accurate computational method for ac4C modification site identification is desperately needed.

The purpose of this study is to establish an advanced prediction model, DeepAc4C, to further improve the performance of ac4C site identification in human mRNA. To uncover sequence patterns in multiple views, we combined the physicochemical features with distributed nucleic acid representative information. Then, one-dimensional convolutional neural networks (CNNs) were applied to integrate information of the features and to perform classification. We demonstrate that training of one-dimensional CNNs with hybrid features that fuse physicochemical patterns and semantic information can cause them to outperform the existing ac4C site predictors.

## 2 Materials and methods

Figure 1 illustrates the workflow of constructing the DeepAc4C model, which involves three main steps: (i) sequence preprocessing, (ii) sequence encoding and feature dimensionality reduction, and (iii) feature combination, neural model training, and evaluation. More details regarding each step are described below.

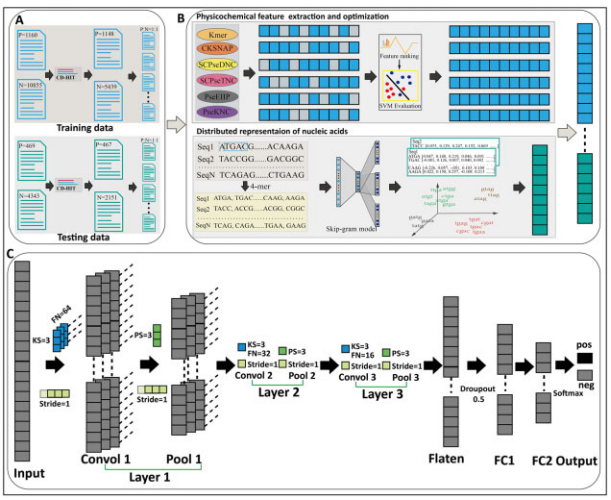


**Fig. 1.** The DeepAc4C workflow. (**A**) Data collection, sequence redundancy elimination, and dataset balancing. (**B**) Physicochemical feature encoding, feature optimization, and distributed representation of nucleic acids. (**C**) Neural network architectures of DeepAc4C. KS, kernel size; PS, pooling size; FN, filter number; FC, fully connected layer

### 2.1 Data collection and preprocessing

In this study, we aimed to identify ac4c sites in human mRNA. The datasets were obtained from PACES (http://www.rnanut.net/paces/) (Zhao *et al.*, 2019). These datasets were also used in the construction of XG-ac4C (Alam *et al.*, 2020). The datasets were constructed using 2134 genes with positive and negative ac4C sites that were experimentally validated by high-throughput acRIP-seq (Arango *et al.*, 2018). The training dataset was composed of 1160 positive samples and 10 855 negative samples. The independent testing dataset was composed of 469 positive samples and 4343 negative samples. Because redundant sequences in the training dataset may lack sample representativeness, PACES and XG-ac4C predictions might be overfitted. In our work, CD-HIT software (Fu *et al.*, 2012) was used to eliminate sample redundancy with a threshold of 0.4. Finally, we obtained 1148 positive samples and 5439 negative samples for the training dataset, and 467 positive samples, and 2151 negative samples for the independent test dataset (Fig. 1A).

PACES and XG-ac4C were evaluated on redundancy and seriously imbalanced data (the ratio of positive and negative samples is about 1:9) based on only two metrics, namely operating characteristic curve (ROC) and PRC. To provide a comprehensive evaluation of the predictive ability in model training, evaluation, and testing, 10 balanced datasets were generated for both training and testing (Fig. 1A). Thus, each training and a testing subset was composed of 1148 and 467 positive samples, respectively, and an equal number of negative samples were randomly chosen from the corresponding negative dataset. The ten balanced training datasets were hereafter referred to as TD1, TD2, . . .. . . TD10.

### 2.2 Physicochemical feature extraction

Extracting strong discriminative features is crucial for building a reliable and superior model. In this study, six physicochemical feature encoding strategies were employed to formulate the DNA fragments (Fig. 1B). These features are Kmer, the composition of k-spaced nucleic acid pairs (CKSNAP), series correlation pseudo dinucleotide composition (SCPseDNC), series correlation pseudo trinucleotide composition (SCPseTNC), pseudo *k*-tupler composition (PseKNC), and electron–ion interaction pseudopotentials of trinucleotides (PseEIIP) (Chen *et al.*, 2020). The above six feature encoding algorithms are described in detail in the Supplementary Methods. A brief introduction of these descriptors is as follows:

Kmer is the most direct approach to represent the DNA sequences, which are defined as the occurrence frequencies of *k*-neighboring nucleic acids (Liu *et al.*, 2015, 2019). CKSNAP calculates the frequency of nucleic acid pairs separated by an arbitrary number (*k*) of nucleic acids (Chen *et al.*, 2020). PseEIIP computes the energy of delocalized electrons of trinucleotides based on the normalized trinucleotide frequency and the electron energy of the four nucleotides (A: 0.1260, C: 0.1340, G: 0.0806, and T: 0.1335) (Nair and Sreenadhan, 2006). The remaining three encodings were used to mine sequence order information and effects (Liu *et al.*, 2015). PseKNC characterizes the contiguous local sequence-order information and the global sequence-order information of a sequence based on the trinucleotide occurrence frequency and six physicochemical indices (rise, roll, shift, slide, tilt, and twist) (Guo *et al.*, 2014). SCPseDNC and SCPseTNC are similar to PseKNC. SCPseDNC considers the series correlations of dinucleotide physicochemical properties, SCPseTNC considers the series correlations of trinucleotide physicochemical properties (Chen *et al.*, 2014).

### 2.3 Distributed representation of nucleic acids

Recently, a word embedding algorithm (Mikolov *et al.*, 2013) was proposed for efficient distributed representation of biological sequences, including proteins (Asgari and Mofrad, 2015), DNA (Khan, 2019), mRNA (Zou *et al.*, 2019), non-coding RNA (Aoki and Sakakibara, 2018; Chaabane *et al.*, 2020), and 16S/18S rRNA (Wang *et al.*, 2020b ; Woloszynek *et al.*, 2019). In this work, the nucleic sequences were embedded using the word2vec framework (Church, 2017; Mikolov *et al.*, 2013), in which each word is characterized by its context, i.e. neighboring words and similar words have

close vectors. The method is briefly described as follows (Fig. 1B): First, nucleic acid sequences of k length were regarded as a sentence. Then, the bio-corpus was generated in an overlapping manner by moving a window of size $k$ along a sequence with a stride length of 1. Given this bio-corpus, the next step was to embed each word into a fixed $N$-dimensional numeric vector using word2vec with a skip-gram model, which predicts the surrounding words from the current word. Thus, each word was presented as a numeric vector of size $N$, and each sequence was represented by the average of all corpus in the sequence, which is a vector of size $N$.

### 2.4 Feature optimization and dimension reduction

Feature dimensions of five of the six physicochemical feature descriptors depend on the related parameters. To maximize the effectiveness of each individual descriptor, the five parameters were optimized. The search range for each parameter is listed in Supplementary Table S1. The support vector machine (SVM) algorithm was employed for model training and evaluation based on 10-fold cross-validation. We implemented SVM with the Python package in scikit-learn (v 0.22.1). The search range of the two critical parameters C and $\gamma$ was [0.01, 0.05, 0.1, 0, 1, 5, ..., 90, 95, 100] and [0.0001, 0.0002, 0.0004, 0.0006, 0.0008, ..., 2, 4, 6, 8], respectively. The radial basis function (RBF) was used as the kernel function. However, a direct combination of all six feature types may cause high dimensionality and information redundancy, which further influences model performance and increases computing complexity and time. Thus, a two-step feature optimization method including feature importance ranking based on F-score and a sequential forward search (SFS) based on the accuracy (ACC) was applied to choose the optimal feature subsets (Fig. 1B). The detailed procedure was previously described by Wang *et al.* (2020a,b).

### 2.5 Neural network architectures

The six optimized features and the embedded features were concatenated into a vector and then fed into a 1D CNN algorithm for training and testing (Lv *et al.*, 2020). We used tensorflow 2.4 to implement the CNN model. The main architecture of the CNN model consisted of three convolution layers, three pooling layers, and three fully connected layers (Fig. 1C). The combined features served as the input layer. Then, the convolution operation was used to extract potential feature patterns. The kernel size (KS) was set as three and the stride size was set as one, which means that three adjacent features in a kernel window are used as input for a neuron and the kernel window moves along the input vector with a step size of one. To summarize the convolution output of three adjacent kernels, each convolution layer is followed by a max-pooling layer, which extracts the maximum value in the pooling window with a size of three. The output of the last max-pooling layer was flattened and a dropout layer was generated. Next, three fully connected layers were applied to connect the dropout layer. A softmax function was used for binary classification in the final output later. A ReLU function was used in all convolution layers and the first two fully connected layers. During learning, six hyperparameters (the number of kernels for each convolution layer, the number of units in the fully connected layer, and the learning rates) were optimized. A Keras tuner library was used for automatically turning the hyperparameters, as listed in Supplementary Table S2.

### 2.6 Model training and evaluation

Each training dataset was further divided into a sub-training set (90% of the training dataset) and a validation set (10% of the training dataset). Thus, the number of samples for model training, validation, and testing were 2066, 230, and 934, respectively. The sub-training set was used to fit the model with optimal parameters as listed in Supplementary Table S2. The validation set was used to validate the performance of the model with the most suitable parameters, and the testing data was used to provide an unbiased performance evaluation of the final model.

Five metrics (Wei *et al.*, 2014, 2017, 2019) were used to comprehensively measure the performance of the ensemble model, which are ACC, specificity (SP), sensitivity (SN), Matthews correlation coefficient (MCC), and AUC. Each metric was calculated as follows:

$$\mathrm{ACC} = \frac{\mathrm{TP} + \mathrm{TN}}{\mathrm{TP} + \mathrm{TN} + \mathrm{FP} + \mathrm{FN}} \tag{1}$$

$$\mathrm{SN} = \frac{\mathrm{TP}}{\mathrm{TP} + \mathrm{FN}} \tag{2}$$

$$\mathrm{SP} = \frac{\mathrm{TN}}{\mathrm{TN} + \mathrm{FP}} \tag{3}$$

$$\mathrm{MCC} = \frac{\mathrm{TP} \times \mathrm{TN} - \mathrm{FP} \times \mathrm{FN}}{\sqrt{(\mathrm{FP} + \mathrm{TP})(\mathrm{FN} + \mathrm{TP})(\mathrm{FP} + \mathrm{TN})(\mathrm{FN} + \mathrm{TN})}} \tag{4}$$

The metric AUC calculates the area under the receiver operating characteristic curve based on the false positive rate (FPR) and the true positive rate (TPR) under various thresholds. The TPR and the FPR were calculated as follows:

$$\mathrm{TPR} = \frac{\mathrm{TP}}{\mathrm{TP} + \mathrm{FN}} \tag{5}$$

$$\mathrm{FPR} = \frac{\mathrm{FP}}{\mathrm{TN} + \mathrm{FP}} \tag{6}$$

where TP = true positive, FP = false positive, TN = true negative, and FN = false negative. SN and SP were employed to evaluate the model performance with respect to the positive and negative samples, respectively. The remaining three metrics are global prediction performance indicators.

## 3 Results

### 3.1 Descriptor parameter optimization and feature selection

The feature vector dimensions of five of the six physicochemical descriptors are determined by the algorithm parameters. To make each of the descriptors as informative as possible, these parameters were optimized before they were used for feature selection. The parameter search range and the accuracy are listed in Supplementary Table S1. To reduce computing complexity and enhance model performance, F-scores and SFS were used for feature selection. The parameter optimization and feature selection were processed on TD1. Then, we applied the optimal parameters to the other nine datasets for computational convenience. The feature selection results are illustrated in Figure 2 (A–F). The dimensions for the six encodings were reduced, especially for features with higher dimensionality, such as Kmer (Fig. 2B), PseKNC (Fig. 2D), and SCPseTNC (Fig. 2F), because high dimensional features tend to increase information redundancy. Further, the performance of the models trained using the optimal features was improved in terms of the AUC metric (Fig. 2A–F), suggesting that the feature selection step is beneficial for feature representation.

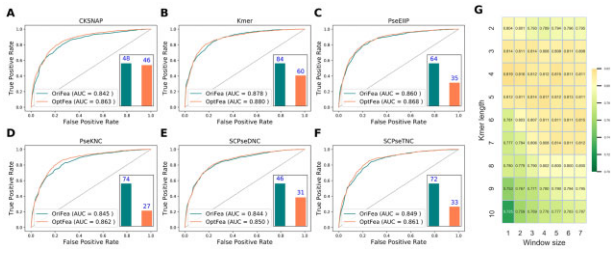


**Fig. 2.** (**A–F**) ROC curves of six feature descriptors with different algorithm parameters. OriFea, original features; OptFea, optimal features. (**G**) The heatmap shows the accuracy values of the model constructed with different k (length of k-mer) and $w$ (window size) values

### 3.2 Distributed representation of nucleic acids

All training nucleic acids were divided into a k-mer corpus, and each k-mer was embedded into a 100-dimensional vector using word2vec with a skip-gram model. In this process, the length of the sliding window (length $k$ of the nucleic acid sequence k-mer) and the number of surrounding words (window size $w$ from word2vec) were two critical parameters that needed to be optimized. To determine the optimal $k$ *and* $w$ values, we varied these two parameters to generate 63 models. The $k$ value was varied from 2 to 10 and the $w$ value was varied from 1 to 7. The ACC values for all possible combinations are depicted in Figure 2 (G). We observed that the ACC values gradually increased when the $k$ value increased from 2 to 4. Then, the ACC values gradually decreased when $k$ from 5 to 10. The $w$ value was inversely correlated with the ACC. Taken together, the heatmap region with the maximum ACC value is based on a k-mer length of 4 and a window size of 1, which were used for the final skip-gram model.

### 3.3 DeepAc4C neural model

In DeepAc4C, we combined the sequence physicochemical features (232D) and the embedded semantic features (100D) as inputs for the neural network. The combined 332D features were processed using the architecture of our deep learning model (Fig. 1C). The CNN model was fitted on the training dataset using an early stopping strategy that was based on the validation loss to avoid overfitting. Then, the model was validated. Test data were used to test the model that showed the best performance with the validation data. Table 1 shows the ACC values for the training and validation sets and the ACC, MCC, and AUC values for the testing set. For the 10 balanced training datasets, the maximum ACC was achieved on TD5 (0.8490) and the minimum ACC was obtained on TD6 (0.7996). The variances of independent test values are relatively small, which suggests the models were stable. For the sake of convenience and comparison, the average values were used to measure the model performance. DeepAc4C achieved an average training ACC of 0.8242, an average validation ACC of 0.8139, and an average ACC of 0.7919, MCC of 0.5857, and AUC of 0.8649 using independent test data.

As described above, each training subset represents only part of the information from the complete training dataset. Therefore, an ensemble model (soft voting, threshold = 0.5) was built to integrate all individual neural models. The integrated model was evaluated using the independent test data. The ensemble model achieved better performance than the individual models (Table 1), indicating that the ensemble strategy improves model performance.

To further evaluate the effectiveness of the neural network architecture, we compared DeepAc4c with nine popular conventional machine learning algorithms, including the adaboost classifier (ADAB), bagging (BAG), decision tree (DT), $k$-nearest neighbor (KNN), light gradient boosting machine (LGB), logistic regression (LR), naive bayesian (NB), random forest (RF), and support vector machine (SVM) algorithms. For a fair comparison, the model was trained using the balanced training dataset and evaluated using the independent test dataset. Figure 3 (A) shows the average values of five metrics (refer to Supplementary Table S3 for more details). DeepAc4c achieved the best scores for all the five metrics, indicating that the proposed model is significantly superior for ac4C identification compared to traditional classifiers.

**Table 1.** Performance of ten models trained on balanced datasets

| Model | Training ACC | Validation ACC | Test ACC | Test MCC | Test AUC |
|---|---|---|---|---|---|
| TD1 | 0.8093 | 0.8043 | 0.7934 | 0.5871 | 0.8620 |
| TD2 | 0.8195 | 0.8000 | 0.7943 | 0.5921 | 0.8660 |
| TD3 | 0.8209 | 0.8043 | 0.7874 | 0.5777 | 0.8641 |
| TD4 | 0.8490 | 0.8348 | 0.7969 | 0.5945 | 0.8658 |
| TD5 | 0.8403 | 0.8043 | 0.7950 | 0.5902 | 0.8646 |
| TD6 | 0.7996 | 0.7913 | 0.7938 | 0.5897 | 0.8645 |
| TD7 | 0.8209 | 0.8609 | 0.7911 | 0.5836 | 0.8671 |
| TD8 | 0.8475 | 0.8043 | 0.7978 | 0.5959 | 0.8657 |
| TD9 | 0.8078 | 0.8130 | 0.7846 | 0.5703 | 0.8615 |
| TD10 | 0.8277 | 0.8217 | 0.7850 | 0.5725 | 0.8680 |
| Average[a] | 0.8242 | 0.8139 | 0.7919 | 0.5857 | 0.8649 |
| Ensemble[b] | — | — | 0.7979 | 0.5965 | 0.8734 |

[a]Average: metrics average value for TD1 to TD10.
[b]Ensemble: metrics value for the model ensemble.

### 3.4 Feature contribution and dependency analysis

SHapley Additive exPlanation (SHAP) values (Lundberg *et al.*, 2017) were used to explain the prediction model. SHAP utilizes a game theory approach that assigns payouts to players based on their contribution to the total payout (Shapley, 1953). The SHAP method has several advantages, such as model agnosticity, local accuracy, missingness, and consistency (Moncada-Torres *et al.*, 2021). First, a SHAP summary plot was used to calculate the top 20 most important features. As depicted in Figure 4 (A), nine features were generated by the six physicochemical descriptors. The remaining 11 features are embedded features, which imply that both feature types are crucial for model construction. Sequence motif/patterns that are strongly related to ac4C recognition are composed of guanosine nucleotides (G), such as GG.gap2 (CKSNAP), GG.gap1 (CKSNAP), GG (Kmer), and GGG (Kmer). Moreover, thymine nucleotides (T) in T (Kmer) and cytosine nucleotides (C) in TC.gap1 (CKSNAP) are important for ac4C identification. As the function of cytidine acetylation has yet to be fully elucidated, the biological function of the above sequence patterns still lacks a rational explanation. Of note, the 'CXX' motif was enriched within the ac4C peaks detected by the acRIP-seq method (Arango *et al.*, 2018), and the recently ac$^4$C-seq method (Sas-Chen *et al.*, 2020) (a chemical genomic method for identification of ac4C site at single-nucleotide resolution) showed that 98% novel detected ac4C sites occurred at a 'CCG' motif. We speculated that the differences were due to sequence context length, where motif enrich performed on the 10–20 nt around the ac4C center, while SHAPE values were calculated based on the whole 415 nt sequence. These further indicated that implicit features in ac4C context can be extracted by DeepAc4C.

We also analyzed how the values of the top 20 most important features affect the model prediction. Figure 4 (B) shows the corresponding summary plots that illustrate how high and low feature values were related to model output. Each point represents an instance of the dataset (i.e. a sequence sample). The SHAP value on the $x$-axis corresponds to the impact that each instance had on the model prediction for that specific sample. High GG.gap2 (CKSNAP) values were associated with positive impacts on ac4C identification, while low values indicated negative impacts. Similar trends are observed in the other 11 of the top 20 features. Other trends are presented by the other eight features, i.e. high values of T (Kmer) decrease the model behavior, while low T (Kmer) values enhance model performance.

The SHAP dependence plots were used to further explain how a single feature affects the model output. The dependence plots of the top 20 features are shown in Figure 4 (C–J) and Supplementary Figure S1. These plots reveal a few interesting observations. First, the SHAP values of the top four features (Fig. 4C–F), GG.gap2 (CKSNAAP), GGG (PseEIIP), GG (Kmer), and GGG (Kmer), have a larger range than others, which suggests why these features dominated the behavior of the model. The small range observed for EmbedFea35 and EmbedFea89 explains why these features are less

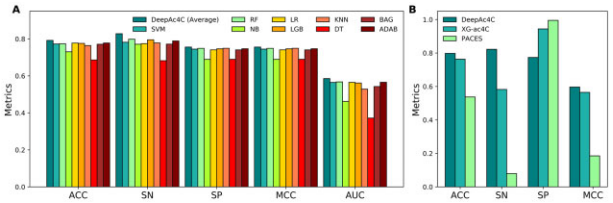


**Fig. 3.** Performance comparison of DeepAc4C and nine conventional machine learning methods (**A**) and predictors PACES and XG-ac4C (**B**)

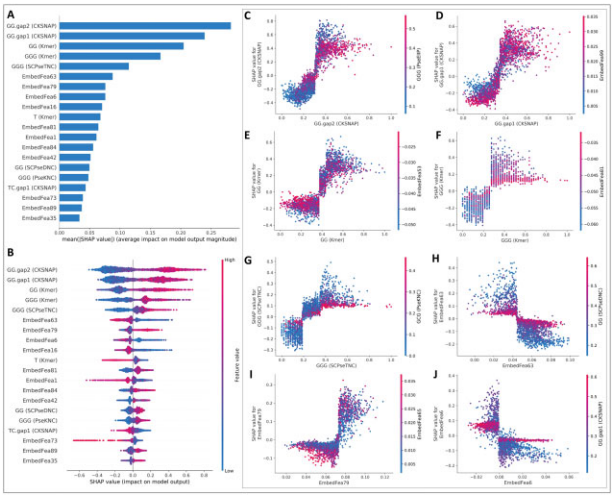

**Fig. 4.** Feature contribution and dependency analysis. (**A**) The 20 most important features. (**B**) Summary plot for SHAP values. For each feature, one point corresponds to a single sample. The SHAP value along the *x*-axis represents the impact that feature had on the model's output for that specific sample. Features in the higher position in the plot mean the more important it is for the model. (**C–J**) The SHAP dependence plots. These plots show the effect that a single feature has on the model's predictions and the interaction effects across features. Each point corresponds to an individual sample, the value along the *x*-axis corresponds to the feature value, the color represents the value of the interacting feature

important than others. Changes in their values have less influence on the corresponding SHAP values (Supplementary Fig. S1K and L). Second, the feature dependence plots can identify important feature turning points. For example, the proposed model using 0.3 as a GG.gap2 (CKSNAP) turning point showed that feature values >0.3 contribute to model prediction (Fig. 4C). Figure 4 (G) shows that GGG (SCPseDNC) has two-segmented points, where feature value = 0.2 is a point that changes the SHAP values from negative to positive, while 0.4 pushes the SHAP values to a higher range. These results indicate that the model can capture non-linear relations existing in the data.

Lastly, the SHAP dependence plots also provide meaningful insights into interaction effects across features. For example, Figure 4 (C) shows that low GG.gap2 values (range 0.1 to −0.3) with low GGG (PseEIIP) values (0.1–0.2) have an adverse impact on model behavior, while moderate GG.gap2 value (0.2–0.6) and high GGG values (PseEIIP) are favorable for ac4C identification. Similar feature interaction patterns were observed in two other feature pairs (Fig. 4F and G). In contrast, Figure 5 (H) and (J) shows that low EmbedFea63 with high GG (SCPseDNC) values contribute to accurate model prediction, while high EmbedFea63 values have the opposite effect. More feature interaction patterns can be seen in Figure 4 (C–J) and Supplementary Figure S1.

### 3.5 Comparison with existing predictors

PACES and XG-ac4C are two bioinformatics tools for identifying ac4c sites in mRNA. Here, we compared DeepAc4C with XG-ac4C (Alam *et al.*, 2020) and PACES (Zhao *et al.*, 2019). The test dataset used for PACES and XG-ac4C is redundant and seriously imbalanced. Further, only two metrics (AUC and PRC) are used for performance evaluation. To provide a comprehensive evaluation of the predictive ability of our model, the balanced and homology-reduced (CD-HIT threshold of 0.4) test datasets were used for model testing and comparison.

Figure 3 (B) provides details of the comparative analysis results. DeepAc4C exhibited the best performances, followed by XG-ac4C, while PACES ranked last. DeepAc4C outperformed XG-ac4C in the ACC, SN, and MCC metrics, with improvements of 3.47, 23.98, and 3.19%, respectively. For SP, the XG-ac4C returned higher values than our model. However, it is worth noting that the DeepAc4C achieved more balanced performance with |SN-SP| = 4.88%, while XG-ac4C returned |SN-SP| = 36.15%. As indicated in Equations (2) and (3), SN and SP describe the true positive rate and true negative rate, respectively. These two metrics measure a predictor from two different angles and actually constrain with each other (Chou, 1993; Liu *et al.*, 2018). It is important to guarantee a balance between SN and SP for an accurate model. DeepAc4C achieved an SN of 82.22% and SP of 77.34%, while XG-ac4C resulted in an SN of 58.24% and SP of 94.39%, suggesting the XG-ac4C tends to predict a query sequence as non-Ac4c sites. PACES resulted from a more serious biased result with |SN-SP| = 91.59%, indicating this predictor can hardly predict the true positive samples.

Recently, Sas-Chen *et al.* (2020) reported quantitative, nucleotide-resolution profiling of ac4C in archaea *Thermococcus kodakarensis*, where 119 ac4C sites were identified from mRNA, and 97 of them were successfully extracted from the reference genome ASM996v1 (Supplementary Text S1). The 97 sequences were further used for the model's performance comparison to check their cross-species prediction ability. The results are listed in Supplementary Table S4. DeepAc4C achieved the best performance and predicted 29 out of 97 positive samples, XG-ac4C achieved the second-best performance and correctly predicted 21 positive samples, and none positive samples were correctly predicted by PACES. It should be noted that such testing is not very appropriate because the sequence patterns between our training datasets and the 97 positive samples are different. The training datasets containing consecutive 'CXX' motif, while such patterns were absent from the 97 positive samples. Therefore, the above three models achieved relatively low accuracy on the 97 positive samples. In conclusion, these results demonstrate that DeepAc4C is significantly better than the existing prediction algorithms for Ac4C identification.

It should be noted that the datasets used for model training and testing are generated by the immunoprecipitation-based approach, where the possibility of antibody promiscuity cannot be ruled out. Furthermore, sequences without consecutive 'CXX' motifs were absent from the current training datasets. Consequently, the prediction performances of the above three models are partly biased to specific sequence patterns. Therefore, making full use of the novel high base-resolution data (Sas-Chen *et al.*, 2020) will conductive to control the false discovery rate.

## 4 Conclusion

In this study, a deep learning predictor called DeepAc4C was developed to accurately identify ac4C modifications in human mRNA. The hybrid features composed of physicochemical patterns and semantic information were used for sequence pattern representation. As a result, our model outperforms the existing predictors for ac4C site prediction and achieves a more balanced performance. Furthermore, we used SHAP values to investigate the impact of specific features on the model predictions and their interaction effects. For convenience, a user-friendly web server that implements DeepAc4C has been made available to the public. We expect that DeepAc4C will be a useful tool that can be complementary to hands-on experiments for the computational identification of ac4C sites. Together, these approaches will facilitate our functional understanding of ac4C in RNA.

## Acknowledgements

The authors thank anonymous reviewers for their kind suggestions and comments for improving this article.

## Funding

The work was supported by the National Natural Science Foundation of China (Nos. 61922020, 61771331, 62002051, and 62072385) and the Special Science Foundation of Quzhou (2020D003).

*Conflict of Interest*: none declared.

## References


Alam,W. *et al.* (2020) XG-ac4C: identification of N4-acetylcytidine (ac4C) in mRNA using eXtreme gradient boosting with electron-ion interaction pseudopotentials. *Sci. Rep*., **10**, 20942.

Aoki,G. and Sakakibara,Y. (2018) Convolutional neural networks for classification of alignments of non-coding RNA sequences. *Bioinformatics*, **34**, i237–i244.

Arango,D. *et al.* (2018) Acetylation of cytidine in mRNA promotes translation efficiency. *Cell*, **175**, 1872–1886.

Asgari,E. and Mofrad,M.R.K. (2015) Continuous distributed representation of biological sequences for deep proteomics and genomics. *PLoS One*, **10**, e0141287.

Boccaletto,P. *et al.* (2018) MODOMICS: a database of RNA modification pathways. 2017 update. *Nucleic Acids Res*., **46**, D303–D307.

Chaabane,M. *et al.* (2020) circDeep: deep learning approach for circular RNA classification from other long non-coding RNA. *Bioinformatics*, **36**, 73–80.

Chen,W. *et al.* (2014) PseKNC: a flexible web server for generating pseudo K-tuple nucleotide composition. *Anal. Biochem*., **456**, 53–60.

Chen,Z. *et al.* (2020) iLearn: an integrated platform and meta-learner for feature engineering, machine-learning analysis and modeling of DNA, RNA and protein sequence data. *Brief. Bioinform*., **21**, 1047–1057.

Chou,K.C. (1993) A vectorized sequence-coupling model for predicting HIV protease cleavage sites in proteins. *J. Biol. Chem*., **268**, 16938–16948.

Church,K.W. (2017) Emerging trends Word2Vec. *Nat. Lang. Eng*., **23**, 155–162.

Fu,L.M. *et al.* (2012) CD-HIT: accelerated for clustering the next-generation sequencing data. *Bioinformatics*, **28**, 3150–3152.

Guo,S.H. *et al.* (2014) iNuc-PseKNC: a sequence-based predictor for predicting nucleosome positioning in genomes with pseudo k-tuple nucleotide composition. *Bioinformatics*, **30**, 1522–1529.

Jin,G.H. *et al.* (2020) The processing, gene regulation, biological functions, and clinical relevance of N4-acetylcytidine onRNA: a systematic review. *Mol. Ther. Nucl. Acids*, **20**, 13–24.

Khan,S. (2019) DeepAcid: classification of macromolecule type based on sequences of amino acids. arXiv, preprint arXiv:1907.03532.

Li,J.P.Y. *et al.* (2020) DeepATT: a hybrid category attention neural network for identifying functional effects of DNA sequences. *Brief. Bioinform*., **22**, bbaa159.

Liu,B. *et al.* (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.

Liu,B. *et al.* (2018) iEnhancer-EL: identifying enhancers and their strength with ensemble learning approach. *Bioinformatics*, **34**, 3835–3842.

Liu,B. *et al.* (2015) repDNA: a Python package to generate various modes of feature vectors for DNA sequences by incorporating user-defined physicochemical properties and sequence-order effects. *Bioinformatics*, **31**, 1307–1309.

Lundberg,S.M. *et al.* (2017) A unified approach to interpreting model predictions. In: Guyon, I. *et al.* (ed.) *Advances in Neural Information Processing Systems 30*. Neural Information Processing Systems (NIPS), La Jolla, CA.

Lv,H. *et al.* (2020) Deep-Kcr: accurate detection of lysine crotonylation sites using deep learning method. *Brief. Bioinform*., **22**, bbaa255.

Mikolov,T. *et al.* (2013) Efficient estimation of word representations in vector space. arXiv, preprint arXiv:1301.3781.

Moncada-Torres,A. *et al.* (2021) Explainable machine learning can outperform Cox regression predictions and provide insights in breast cancer survival. *Sci. Rep*., **11**, 6968–6968.

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

Oashi,Z. *et al.* (1972) Characterization of C + located in the first position of the anticodon of *Escherichia coli* tRNA Met as N4-acetylcytidine. *Biochim. Biophys. Acta*, **262**, 209–213.

Sas-Chen,A. *et al.* (2020) Dynamic RNA acetylation revealed by quantitative cross-evolutionary mapping. *Nature*, **583**, 638–669.

Shapley,L.S. (1953) A value for n-person games. *Contrib. Theory Games*, **2**, 307–317.

Staehelin,M. *et al.* (1968) Structure of a mammalian serine tRNA. *Nature*, **219**, 1363–1365.

Tardu,M. *et al.* (2019) Identification and quantification of modified nucleosides in *Saccharomyces cerevisiae* mRNAs. *ACS Chem. Biol*., **14**, 1403–1409.

Thomas,G. *et al.* (1978) N4-Acetylcytidine. A previously unidentified labile component of the small subunit of eukaryotic ribosomes. *J. Biol. Chem*., **253**, 1101–1105.

Thomas,J.M. *et al.* (2018) A chemical signature for cytidine acetylation in RNA. *J. Am. Chem. Soc*., **140**, 12667–12670.

Thomas,J.M. *et al.* (2019) Nucleotide resolution sequencing of N4-acetylcytidine in RNA. In: Shukla, A.K. (ed.) *Chemical and Synthetic Biology Approaches to Understand Cellular Functions – Pt A*. Elsevier Academic Press Inc, San Diego, CA, pp. 31–51.

Wang,C. *et al.* (2020a) NonClasGP-Pred: robust and efficient prediction of non-classically secreted proteins by integrating subset-specific optimal models of imbalanced data. *Microb. Genom*., **6**, mgen000483.

Wang,C. *et al.* (2020b) Its2vec: fungal species identification using sequence embedding and random forest classification. *Biomed. Res. Int*., **2020**, 2468789.

Wei,L. *et al.* (2014) Improved and promising identification of human microRNAs by incorporating a high-quality negative set. *IEEE/ACM Trans. Comput. Biol. Bioinform*., **11**, 192–201.

Wei,L. *et al.* (2019) Integration of deep feature representations and handcrafted features to improve the prediction of N-6-methyladenosine sites. *Neurocomputing*, **324**, 3–9.

Wei,L. *et al.* (2017) Improved prediction of protein-protein interactions using novel negative samples, features, and an ensemble classifier. *Artif. Intell. Med*., **83**, 67–74.

Woloszynek,S. *et al.* (2019) 16S rRNA sequence embeddings: meaningful numeric feature representations of nucleotide sequences that are convenient for downstream analyses. *PLoS Comput. Biol*., **15**, 25.

Zachau,H. *et al.* (1966) Nucleotide sequences of two serine-specific transfer ribonucleic acids. *Angew. Chem. Int. Ed. Engl*., **5**, 422–422.

Zhao,W.Q. *et al.* (2019) PACES: prediction of N4-acetylcytidine (ac4C) modification sites in mRNA. *Sci. Rep*., **9**, 11112–11117.

Zou,Q. *et al.* (2019) Gene2vec: gene subsequence embedding for prediction of mammalian N-6-methyladenosine sites from mRNA. *RNA*, **25**, 205–218.