ACS OMEGA

http://pubs.acs.org/journal/acsodf

Article

Fast and Efficient Design of Deep Neural Networks for Predicting N^7-Methylguanosine Sites Using autoBioSeqpy

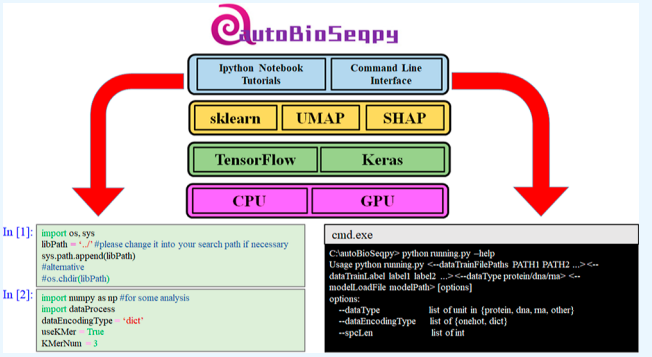
Yonglin Zhang,[¶] Lezheng Yu,[¶] Runyu Jing,* Bin Han,* and Jiesi Luo*

Cite This: *ACS Omega* 2023, 8, 19728−19740

Read Online

ACCESS | Metrics & More | Article Recommendations | Supporting Information

ABSTRACT: N^7-Methylguanosine (m^{7}G) is a crucial post-transcriptional RNA modification that plays a pivotal role in regulating gene expression. Accurately identifying m^{7}G sites is a fundamental step in understanding the biological functions and regulatory mechanisms associated with this modification. While whole-genome sequencing is the gold standard for RNA modification site detection, it is a time-consuming, expensive, and intricate process. Recently, computational approaches, especially deep learning (DL) techniques, have gained popularity in achieving this objective. Convolutional neural networks and recurrent neural networks are examples of DL algorithms that have emerged as versatile tools for modeling biological sequence data. However, developing an efficient network architecture with superior performance remains a challenging task, requiring significant expertise, time, and effort. To address this, we previously introduced a tool called autoBioSeqpy, which streamlines the design and implementation of DL networks for biological sequence classification. In this study, we utilized autoBioSeqpy to develop, train, evaluate, and fine-tune sequence-level DL models for predicting m^{7}G sites. We provided detailed descriptions of these models, along with a step-by-step guide on their execution. The same methodology can be applied to other systems dealing with similar biological questions. The benchmark data and code utilized in this study can be accessed for free at http://github.com/jingry/autoBioSeeqpy/tree/2.0/examples/m7G.



INTRODUCTION

The epitranscriptome, which refers to post-transcriptional RNA modifications, plays a critical role in regulating gene expression.[1] To date, more than 170 different types of RNA modifications have been identified, with N^6-methyladenosine (m^{6}A), N^1-methyladenosine (m^{1}A), 5-methylcytosine (m^{5}C), and pseudouridine (ψ) being the most extensively studied among them.[2−4] In recent years, another RNA modification, N^7-methylguanosine (m^{7}G), has attracted the attention of researchers worldwide.[5−7] The m^{7}G modification is not only commonly found in transfer RNA, ribosomal RNA, and messenger RNA (mRNA) 5′cap, but it is also frequently present in the internal regions of mRNA in mammals.[8,9] Given its positively charged nature, the m^{7}G modification is involved in numerous important biological processes, including protein synthesis, gene expression regulation, transcript stabilization, and cell viability.[9,10] It has been suggested that abnormal m^{7}G modifications can have a significant impact on RNA processing and function and may be associated with the development of various human diseases.[11]

Advances in high-throughput sequencing have enabled transcriptome-wide mapping and analysis of RNA modifications.[12] To reveal the distribution features of the m^{7}G methylome in human cells, several HTS-based approaches have been proposed recently, including Chu's strategy,[13] AlkAniline-Seq,[14] MeRIP-seq,[15] TRAC-Seq,[16] m^{7}G-MaP-seq,[17] m^{7}G-miCLIP-seq,[18] You and Yuan's method,[19] m^{7}G-seq,[20] etc. Despite these technologies having become the de facto method for detecting m^{7}G sites, existing drawbacks such as high cost, long period, and complex procedures have significantly limited their widespread use. To address these challenges, computational methods tailored to economical, fast, and large-scale predictions have been developed. We have summarized the currently available computational methods or prediction tools for the m^{7}G sites in Table S1. As the first predictor for m^{7}G sites, iRNA-m7G was developed by Chen et al. in 2019 using a support vector machine (SVM) and three different types of features.[21] Song et al. proposed a second predictor (m7GHub), which was also developed based on SVM and multiple sequences and genomic features.[22] Other SVM-based predictors, m7g_model and m7GPredictor, were constructed by Yang et al. and Liu et al., respectively.[23,24]

Received: February 28, 2023
Accepted: May 10, 2023
Published: May 23, 2023

Based on the extreme gradient boosting (XGBoost) algorithm and six types of feature encoding schemes, Bi et al. proposed a new classifier, called XG-m7G, for identifying m^{7}G sites in the transcriptome.[25] Recently, Dai et al. introduced another XGBoost-based predictor, m7G-IFL, which used an iterative feature representation algorithm to encode RNA sequences.[26] In the realm of m7G prediction, the most commonly utilized machine learning algorithms are SVM and XGBoost. In contrast, the feature methods used to characterize m^{7}G sites in RNA sequences are more diverse, such as nucleotide property and frequency, pseudo-nucleotide composition, nucleotide chemical property, nucleotide density, nucleotide composition, pseudo k-tuple nucleotide composition, composition of k-spaced nucleic acid pairs, enhanced nucleic acid composition, and so on. Moreover, HN-CNN,[27] m7GDisAI,[28] and bounded robust principal component analysis have been developed to discover potential associations between m^{7}G sites and diseases. Though remarkable progresses have been made in this research field, significant room for improvement still exists.

Deep learning (DL) has emerged as a leading machine learning method, revolutionizing the field of artificial intelligence by delivering human-like performance in a range of applications, including computer vision, speech recognition, and complex strategic games.[29] One of the notable advantages of DL compared to other machine learning algorithms is that it does not require manual feature extraction.[30] DL relies on a synthetic neural network architecture, which consists of interconnected nodes arranged in multiple layers. This architecture enables the network to automatically perform data representation and feature extraction, allowing it to recognize and classify patterns in complex data sets. Recently, DL algorithms have gained significant attention for their application in predicting RNA modifications. Many of the current approaches in RNA modification prediction utilize standard convolutional neural networks (CNNs), recurrent neural networks (RNNs), or other established neural network architectures that are well-suited for modeling the complex features of RNA sequences.[31] However, the successful application of DL requires not only substantial domain knowledge but also a significant investment of time and effort as the performance of the DL crucially depends on designing a tailored network architecture for the data.[32] Thanks to DL frameworks such as TensorFlow, PyTorch, or Keras, building and training neural networks have become easier for developers, researchers, methodologists, academics, and anyone interested in using them. These frameworks provide a convenient interface for implementing the operations required to build and train neural networks, allowing users to focus on designing and experimenting with different architectures and hyperparameters. In addition, several libraries or tools based on the above frameworks have emerged to make DL more accessible in certain areas.[33−35] One of these, a Keras-based tool for biological sequence classification called autoBioSeqpy, was developed previously by our group.[36] The design concept of autoBioSeqpy is to separate and integrate the entire DL modeling process. We have separated out the network architecture design as an independent part so that users can easily develop and tune the model as they wish. We have also integrated all parts in a command line manner so that complex workflow can be implemented automatically with simple command line arguments. Therefore, throughout the use of autoBioSeqpy, the users only need to provide the data sets and the model architecture codes. Recently, we have applied autoBioSeqpy to practice with good results, e.g., druggable protein prediction,[37] DL algorithm development for DNA N^4-methylcytosine,[38] anti-cancer peptide design,[39] and the distinction between bacterial type III and IV secreted effectors.[40] Hence, the present study will utilize autoBioSeqpy for the rapid and efficient creation, training, and utilization of DL models for the detection of m^{7}G sites. We demonstrate on a published benchmark data set how autoBioSeqpy allows users to compare and analyze the performance of different network architectures. Based on the evaluation results, users can select the best model architecture for the data for subsequent applications.

MATERIALS AND METHODS

Benchmark Data Set. For a fair comparison with other exiting methods, we employed the same benchmark data set derived from.[21] The 801 m^{7}G site-containing RNA sequences detected from human HeLa and HepG2 cells were collected as positive samples. All sequences contained a central m^{7}G with a length of 41 base pairs (bp). The reduction of sequence redundancy was carried out using CD-HIT with the sequence identify threshold set to 0.80.[41] After this procedure, 741 sequences were kept in the data set. The same numbers of non-methylated guanosine sites were randomly sampled from the whole-genome sequences as negative samples.

DL Background. DL relies on neural networks, a classic machine learning algorithm first proposed in the 1940s, in which neuron-like nodes in simple transformations called layers can mimic the way human brain analyzes information. Currently, the major classes of neural networks used for DL include deep neural network (DNN),[42] convolutional neural network CNN,[43] RNN,[44] autoencoders,[45] and generative adversarial network (GAN).[46] DNN is the most fundamental architecture for DL models, which consists of multiple layers of neurons stacked in an interconnected manner. More advanced models are built upon the foundational architectures of deep neural networks (DNNs). The convolutional layer in a CNN consists of filters that analyze the output from the previous layer, calculate the weighted sum of local input values, and generate the input values for the next layer. The hidden layers of the RNN can be thought of as memory states, enabling the model to capture dependencies between previously observed values and updated values at each time step in the sequence. Long short-term memory (LSTM)[47] and gated recurrent unit (GRU)[48] are two of the most widely used variants of RNNs, known for their ability to learn more efficiently in tasks that involve long-term dependencies. AEs consist of an encoder and a decoder that work together to learn a low-dimensional representation of the input data. The encoder maps the input to a compressed representation in a lower-dimensional space, while the decoder reconstructs the original input from the compressed representation. GANs are composed of two neural networks, a generator and a discriminator, which are trained in parallel to generate data points that are indistinguishable from the real data.

Overview of autoBioSeqpy. autoBioSeqpy contains several modules to support automatic data transfer and modeling. The *ParaParser* module is designed to receive and configure parameters used in the command-line instructions, such as data types, encoding methods, file paths, labels, figures, GPU, batch size, the epoch, loss, etc. The *dataProcess* module contains several encoding methods to transform input character sequences into a format than can be processed by

https://doi.org/10.1021/acsomega.3c01371
ACS Omega 2023, 8, 19728−19740

DNNs. The *moduleRead* module loads and initializes the user-designed neural network architecture code and further launches Keras for modeling. The *analysisPlot* module uses a range of metrics and figures to evaluate the performance of DL models. A distinctive feature of autoBioSeqpy is that it runs in a command line environment, so many complex functions can be easily implemented by using simple parameters. Table S2 provides a detailed explanation of some commonly used parameters.

Feature Encoding Methods. *One-Hot Encoding.* A 4-number one-hot vector is used to represent each nucleotide in an RNA sequence. A one-hot vector is a vector that contains a single one and all other elements are zeros, with the position of the one indicating the nucleotide species according to an arbitrary but consistent mapping (for example, A: [1, 0, 0, 0], C: [0, 1, 0, 0], G: [0, 0, 1, 0], and U: [0, 0, 0, 1]). Thus, each RNA sequence is represented by a two-dimensional vector of shape $(4, L)$, where L is the length of the sequence.

RNA Nucleotide Compositions. These composition-based features consist of 4 mono-nucleotide frequencies (% A, % C, % G, % U), 16 di-nucleotide frequencies (% AA, % AC, % AG, % AU, % CA, % CC, % CG, % CU, % GA, % GC, % GG, % GU, % UA, % UC, % UG, % UU), 64 tri-nucleotide frequencies (% AAA − % UUU), 256 tetra-nucleotide frequencies (% AAAA − % UUUU), and 1024 penta-nucleotide frequencies (% AAAAA − % UUUUU).

Secondary Structure and Thermodynamic Stability Features. The RNA secondary structure was predicted by the "RNAfold" function from ViennaRNA package based on free energy minimization and the partition function method.[49] These folding-based features include minimum free energy (MFE), dG, MFEI1, MFEI2, MFEI3, energy free energy (EFE), nEFE, diff, frequency of the MFE structure (Freq), nFreq, dP, GC_Stem, AU_Stem, GU_Stem, and Avg_BP_stem.[50] Three of these features, including MFE, EFE, and Freq, are calculated directly by RNAfold. dG is the normalized minimum free energy per length (=MFE/length), MFEI1 is the ratio between dG and % G + C (=dG/% G + C), MFEI2 is the ratio of the minimum free energy to the total number of base pairs in the secondary structure (=MFE/tot_bp), MFEI3 is the ratio of dG to the number of stems (=dG/stem), nEFE is normalized ensemble free energy (=EFE/length), diff is the difference between MFE and EFE (=|MFE − EFE|/length), nFreq is the normalized frequency of the MFE structure (=Freq/length), dP is the normalized base-pairing propensity (=tot_bp/length), GC_Stem is the ratio of the number of GC to the number of stems, AU_Stem is the ratio of the number of AU to the number of stems, GU_Stem is the ratio of the number of GU to the number of stems, and Avg_BP_stem is the ratio of the number of bases to the number of stems.

Base-Pair Distance Features. The base-pair features include NonBP_A, NonBP_C, NonBP_G, NonBP_U, and Non_BPP. NonBP_A is non-pairing probability for nucleotide A, NonBP_C is non-pairing probability for nucleotide C, NonBP_G is non-pairing probability for nucleotide G, NonBP_U is non-pairing probability for nucleotide U, and Non_BPP is overall non-pairing probability for all four nucleotides.

Local Structure−Sequence Triplet Elements. For any 3 adjacent nucleotides, there are 8 possible structures: "(((", "((.", "(..", "(.(", ".((", ".(.", "..(", and "...", where brackets "(" represented paired nucleotide and dot "." represented an unpaired nucleotide. Considering the middle nucleotide among the adjacent nucleotides, which can be A, C, G, or U base, there are 32 possible structure−sequence combinations.[51]

Model Architectures. We investigated various neural network architectures, including CNNs, RNNs, DNNs, and hybrid models. We extensively explored the impact of hyperparameters on the performance of DL models as their optimal settings can vary depending on the specific task. For this reason, we sampled a variety of hyperparameter sets for different model architectures, including convolution layers (1, 2, and 3), kernel size (3, 5, 7, 9, and 11), number of filters (50, 150, and 250), pool size (2, 4, 6, 8, and 10), LSTM layers (1, 2, and 3), number of units in the LSTM layer (32, 64, 128, and 256), GRU layers (1, 2, and 3), and number of units in the GRU layer (32, 64, 128, and 256). Details about optimal hyperparameters and model architectures are provided in Tables S3−S6, respectively.

Model Interpretation and Visualization. autoBioSeqpy also integrates several data analysis methods such as uniform manifold approximation and projection (UMAP) and SHapley Additive exPlanations (SHAP) for understanding and visualizing the developed DL models. The new autoBioSeqpy plugin, LayerUMAP, can generate the manifold projection of any hidden layer of neural network and observe the evolution of internal representation layer by layer during the training process.[52] Another plugin, DeepSHAP, can estimate the contribution of each feature value to the model prediction using the heat maps or logo plots.

Performance Metrics. After training, autoBioSeqpy automatically evaluates the performance of the model. Several standard metrics and plots are employed, including accuracy (ACC), precision (PRE), F-value, recall, Matthew's correlation coefficient (MCC), receiver operating characteristic (ROC), precision-recall (PR) accuracy and loss (acc−loss) curves. They are defined as follows

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

$$\mathrm{PRE} = \frac{\mathrm{TP}}{\mathrm{TP} + \mathrm{FP}} \tag{2}$$

$$F\text{-value} = 2\frac{\mathrm{TP}}{2\mathrm{TP} + \mathrm{FP} + \mathrm{FN}} \tag{3}$$

$$\mathrm{Recall} = \frac{\mathrm{TP}}{\mathrm{TP} + \mathrm{FN}} \tag{4}$$

$$\mathrm{MCC} = \frac{(\mathrm{TP} \times \mathrm{TN}) - (\mathrm{FN} \times \mathrm{FP})}{\sqrt{(\mathrm{TP} + \mathrm{FN})(\mathrm{TN} + \mathrm{FP})(\mathrm{TP} + \mathrm{FP})(\mathrm{TN} + \mathrm{FN})}} \tag{5}$$

where TP, TN, FP, and FN are the numbers of true positives, true negatives, false positives, and false negatives, respectively. The ROC curve displays the relationship between the true positive rate and false positive rate of a binary classifier by considering all possible classification thresholds that can be interpreted as probabilities. PR plots precision against the recall at all possible thresholds. The area under the ROC and PR curves falls in the interval of [0,1] (0.5 = random guessing, 1 = perfect classification) and gives an impression of the general performance of the model.

https://doi.org/10.1021/acsomega.3c01371
ACS Omega 2023, 8, 19728−19740

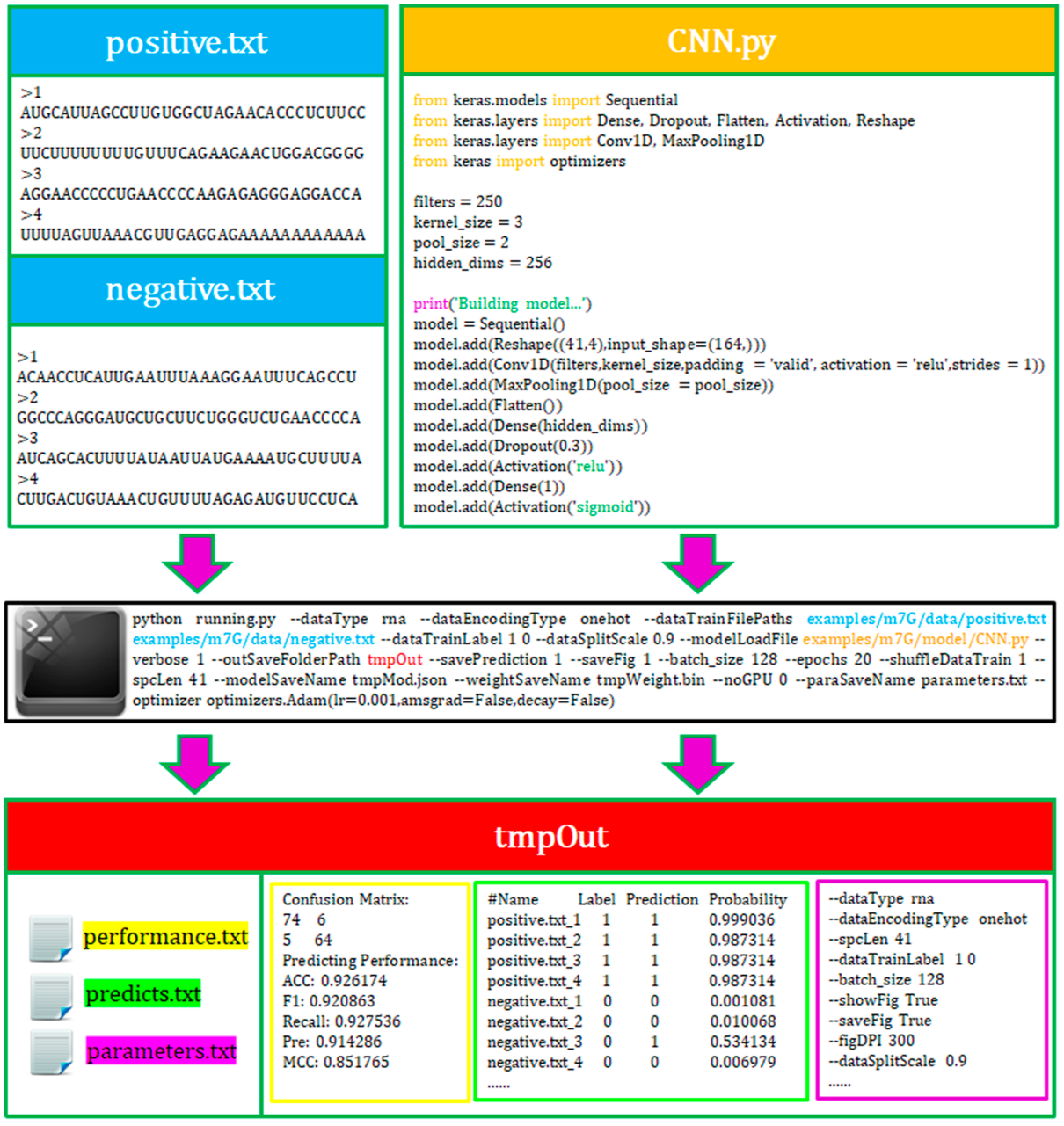


Figure 1. Usage of autoBioSeqpy. Training the CNN architecture as an example, including data sets, code, command, and output.

RESULTS AND DISCUSSION

Practical Application of autoBioSeqpy to Develop, Evaluate, and Compare DL Models for m^{7}G Sites' Prediction. Several representative DL models constructed with different network architectures were used for benchmarking. We will now elucidate in detail how to implement and train the models in autoBioSeqpy using three representative cases. Figure 1 illustrates the application of a CNN on RNA sequences, with the input data type set to RNA (--dataType rna). To convert each sequence into a two-dimensional vector of dimensions (4, L), where L represents the sequence length, the "--dataEncodingType onehot" parameter was used. In this case, the sequence length was set to 41 (--spcLen 41) to extract a 20 bp flanking sequence on either side of the central methylated guanosine. The "--dataSplitScale" parameter was set to 0.9 to randomly divide the input data sets into a 90% training-validation set and a 10% test set, stratified by category (--dataTrainLabel 1 0), with label 1 indicating positive samples (m^{7}G sites) and label 0 indicating negative samples (non-m^{7}G sites). The training data, consisting of positive and negative samples, was shuffled (--shuffleDataTrain 1) to prevent overfitting, and the model was trained for 20 epochs (--epochs 20) with a batch size of 128 (--batch_size 128) using a NVIDIA GeForce RTX 3070 GPU (--noGPU 0). The Adam optimizer was employed with a learning rate of 0.001 [--optimizer optimizers.Adam(lr = 0.001, amsgrad = False, decay = False)] to minimize categorical cross-entropy loss between the predicted and target outputs. After execution, various result files, such as prediction performance, prediction probabilities, command-line arguments, epoch−loss curve, ROC curve, PR curve, and model weights, were saved in the "tmpOut" folder (--outSaveFolderPath tmpOut). autoBioSeqpy generated the epoch−loss curve, ROC curve, and PR curve

https://doi.org/10.1021/acsomega.3c01371
ACS Omega 2023, 8, 19728−19740

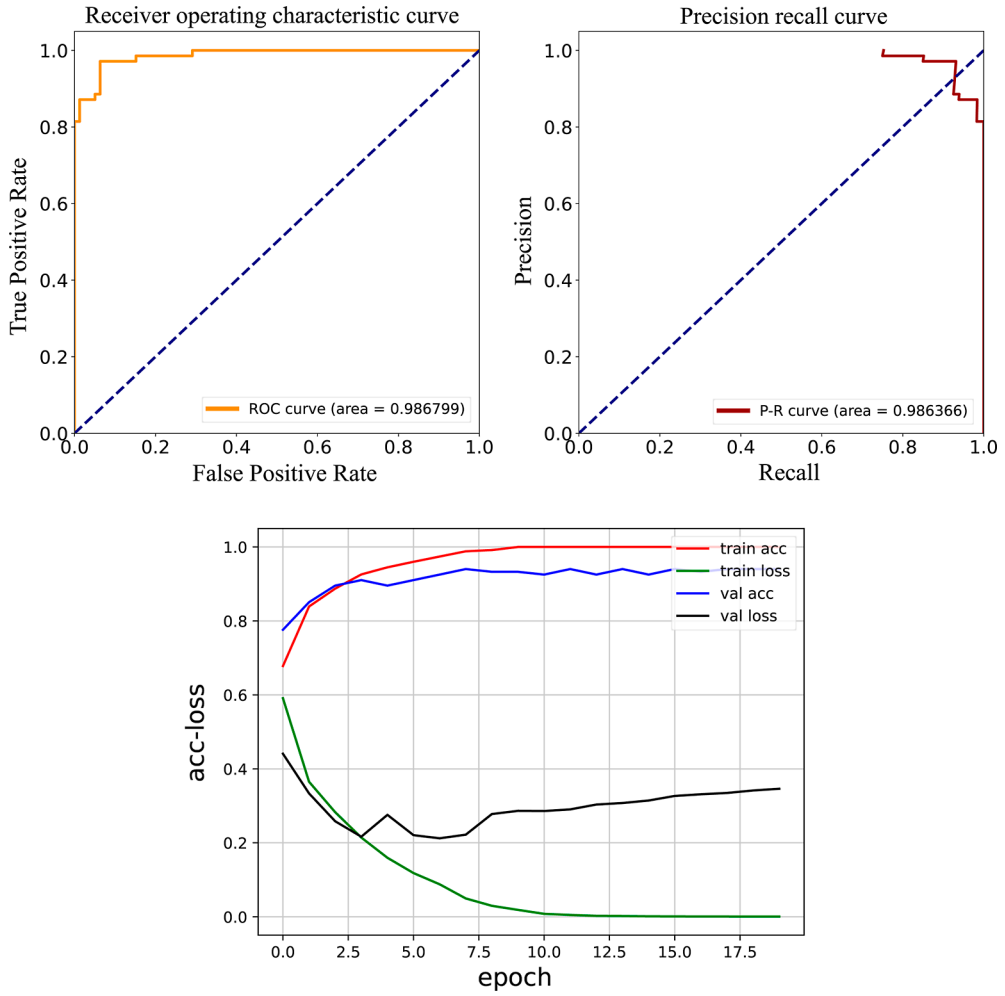


Figure 2. ROC, PR, and acc−loss curves were generated by the autoBioSeqpy tool for the CNN model using the benchmark data set.

presented in Figure 2. The above example of the CNN model is also applicable to other sequence-level DL models, such as BiLSTM, BiGRU, CNN-BiGRU, and CNN-BiLSTM. In practice, the users do not need to change the most of the command-line arguments but only need to replace the model file name, e.g., change "CNN.py" to "BiLSTM.py". Figure 3 shows another use case and example application of autoBioSeqpy. To train the DNN model, autoBioSeqpy was instructed to use feature vectors as the input data format (--dataType other). The built-in encoding method was turned off by setting the parameter "--dataEncodingType" to "other", allowing autoBioSeqpy to read in externally calculated features directly. The 1416-dimensional features, including 1364 RNA nucleotide compositions, 15 secondary structure and thermodynamic stability features, 5 base-pair distance features, and 32 local structure-sequence triplet elements, were calculated and used as input to the DNN model. With version ≥2.0, autoBioSeqpy supports the use of multiple models for different types of data. The output layers of each model will be merged into a new output layer through a dense layer. To use this new function, users should provide at least two DL models and four corresponding data sets. For example, if users want to integrate the above CNN and DNN models for training, they can use the following command-line commands (Figure 4). The parameter "--dataTrainModelInd" is designed to specify which model the data set belongs to, so the value should not be greater than the model's index. Specifically, the parameter "0 0 1 1" after "--dataTrainModelInd" means that the first two data set files (positive.txt and negative.txt) will be fed into the first model file (CNN_hybrid.py), and the remaining two data set files (pos_feature.txt and neg_feature.txt) will be fed into the second model file (DNN_hybrid.py).

Once trained, autoBioSeqpy evaluates the model automatically and generates a confusion matrix along with five performance metrics. Along with text output, autoBioSeqpy also creates visualizations of the model's performance through ROC and PR curves (Figures S1 and S2). Figure 5 shows the comparisons between different DL models in terms of seven evaluation metrics. The CNN model achieved the highest modification site classification accuracy (92.6%), which is slightly higher than the second-best-performing models, CNN + DNN and CNN-BiGRU (91.9%). Their performance is followed by CNN-BiLSTM (91.3%), BiGRU (89.9%), BiLSTM (88.6%), and DNN (87.2%). Additionally, the same ranking of the prediction models was observed in *F*1-value, MCC, auROC, and auPR, respectively. The CNN + DNN model showed the highest precision (95.5%), while the BiGRU model provided the highest recall (97.0%). Overall, CNN delivered the best performance on most performance criteria, while sequence-level deep models consistently outperformed DNN. The results indicate that DL models trained on raw RNA sequence features are more effective in capturing the

https://doi.org/10.1021/acsomega.3c01371
ACS Omega 2023, 8, 19728−19740

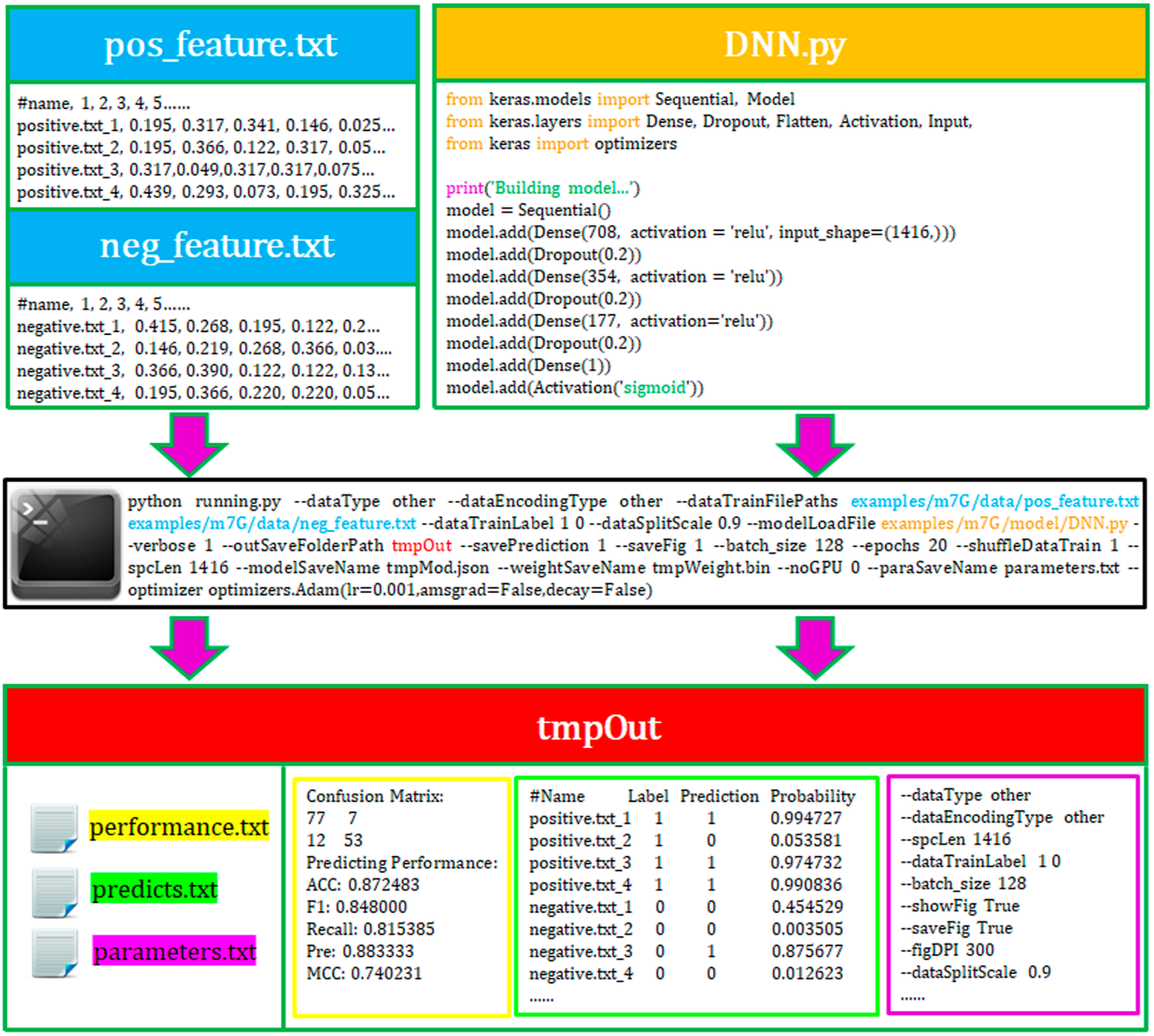


Figure 3. Usage example of DNN architecture.

distinguishing distribution patterns between m^7G sites and non-m^7G sites.

LayerUMAP Dissects DL Models Layer by Layer and Observes the Evolution of m^7G Sites and Non-m^7G Sites. By using LayerUMAP, it is possible to access the hidden layers of the trained models created by autoBioSeqpy. It is also a command-line tool that is particularly easy to use, but it must be used in conjunction with autoBioSeqpy. As an example, the CNN model was trained using the command shown in Figure 1. To perform a layer-by-layer dissection of the architecture, the following command can be used:

```
python tool/layerUMAP.py --paraFile tmpOut/parameters.txt --outFigFolder tmpOut --metric cosine --n_neighbors 28 --min_dist 0.8 --interactive 1
```

The command includes the parameter "--paraFile", which contains information about the trained CNN model. Using this information, we generated 2D UMAP maps of m^7G sites (labeled as 1 in the red point cloud) and non-m^7G sites (labeled as 0 in the purple point cloud) based on the latent space at different network layers, as shown in Figure 6. The maps allow us to observe how the features learned by the model evolve along the layer hierarchy, with m^7G and non-m^7G sites mixed at the first few layers and culminating in clear separation in the output layer. To make the distribution of data points in the projection more spread, the UMAP parameters n_neighbors, min_dist, and metric were set to 28, 0.8, and cosine, respectively. When the "interactive" parameter is used, layerUMAP provides a list of the names and indexes of the layers in the trained model, allowing the user to choose which layer to use as the projection. By default, the projection of the last hidden layer is output. The parameter "--theme" allows users to specify the color, background, and theme for UMAP plotting. Here, we selected the fire theme (--theme fire) to investigate the internal representation of samples evolving in the DNN model (Figure 7). Similarly, we observed that the features extracted by deeper dense layer became more discriminative. Finally, using the same command, layerUMAP

https://doi.org/10.1021/acsomega.3c01371
ACS Omega 2023, 8, 19728−19740

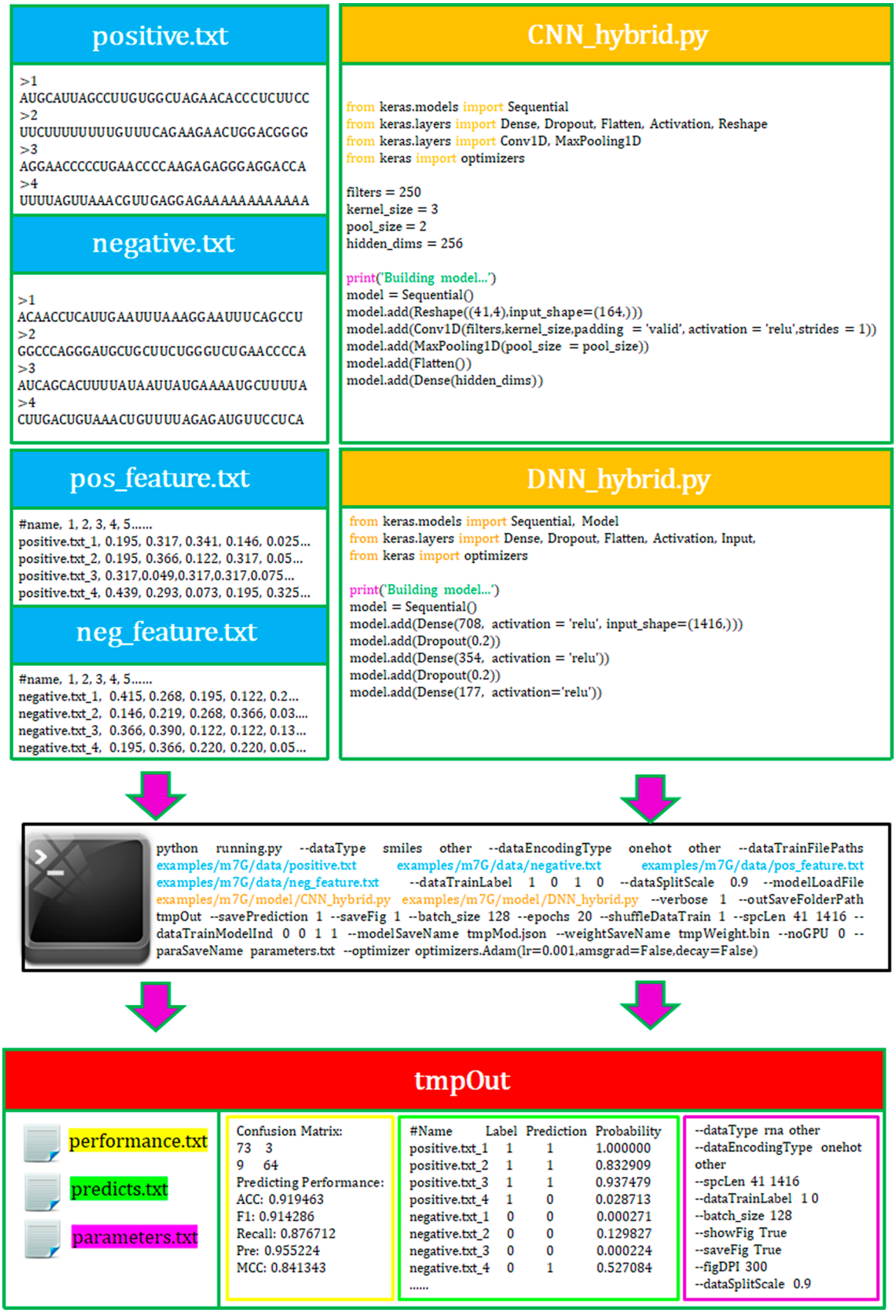


Figure 4. Usage example of the hybrid architecture combining CNN and DNN (CNN + DNN).

showed the projection of the output layers for all proposed DL models (Figure 8).

DeepSHAP Measures the Feature Importance of Raw RNA Sequences for Predicting m^7G Sites. The Deep-

https://doi.org/10.1021/acsomega.3c01371
ACS Omega 2023, 8, 19728−19740

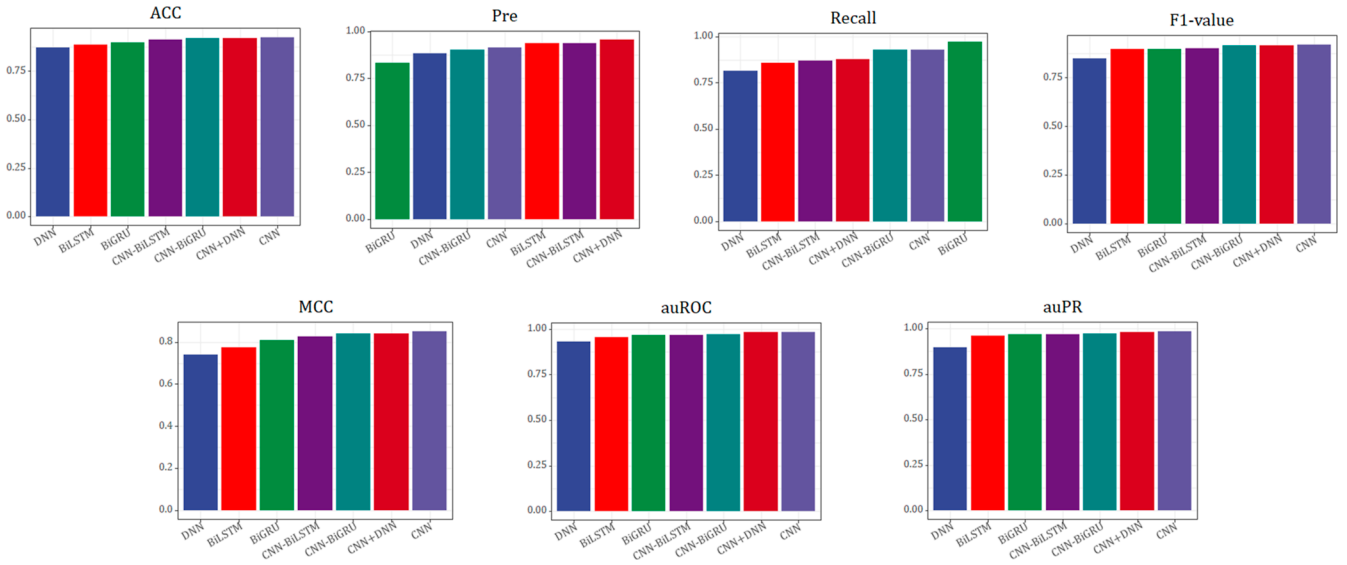


Figure 5. Performance comparison of different DL models.

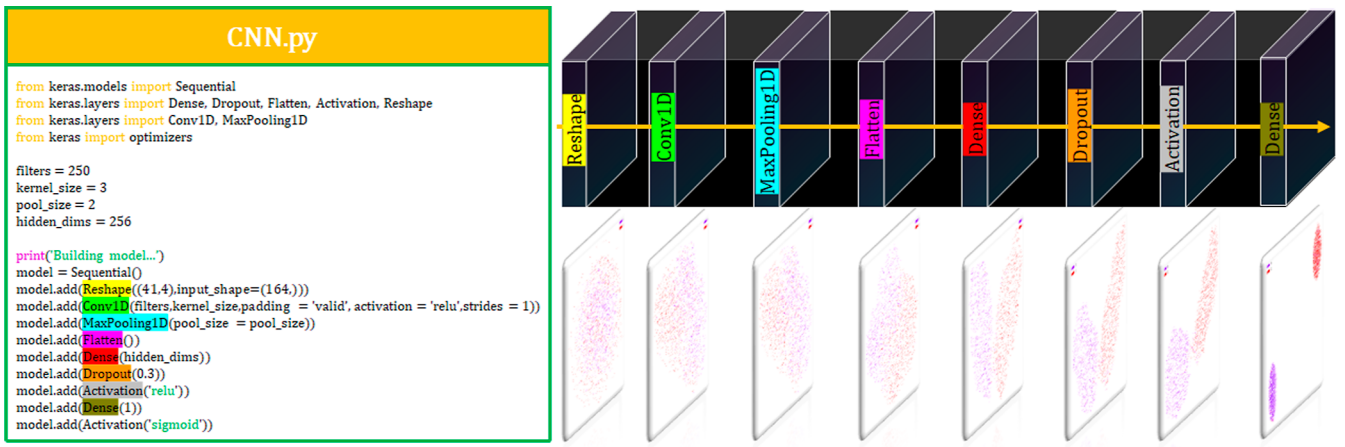


Figure 6. UMAP projection of the CNN architecture depicts the evolution of the model's layers from one to another. The point clouds in different colors indicate the m^7G and non-m^7G sites and how they are clustered by the model.

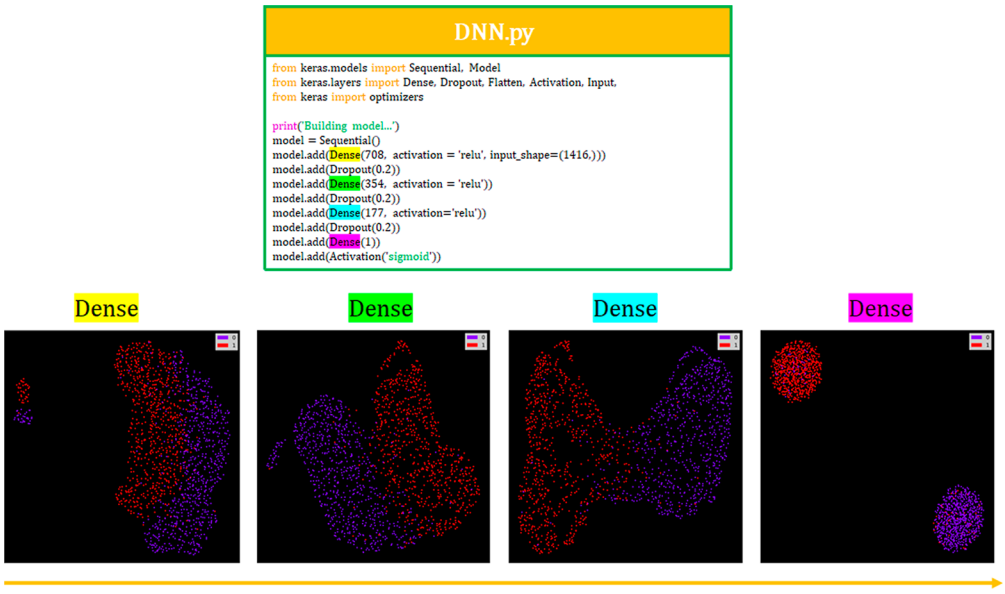


Figure 7. UMAP projection of layer-to-layer evolution of the DNN architecture.

https://doi.org/10.1021/acsomega.3c01371
ACS Omega 2023, 8, 19728−19740

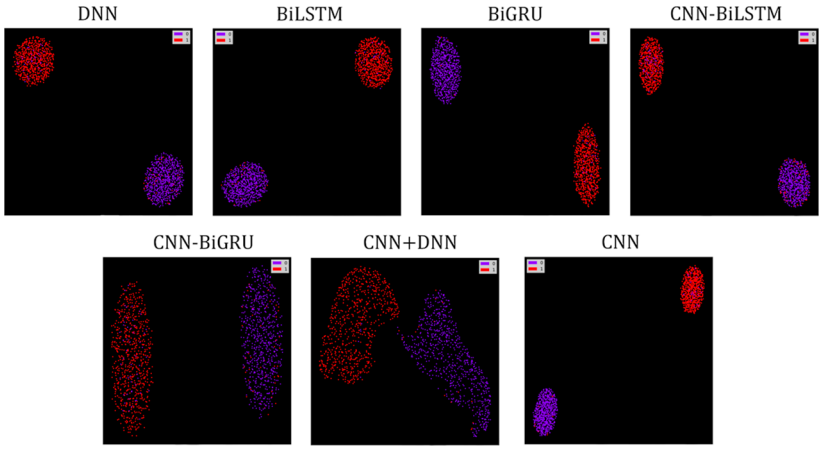


Figure 8. UMAP projection of the output layer representation learned from the DL model.

A

	A	G	C	U
1	0.24	-0.02	0.20	0.09
2	0.41	0.61	0.92	0.11
3	0.75	0.46	1.06	0.11
4	1.64	0.51	1.44	0.10
5	0.78	0.42	0.82	0.54
6	0.51	0.40	0.52	0.20
7	0.64	0.92	0.52	-0.19
8	0.96	1.68	0.49	0.68
9	1.65	1.43	0.60	1.15
10	2.03	0.98	1.03	0.90
11	0.36	-0.12	1.33	2.03
12	0.35	1.61	0.27	0.21
13	0.85	1.04	0.58	-0.23
14	1.50	3.35	1.05	0.98
15	1.55	3.11	1.33	0.69
16	3.40	8.68	2.70	0.36
17	3.46	5.52	1.73	0.50
18	1.69	4.20	1.95	0.74
19	4.82	6.27	2.65	1.44
20	12.2	16.8	15.8	20.8
21	0.00	0.00	0.00	0.00
22	3.53	3.98	0.45	0.61
23	2.52	0.72	1.12	3.84
24	2.05	3.51	0.45	-0.87
25	1.34	2.05	1.17	0.37
26	1.30	7.07	2.07	4.27
27	1.03	6.50	1.80	3.62
28	0.94	2.33	1.10	0.93
29	1.11	3.61	0.39	4.45
30	1.39	4.55	0.83	5.06
31	1.49	5.94	0.89	3.36
32	1.14	6.47	1.09	3.24
33	1.87	6.87	0.61	5.49
34	2.67	5.10	1.33	2.89
35	2.81	4.40	2.11	1.56
36	1.84	3.80	2.87	3.18
37	0.59	0.84	0.88	1.45
38	0.07	0.49	0.45	0.64
39	-0.21	0.19	0.73	0.35
40	-0.18	-0.17	0.44	0.13
41	0.00	0.00	0.00	0.00

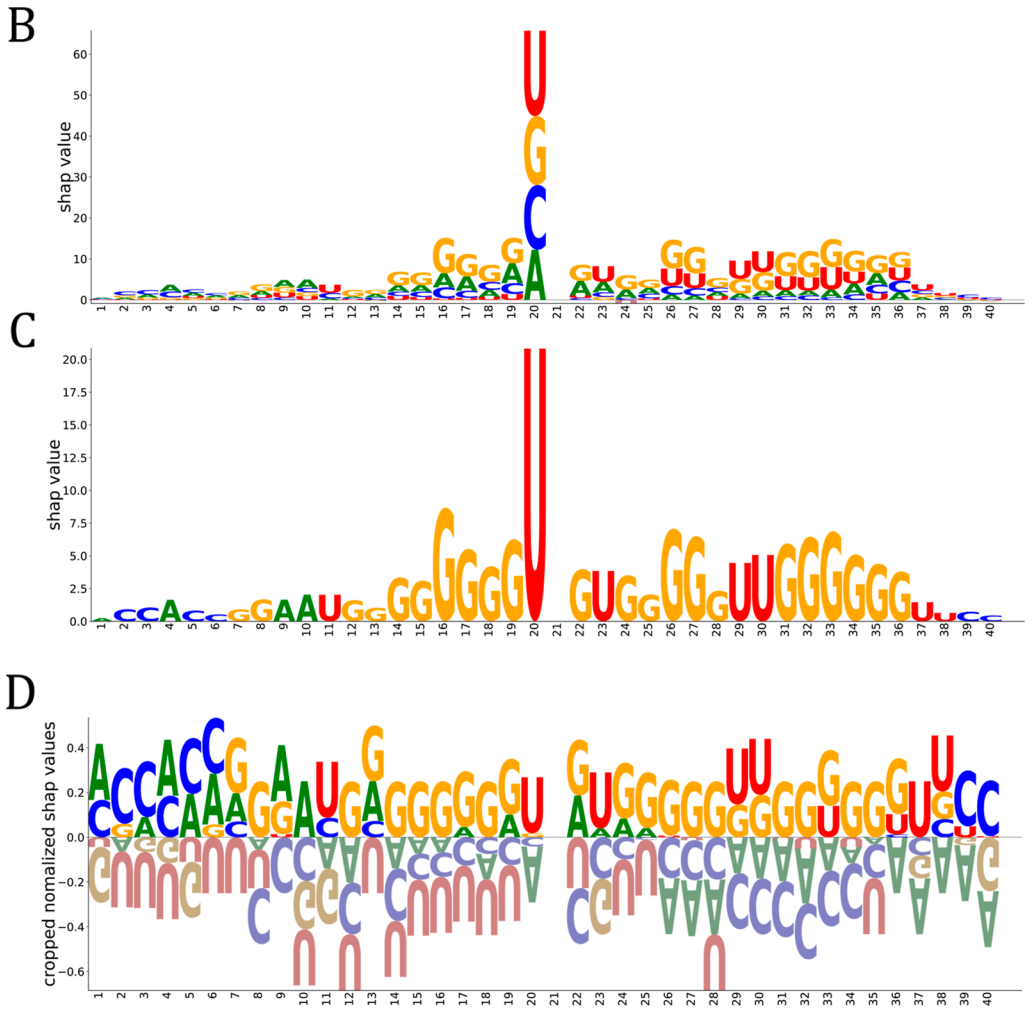


Figure 9. Evaluation of sequence feature importance. (A) SHAP value indicates the contribution of each nucleotide at each position to the CNN model prediction. (B) SHAP values are shown as sequence logos. (C) Sequence logos displaying the maximum SHAP values at each position. (D) Sequence logos displaying normalized SHAP values.

SHAP method was employed to extract one-hot encoding features and train the CNN model with the best hyperparameter configurations, as determined from the grid search procedure described earlier. Following training, a per-sample importance score was assigned to each feature from the trained CNN model. These importance scores represent the effect of the features on the base value in the model output, which is calculated based on a game-theoretic Shapley value[53] for optimal credit allocation. In order to provide a general overview of feature importance in the trained model, the mean absolute SHAP values were calculated for the entire data set (Figure 9A). To facilitate a more intuitive understanding,

https://doi.org/10.1021/acsomega.3c01371
ACS Omega 2023, 8, 19728−19740

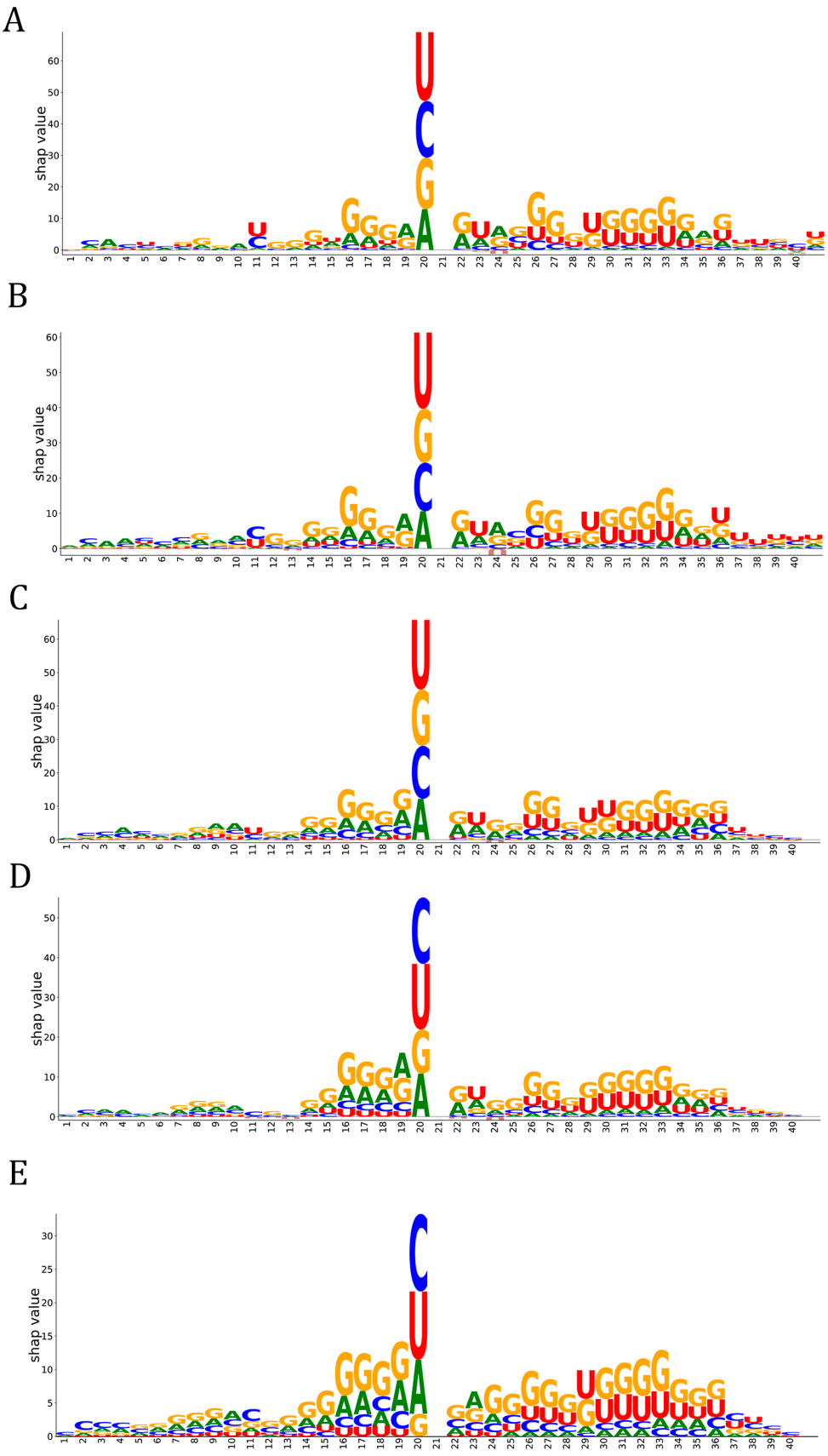


Figure 10. SHAP values associated with m^7G site identification for CNN (A), BiGRU (B), BiLSTM (C), CNN-BiGRU (D), and CNN-BilSTM (E), as determined by DeepSHAP.

https://doi.org/10.1021/acsomega.3c01371
ACS Omega 2023, 8, 19728−19740

DeepSHAP integrates sequence logo plots instead of the summary violin plots to display the above computed SHAP values (Figure 9B). In addition, the maximum SHAP valued for each position are shown in Figure 9C to identify the most important features. Taken together, we found that U at position 20 made the greatest contribution to predicting m^7G for the CNN model. The SHAP values were further normalized to highlight the favored and disfavored nucleotides (Figure 9C). We observed that G at many positions made a significant contribution to the prediction. Finally, we used DeepSHAP to measure and visualize the contribution of input sequences to the output predictions for other sequence-level DL models (Figure 10). The biggest difference in these models occurs at position 20. In CNN, BiGRU, and BiLSTM, the U at this position contributed most to the prediction, while the most importance feature was C at the corresponding position in CNN-BiGRU and CNN-BiLSTM.

Comparison with Previously Published Methods. To further validate DL performance for predicting m^7G sites, we compared our predictions for the best model with those of other state-of-the-art predictors on the same benchmark data set, including iRNA-m7G,[21] m7GHub,[22] XG-m7G,[25] and m7G-IFL.[26] Other new prediction methods, such as BERT-m7G[31] and m7G-DPP,[54] were not considered for comparison due to the lack of an online web server. Results of all comparisons are listed in Table S7. Notably, the detailed results of iRNA-m7G, m7GHub, and m7G-IFL were directly available from Dai et al.'s study,[26] and those of XG-m7G were accessed from Bi et al.'s work.[25] Our proposed best DL model is superior to iRNA-m7G and m7GHub and slightly better than XG-m7G and m7G-IFL. It provided the highest scores of *ACC* (92.6%), *Recall* (92.8%), and *MCC* (0.852). m7G-IFL achieved the second-best performance with *ACC* of 92.5%, *Recall* of 92.4%, and *MCC* of 0.850, respectively, while XG-m7G yielded the highest score of *AUC* (0.972).

CONCLUSIONS

Currently, over 150 types of chemical modifications have been found in cellular RNAs; however, except for a few common types, such as m^6A, m^5C, m^1A, and Ψ, the functions and mode of regulation of vast majority of RNA modifications are still unclear. To investigate their biological function, a necessary first step is to precisely identify the genome locations where the modifications occur. To date, numerous computational methods and tools have been developed for this purpose, especially with the rapid rise of artificial intelligence techniques. Nevertheless, the development of a suitable and effective computational model remains a complex process that requires a significant amount expertise, time investment, and computer modeling capability. This has hindered the extensive application of AI methods in many areas, including RNA chemical modifications. To address this problem, a different solution has been proposed to create an open-source and user-friendly environment that provides developers and end users with comprehensive support for model training, evaluation, and application across a broad range of questions. The autoBioSeqpy tool is our effort in this direction. We developed autoBioSeqpy to facilitate the creation of reproducible workflows and results and reduce the tedious modeling process in the routinely performed biological sequence classification tasks. In the present work, we tried to introduce autoBioSeqpy into the epitranscriptome and explore the use of DL algorithms for a particular epitranscriptomic mark, such as m^7G. Our goal is not to develop a specific algorithm or method but to provide a step-by-step guide to readers on how to use the tool to solve their own problems. Finally, autoBioSeqpy's capability extends far beyond the case studies described here.

ASSOCIATED CONTENT

Supporting Information

The Supporting Information is available free of charge at https://pubs.acs.org/doi/10.1021/acsomega.3c01371.

Summary of six existing m7G prediction tools in the literature; command line parameters and usage instructions for autoBioSeqpy; hyperparameter optimization of the CNN and CNN + BiLSTM models on the benchmark data sets; hyperparameter optimization of the BiLSTM and CNN + BiLSTM models on the benchmark data sets; hyperparameter optimization of the BiGRU and CNN + BiGRU models on the benchmark data sets; detailed description of seven DL architectures related to m7G site prediction; performance comparison of the proposed model with state-of-the-art methods using the same benchmark data sets; ROC, PR, and acc−loss curves generated by the autoBioSeqpy tool for the DNN model on the benchmark data set; and ROC, PR, and acc−loss curves generated by the autoBioSeqpy tool for the CNN + DNN model on the benchmark data set (PDF)

AUTHOR INFORMATION

Corresponding Authors

Runyu Jing − *School of Cyber Science and Engineering, Sichuan University, Chengdu 610017, China;* orcid.org/0000-0003-3375-1014; Email: jingryedu@gmail.com

Bin Han − *GCP Center/Institute of Drug Clinical Trials, Affiliated Hospital of North Sichuan Medical College, Nanchong 637503, China*; Email: jarnihao@163.com

Jiesi Luo − *Basic Medical College, Southwest Medical University, Luzhou 646099 Sichuan, China; Key Medical Laboratory of New Drug Discovery and Druggability Evaluation, Luzhou Key Laboratory of Activity Screening and Druggability Evaluation for Chinese Materia Medica, Southwest Medical University, Luzhou 646099, China;* orcid.org/0000-0002-1199-7024; Email: ljs@swmu.edu.cn

Authors

Yonglin Zhang − *Department of Pharmacy, Affiliated Hospital of North Sichuan Medical College, Nanchong 637000, China*

Lezheng Yu − *School of Chemistry and Materials Science, Guizhou Education University, Guiyang 550024, China*

Complete contact information is available at: https://pubs.acs.org/10.1021/acsomega.3c01371

Author Contributions

¶Y.Z. and L.Y. contributed equally to this work.

Notes

The authors declare no competing financial interest.

ACKNOWLEDGMENTS

This work has been supported by the National Natural Science Foundation of China (no. 22203057), Fund of Science and Technology Department of Guizhou Province ([2017]5790-

07), Natural Science Foundation of Department of Education of Guizhou Province ([2021]021), National Natural Science Foundation of China (no. 81903660), Nanchong City and School Cooperation Project (22SXQT0346), and Joint project of Luzhou Municipal People's Government and Southwest Medical University (2020LZXNYDJ39).

REFERENCES

(1) Liu, L.; Song, B.; Ma, J.; Song, Y.; Zhang, S. Y.; Tang, Y.; Wu, X.; Wei, Z.; Chen, K.; Su, J.; et al. Bioinformatics approaches for deciphering the epitranscriptome: Recent progress and emerging topics. *Comput. Struct. Biotechnol. J.* **2020**, *18*, 1587−1604.
(2) Liu, K.; Chen, W. iMRM: a platform for simultaneously identifying multiple kinds of RNA modifications. *Bioinformatics* **2020**, *36*, 3336−3342.
(3) Dao, F. Y.; Lv, H.; Yang, Y. H.; Zulfiqar, H.; Gao, H.; Lin, H. Computational identification of N6-methyladenosine sites in multiple tissues of mammals. *Comput. Struct. Biotechnol. J.* **2020**, *18*, 1084−1091.
(4) Courtney, D. G. Post-Transcriptional Regulation of Viral RNA through Epitranscriptional Modification. *Cells* **2021**, *10*, 1129.
(5) Chen, J.; Li, K.; Chen, J.; Wang, X.; Ling, R.; Cheng, M.; Chen, Z.; Chen, F.; He, Q.; Li, S.; et al. Aberrant translation regulated by METTL1/WDR4-mediated tRNA N7-methylguanosine modification drives head and neck squamous cell carcinoma progression. *Cancer Commun.* **2022**, *42*, 223−244.
(6) Regmi, P.; He, Z. Q.; Lia, T.; Paudyal, A.; Li, F. Y. N7-Methylguanosine Genes Related Prognostic Biomarker in Hepatocellular Carcinoma. *Front. Genet.* **2022**, *13*, 918983.
(7) Yang, B.; Wang, J. Q.; Tan, Y.; Yuan, R.; Chen, Z. S.; Zou, C. RNA methylation and cancer treatment. *Pharmacol. Res.* **2021**, *174*, 105937.
(8) Trotman, J. B.; Schoenberg, D. R. A recap of RNA recapping. *Wiley Interdiscip. Rev.: RNA* **2019**, *10*, No. e1504.
(9) Wulf, M. G.; Buswell, J.; Chan, S. H.; Dai, N.; Marks, K.; Martin, E. R.; Tzertzinis, G.; Whipple, J. M.; Corrêa, I. R., Jr.; Schildkraut, I. The yeast scavenger decapping enzyme DcpS and its application for in vitro RNA recapping. *Sci. Rep.* **2019**, *9*, 8594.
(10) Furuichi, Y.; LaFiandra, A.; Shatkin, A. J. 5'-Terminal structure and mRNA stability. *Nature* **1977**, *266*, 235−239.
(11) Cui, L.; Ma, R.; Cai, J.; Guo, C.; Chen, Z.; Yao, L.; Wang, Y.; Fan, R.; Wang, X.; Shi, Y. RNA modifications: importance in immune cell biology and related diseases. *Signal Transduction Targeted Ther.* **2022**, *7*, 334.
(12) Zheng, G.; Qin, Y.; Clark, W. C.; Dai, Q.; Yi, C.; He, C.; Lambowitz, A. M.; Pan, T. Efficient and quantitative high-throughput tRNA sequencing. *Nat. Methods* **2015**, *12*, 835−837.
(13) Chu, J. M.; Ye, T. T.; Ma, C. J.; Lan, M. D.; Liu, T.; Yuan, B. F.; Feng, Y. Q. Existence of Internal N7-Methylguanosine Modification in mRNA Determined by Differential Enzyme Treatment Coupled with Mass Spectrometry Analysis. *ACS Chem. Biol.* **2018**, *13*, 3243−3250.
(14) Marchand, V.; Ayadi, L.; Ernst, F. G. M.; Hertler, J.; Bourguignon-Igel, V.; Galvanin, A.; Kotter, A.; Helm, M.; Lafontaine, D. L. J.; Motorin, Y. AlkAniline-Seq: Profiling of m7G and m3C RNA Modifications at Single Nucleotide Resolution. *Angew. Chem., Int. Ed. Engl.* **2018**, *57*, 16785−16790.
(15) Zhang, L. S.; Liu, C.; Ma, H.; Dai, Q.; Sun, H. L.; Luo, G.; Zhang, Z.; Zhang, L.; Hu, L.; Dong, X.; He, C. Transcriptome-wide Mapping of Internal N7-Methylguanosine Methylome in Mammalian mRNA. *Mol. Cell* **2019**, *74*, 1304−1316.e8.
(16) Lin, S.; Liu, Q.; Jiang, Y. Z.; Gregory, R. I. Nucleotide resolution profiling of m7G tRNA modification by TRAC-Seq. *Nat. Protoc.* **2019**, *14*, 3220−3242.
(17) Enroth, C.; Poulsen, L. D.; Iversen, S.; Kirpekar, F.; Albrechtsen, A.; Vinther, J. Detection of internal N7-methylguanosine (m7G) RNA modifications by mutational profiling sequencing. *Nucleic Acids Res.* **2019**, *47*, No. e126.
(18) Malbec, L.; Zhang, T.; Chen, Y. S.; Zhang, Y.; Sun, B. F.; Shi, B. Y.; Zhao, Y. L.; Yang, Y.; Yang, Y. G. Dynamic methylome of internal mRNA N7-methylguanosine and its regulatory role in translation. *Cell Res.* **2019**, *29*, 927−941.
(19) You, X. J.; Yuan, B. F. Detecting Internal N7-Methylguanosine mRNA Modifications by Differential Enzymatic Digestion Coupled with Mass Spectrometry Analysis. *Methods Mol. Biol.* **2021**, *2298*, 247−259.
(20) Zhang, L. S.; Liu, C.; He, C. Transcriptome-Wide Detection of Internal N7-Methylguanosine. *Methods Mol. Biol.* **2021**, *2298*, 97−104.
(21) Chen, W.; Feng, P.; Song, X.; Lv, H.; Lin, H. iRNA-m7G: Identifying N7-methylguanosine Sites by Fusing Multiple Features. *Mol. Ther.–Nucleic Acids* **2019**, *18*, 269−274.
(22) Song, B.; Tang, Y.; Chen, K.; Wei, Z.; Rong, R.; Lu, Z.; Su, J.; de Magalhães, J. P.; Rigden, D. J.; Meng, J. m7GHub: deciphering the location, regulation and pathogenesis of internal mRNA N7-methylguanosine (m7G) sites in human. *Bioinformatics* **2020**, *36*, 3528−3536.
(23) Yang, Y. H.; Ma, C.; Wang, J. S.; Yang, H.; Ding, H.; Han, S. G.; Li, Y. W. Prediction of N7-methylguanosine sites in human RNA based on optimal sequence features. *Genomics* **2020**, *112*, 4342−4347.
(24) Liu, X.; Liu, Z.; Mao, X.; Li, Q. m7GPredictor: An improved machine learning-based model for predicting internal m7G modifications using sequence properties. *Anal. Biochem.* **2020**, *609*, 113905.
(25) Bi, Y.; Xiang, D.; Ge, Z.; Li, F.; Jia, C.; Song, J. An Interpretable Prediction Model for Identifying N7-Methylguanosine Sites Based on XGBoost and SHAP. *Mol. Ther.—Nucleic Acids* **2020**, *22*, 362−372.
(26) Dai, C.; Feng, P.; Cui, L.; Su, R.; Chen, W.; Wei, L. Iterative feature representation algorithm to improve the predictive performance of N7-methylguanosine sites. *Briefings Bioinf.* **2021**, *22*, bbaa278.
(27) Zhang, L.; Chen, J.; Ma, J.; Liu, H. HN-CNN: A Heterogeneous Network Based on Convolutional Neural Network for m7 G Site Disease Association Prediction. *Front. Genet.* **2021**, *12*, 655284.
(28) Ma, J.; Zhang, L.; Chen, J.; Song, B.; Zang, C.; Liu, H. m7GDisAI: N7-methylguanosine (m7G) sites and diseases associations inference based on heterogeneous network. *BMC Bioinf.* **2021**, *22*, 152.
(29) LeCun, Y.; Bengio, Y.; Hinton, G. Deep learning. *Nature* **2015**, *521*, 436−444.
(30) Mousavi, S. M.; Beroza, G. C. Deep-learning seismology. *Science* **2022**, *377*, No. eabm4470.
(31) Zhang, L.; Qin, X.; Liu, M.; Liu, G.; Ren, Y. BERT-m7G: A Transformer Architecture Based on BERT and Stacking Ensemble to Identify RNA N7-Methylguanosine Sites from Sequence Information. *Comput. Math. Methods Med.* **2021**, *2021*, 7764764.
(32) Dong, J.; Zhao, M.; Liu, Y.; Su, Y.; Zeng, X. Deep learning in retrosynthesis planning: datasets, models and tools. *Briefings Bioinf.* **2022**, *23*, bbab391.
(33) Hu, Z.; Tang, A.; Singh, J.; Bhattacharya, S.; Butte, A. J. A robust and interpretable end-to-end deep learning model for cytometry data. *Proc. Natl. Acad. Sci. U.S.A.* **2020**, *117*, 21373−21380.
(34) Tsutsui, M.; Takaai, T.; Yokota, K.; Kawai, T.; Washio, T. Deep Learning-Enhanced Nanopore Sensing of Single-Nanoparticle Translocation Dynamics. *Small Methods* **2021**, *5*, No. e2100191.
(35) Routhier, E.; Bin Kamruddin, A.; Mozziconacci, J. keras_dna: a wrapper for fast implementation of deep learning models in genomics. *Bioinformatics* **2021**, *37*, 1593−1594.
(36) Jing, R.; Li, Y.; Xue, L.; Liu, F.; Li, M.; Luo, J. autoBioSeqpy: A Deep Learning Tool for the Classification of Biological Sequences. *J. Chem. Inf. Model.* **2020**, *60*, 3755−3764.
(37) Yu, L.; Xue, L.; Liu, F.; Li, Y.; Jing, R.; Luo, J. The applications of deep learning algorithms on in silico druggable proteins identification. *J. Adv. Res.* **2022**, *41*, 219−231.
(38) Yu, L.; Zhang, Y.; Xue, L.; Liu, F.; Chen, Q.; Luo, J.; Jing, R. Systematic Analysis and Accurate Identification of DNA N4-

https://doi.org/10.1021/acsomega.3c01371
ACS Omega 2023, 8, 19728−19740

Methylcytosine Sites by Deep Learning. *Front. Microbiol.* **2022**, *13*, 843425.
(39) Yu, L.; Jing, R.; Liu, F.; Luo, J.; Li, Y. DeepACP: A Novel Computational Approach for Accurate Identification of Anticancer Peptides by Deep Learning Algorithm. *Mol. Ther.–Nucleic Acids* **2020**, *22*, 862−870.
(40) Jing, R.; Wen, T.; Liao, C.; Xue, L.; Liu, F.; Yu, L.; Luo, J. DeepT3 2.0: improving type III secreted effector predictions by an integrative deep learning framework. *NAR: Genomics Bioinf.* **2021**, *3*, lqab086.
(41) Fu, L.; Niu, B.; Zhu, Z.; Wu, S.; Li, W. CD-HIT: accelerated for clustering the next-generation sequencing data. *Bioinformatics* **2012**, *28*, 3150−3152.
(42) Shaban, W. M.; Rabie, A. H.; Saleh, A. I.; Abo-Elsoud, M. A. Detecting COVID-19 patients based on fuzzy inference engine and Deep Neural Network. *Appl. Soft Comput.* **2021**, *99*, 106906.
(43) Huang, J.; Wang, M.; Ju, H.; Shi, Z.; Ding, W.; Zhang, D. SD-CNN: A static-dynamic convolutional neural network for functional brain networks. *Med. Image Anal.* **2023**, *83*, 102679.
(44) Shen, J.; Liu, F.; Tu, Y.; Tang, C. Finding gene network topologies for given biological function with recurrent neural network. *Nat. Commun.* **2021**, *12*, 3125.
(45) Wang, T.; Ng, W. W. Y.; Pelillo, M.; Kwong, S. LiSSA: Localized Stochastic Sensitive Autoencoders. *IEEE Trans. Cybern.* **2021**, *51*, 2748−2760.
(46) Tan, H.; Liu, X.; Liu, M.; Yin, B.; Li, X. KT-GAN: Knowledge-Transfer Generative Adversarial Network for Text-to-Image Synthesis. *IEEE Trans. Image Process.* **2021**, *30*, 1275−1290.
(47) Hochreiter, S.; Schmidhuber, J. Long short-term memory. *Neural Comput.* **1997**, *9*, 1735−1780.
(48) Zhang, B.; Xiong, D.; Xie, J.; Su, J. Neural Machine Translation With GRU-Gated Attention Model. *IEEE. Trans. Neural Netw. Learn Syst.* **2020**, *31*, 4688−4698.
(49) Lorenz, R.; Bernhart, S. H.; Höner zu Siederdissen, C.; Tafer, H.; Flamm, C.; Stadler, P. F.; Hofacker, I. L. ViennaRNA Package 2.0. *Algorithm Mol. Biol.* **2011**, *6*, 26.
(50) Lertampaiporn, S.; Thammarongtham, C.; Nukoolkit, C.; Kaewkamnerdpong, B.; Ruengjitchatchawalya, M. Identification of non-coding RNAs with a new composite feature in the Hybrid Random Forest Ensemble algorithm. *Nucleic Acids Res.* **2014**, *42*, No. e93.
(51) Xue, C.; Li, F.; He, T.; Liu, G. P.; Li, Y.; Zhang, X. Classification of real and pseudo microRNA precursors using local structure-sequence features and support vector machine. *BMC. Bioinf.* **2005**, *6*, 310.
(52) Jing, R.; Xue, L.; Li, M.; Yu, L.; Luo, J. layerUMAP: A tool for visualizing and understanding deep learning models in biological sequence classification using UMAP. *iScience* **2022**, *25*, 105530.
(53) Kim, H. K.; Yu, G.; Park, J.; Min, S.; Lee, S.; Yoon, S.; Kim, H. H. Predicting the efficiency of prime editing guide RNAs in human cells. *Nat. Biotechnol.* **2021**, *39*, 198−206.
(54) Zou, H.; Yin, Z. m7G-DPP: Identifying N7-methylguanosine sites based on dinucleotide physicochemical properties of RNA. *Biophys. Chem.* **2021**, *279*, 106697.

https://doi.org/10.1021/acsomega.3c01371
ACS Omega 2023, 8, 19728−19740