2024 年第二届大湾区杯科技竞赛

题　　号	**CG2404**		
标　　题	Prediction of gene expression using DNA promoter sequences and their mRNA half-lives		
成员信息	姓名：李梓妍	单位：华南理工大学数学学院	邮箱：milly825_m@outlook.com
	姓名：黄晓纯	单位：华南理工大学数学学院	邮箱：1683661059@qq.com
	姓名：段宏宇	单位：华南理工大学数学学院	邮箱：dhy.scut@outlook.com

签名（可电子签名）：李梓妍、黄晓纯、段宏宇

Prediction of gene expression using DNA promoter sequences and their mRNA half-lives

Hongyu Duan[1], Xiaochun Huang[1], Ziyan Li[1]

1 Department of Statistics and Financial Mathematics, School of Mathematics, South China University of Technology, Guangzhou, 510006, China

Abstract

Gene expression is a core process of cellular function and organismal phenotype forma -tion, and its regulatory mechanism is complex and delicate. mRNAs are formed by transcription of genetic information from DNA sequences, and then translated into proteins to carry out their functions. mRNA half-life, i.e., the duration of its stable existence in the cell, is affected by the features of the DNA sequences such as 5-termi -nal, 3-terminal, and GC content, which play an important role in the dynamic regula -tion of gene expression.

Deep learning, through the construction of complex neural networks, is able to efficiently process and analyse large-scale biological data, including gene sequences, transcription factor binding sites, epigenetic markers, and so on. With the help of deep learning and xAI methods, important features that affect the prediction results can be efficiently identified, which provides a strong support to find out the key segments in DNA sequences that affect gene expression.

In this paper, we propose a convolutional neural network (CNN)-based model that uses DNA sequences and encoded half-lives as inputs and outputs continuous numerical results. By comparing with other models we have built (including multimodal and uni -modal models), we find that the CNN model based on half-life encoding outperforms the other models in all metrics, showing higher accuracy and efficiency.

Our source code is available at https://github.com/MillyLii/AI4Sci.

Introduction

Gene expression is a central process in cellular function and the formation of organismal phenotypes, which ensures the correct transmission of genetic information and the ord -erly conduct of cellular activities through fine regulatory mechanisms. The process of gene expression involves transcription from DNA to mRNA and translation of mRNA into proteins to carry out biological functions. DNA sequences contain all the genetic information related to the function of an organism, which is converted into mRNA th -rough the process of transcription, and then translated to produce functional proteins, which in turn affect the metabolism, proliferation, and differentiation of the cell.

In the regulation of gene expression, the half-life of mRNA plays an important role. mRNA half-life refers to the time for which mRNA is stable in the cell, and it directly affects the persistence and intensity of gene expression. The stability of different mRNAs varies depending on their sequence features and structure, and this stability is closely related to several features in the DNA sequence. For example, the sequence of the 5' and 3' ends of the mRNA, as well as factors such as GC content, may affect its

degradation rate and stability. Specifically, specific base alignments in certain DNA sequences may promote or inhibit mRNA degradation, thereby indirectly regulating gene expression levels.

Thus, factors such as base characterization in the DNA sequence, regulatory eleme -nts at the 5' and 3' ends, and GC content affect the half-life of mRNA to varying deg -rees, and these factors play a key role in the dynamic regulation of gene expression. These complex interactions determine the spatial and temporal characteristics of gene expression, which is the basis of cellular function and biological phenotype formation.

Deep learning technology is capable of processing and analyzing various types of large-scale biological data by constructing complex neural network models. Including information such as gene sequences, transcription factor binding sites, epigenetic mar -kers, etc., deep learning provides a powerful tool for biological research. It is not only capable of discovering potential patterns and associations that are difficult to recognize with traditional methods, but also of effectively extracting valuable features from mass -ive amounts of data. This enables us to gain a deeper understanding of gene regulation mechanisms, the molecular basis of disease occurrence, and the possibility of individua -lized treatment, thus advancing the development of precision medicine and biotechno -logy.

Convolutional neural networks (CNNs) have demonstrated their powerful feature extraction capabilities when processing DNA sequence data. DNA sequence is essen -tially a long chain structure consisting of four bases (A, T, C, and G), and localized patterns in these sequences are of great importance for gene function, expression regu -lation, and so on. While traditional sequence analysis methods often rely on manually designed features, CNN avoids manual intervention by automatically learning critical features in the data. Through multilayer convolutional operations, CNNs are able to recognize local patterns in DNA sequences, such as specific base combinations or sequence structures, which may be closely related to transcription factor binding, the location of gene regulatory elements, and other biological processes.

During training, CNNs are able to extract complex hierarchical features from raw DNA sequences, which are subsequently passed to deeper networks for more accurate classification or regression tasks. In this way, CNNs can not only capture local features in the sequence, but also learn higher-level structural information, thus providing more accurate predictions of gene expression, mutation analysis, transcription factor action, etc. The application of CNNs in genomics has shed light on the deeper laws of gene regulatory mechanisms and biological processes.

Materials and Methods

Dataset

The dataset is derived from GM12878 cell line, and the data has been processed based on hg19 coordinate system, which contains rich genomics information, including DNA sequence data and mRNA half-life features related to gene expression. The dataset is divided into three parts: the training set (train.h5), the validation set (valid.h5) and the test set (test.h5), where the training set contains 16,215 data, the validation set has 989 data and the test set contains 990 data. Each data file contains four key fields: gene_id, half-life, promoter and label.

Specifically, gene_id is the identifier of the gene, using the common ENSID standard. Half-life field contains normalized mRNA half-life data, and each record consists of 8-dimensional feature vectors, which cover several factors related to the stability of gene expression, including: 5' end length (UTR5LEN), coding region length (CDSLEN) , intron length (INTRONLEN), 3' end length (UTR3LEN), 5' end GC content (UTR

-5GC), coding region GC content (CDSGC), 3' end GC content (UTR3GC), and open reading frame exon junction density (ORFEXONDENSITY). Promoter fields are the one-hot coding processed DNA sequence data, representing the transcription start site (TSS) and its upstream and downstream 10,000 bases of the sequence, each data is a matrix with the shape of (20,000, 4), and each position in the matrix represents the one-hot coding of the corresponding base in the DNA sequence ('A':0, 'C':1, 'G':2, 'T':3). Label field is the target predicted by the model and indicates the expression level of the gene, where 0 represents low expression and 1 represents high expression.

This dataset is designed to support the development of predictive models of gene expression, especially with important applications in studies to understand the relation -ship between mRNA stability and gene expression. With these multidimensional feat -ures, the models are able to capture the complex connections between gene sequences and their expression characteristics, driving deeper exploration of gene regulatory mech -anisms.

Method

In the competition, we designed an end-to-end deep learning model based on a Convo -lutional Neural Network (CNN) framework designed to predict gene expression levels (high or low expression). The core innovation of the model is to utilize the powerful feature learning capability of CNNs to automatically extract key features from DNA sequence and mRNA half-life data.The DNA sequence data is one-hot encoded to cap -ture the sequence information of genes, while the half-life data provides important biological features related to gene stability. Through the end-to-end training process, the model is able to learn the complex relationship between DNA sequences and gene expression, further improving the prediction accuracy. Ultimately, the CNN-based deep learning model can effectively combine these multidimensional data, providing a techni -cal solution for efficient prediction of gene expression levels.

Inference of DNA sequence embedding We employed the Hyena [1] model in our model to obtain an embedded representation of the DNA sequence and pass it as an input embedding to the CNN model. This approach helps us to better capture the deep features in the DNA sequences, which improves the accuracy of the model for gene expression prediction.

The inference process of the HyenaDNA model involves several key steps that leve -rage its architecture and training to make predictions on genomic sequences. Here' s a summary of the inference process:

1. Input Preparation: The genomic sequence is tokenized into single nucleotide tokens (A, G, C, T, and N) using the model's natural DNA vocabulary. This allows the model to maintain single nucleotide resolution, which is critical for tasks such as identi -fying single nucleotide polymorphisms.

2. Context Window Setup: During inference, a context window is established that can accommodate long sequences (up to 1 million tokens). This context window is filled with the input sequence along with any learnable soft prompt tokens that have been introduced during training to help guide the model's predictions.

3. Hyena Operator Application: The input sequence is processed through the stack of Hyena operators. Each operator applies long convolutions and data-controlled gating to the input, allowing the model to capture long-range dependencies and relationships within the genomic data.

4. Prediction Generation: The model performs next nucleotide prediction based on the processed input. It generates probabilities for the next nucleotide in the sequence, effectively predicting the sequence's continuation or classifying segments of the sequence

based on learned features.

Encoding of half-life data In our model, in addition to processing DNA sequence data, we pay special attention to the encoding of mRNA half-life data. In order to be -tter mine the potential information of the half-life data, we adopt an unsupervised learning approach based on contrastive learning. Specifically, we consider each half-life data as an anchor and randomly add noise to it to generate a positive sample, which means that we create variants associated with the original data by artificially adding perturbations. At the same time, a negative sample, i.e., a sample that is not related or similar to the current anchor, is randomly selected from the entire dataset. This process allows the model to learn more accurate representation features by enhancing the model's understanding of the intrinsic features of the data.

During the training process, the noise and negative samples in each round (epoch) are randomly selected, which ensures that the model is able to learn under diverse training conditions, thus improving its robustness and generalization ability. Our goal is to drive the model's representational ability in semantic space through this approach: making each anchor and its corresponding positive sample as close as possible in the feature space, while keeping the anchors farther away from the negative samples.

To achieve this goal, we employ triplet loss as a loss function:

$$L^{TM} = \|z_a - z_p\|_2 - \|z_a - z_n\|_2 + \alpha \tag{1}$$

where z_a is the anchor, z_p is the positive, z_n is the negative, α is the margin, set to 1. Triplet loss motivates the model to be progressively optimized during training by meas -uring the similarity between anchors, positive samples, and negative samples, such that similar data points are clustered together and dissimilar points are pulled apart.

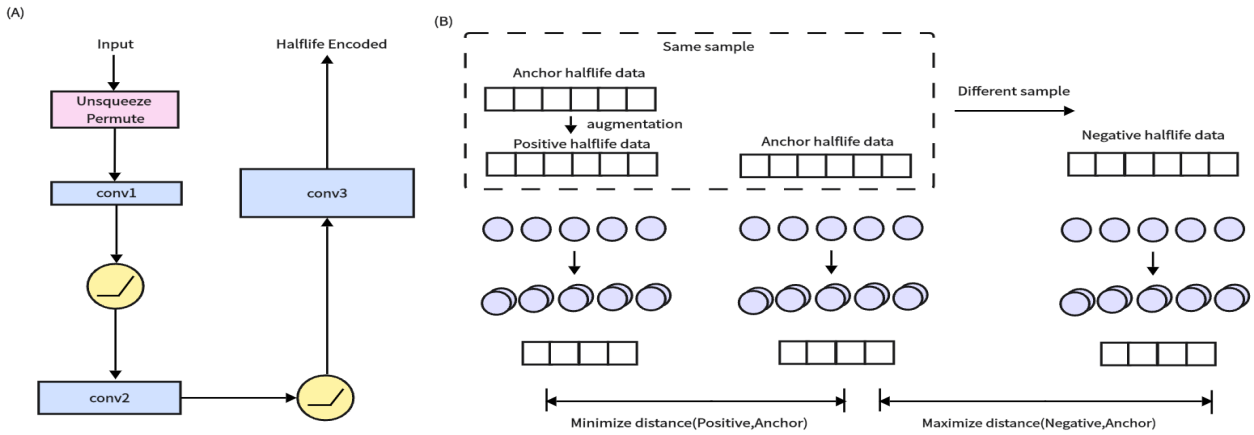

Figure 1. Using Contrastive Learning to Code Half-Life Data (A) The raw half-life data is first processed through three successive convolutional layers, each followed by the application of a ReLU activation function to introduce nonlinear transformations and enhance feature representation. This series of convolutional and activation operations transforms the raw half-life data into a high-level, encoded feature representation, providing richer input for subsequent tasks (B) In the contrast learning framework, the current sample is used as an anchor to which a positive sample is generated by adding random noise and the other samples are used as negative samples. The training objective is to encode the half-life data efficiently by optimizing the Euclidean distance to maximize the distance between the anchor and the positive samples and minimize the distance between the anchor and the negative samples.

CNN-based predictive model Our Convolutional Neural Network (CNN) model contains 4 convolutional layers, and a 2×2 MaxPooling operation is applied after each layer to reduce the spatial dimensions and preserve important features. After each con -volutional layer, an activation function (usually ReLU) is used to introduce nonlineari -ties to enhance the expressive power of the model. After the convolutional layers, the model contains three fully connected layers for mapping the extracted features to the final output of the classification task.

To prevent the model from overfitting during training, we introduce the Dropout technique. Specifically, we randomly select half of the neurons and "dropout" them (i.e., they are not involved in the computation during forward propagation) during the trai -ning process. This approach helps to reduce the over-reliance of the neural network on training data and thus improves the generalization ability of the model. The Dropout ratio is set to 0.5, which means that about 50% of the neurons are randomly discarded during each training session. After each forward propagation and Dropout operation of a small batch of training samples, the model will update the corresponding weights and bias parameters on the remaining neurons using Adam. In this way, the model is able to learn the underlying patterns in the data more efficiently and avoid poor performance on validation or test sets. We trained this model for a total of 3 hours and 51 minutes.

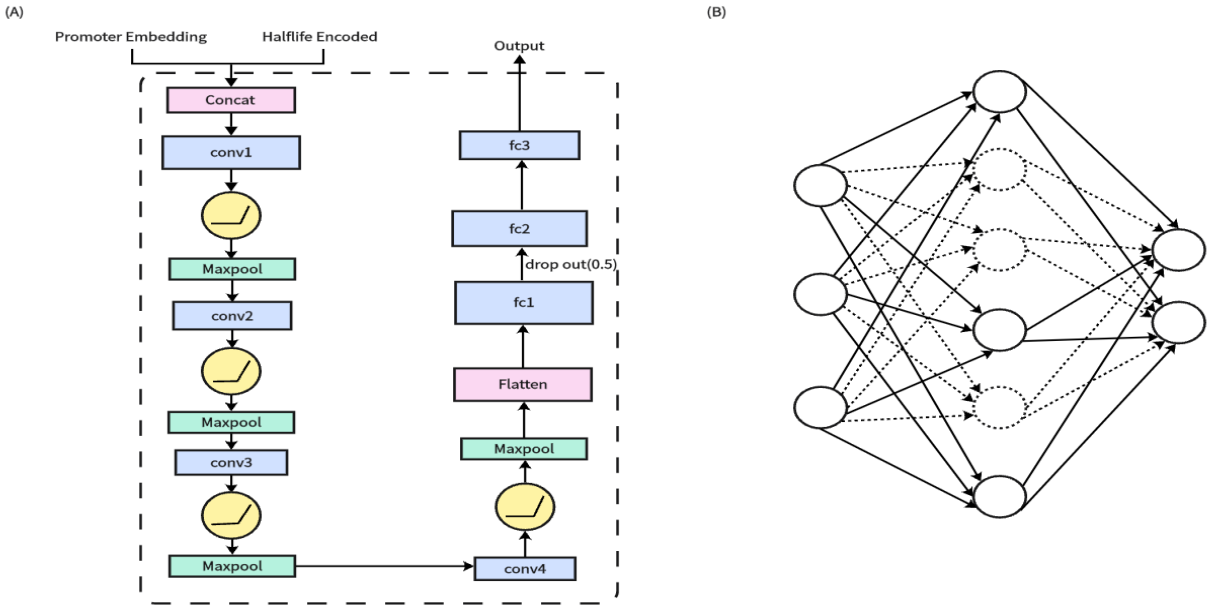

Figure 2. Modeling CNN architectures using half-life coding-based coding (A) The CNN model consists of four convolutional layers, each using ReLU activation and MaxPooling to increase nonlinearity and reduce feature dimensionality. Following the convolutional layers, three fully connected layers process the features before outputting the predictions. The loss function automatically applies a sigmoid function to normalize the model outputs within the range [0, 1], so we don't explicitly define it in the model. (B) Dropout (0.5) is applied during training, where half of the neurons in each layer are randomly discarded in each iteration, ensuring different network configurations are trained with each batch, which helps prevent overfitting.

Evaluation metrics In multi-label classification, each sample can belong to multiple classes. Traditional classification evaluation methods need adjustments to correctly

assess model performance in this context [2]. MultiLabelBinarizer() is a common tool that converts multi-label data into a binary format suitable for multi-class evaluation, allowing each label to be processed individually [3]. This approach maintains data integrity while adapting to the needs of multi-label classification.

Precision measures the proportion of correctly predicted positive instances among all instances predicted as positive. It focuses on the accuracy of the positive predictions.

$$Precision = \frac{TP}{TP + FP} \tag{2}$$

where TP is true positive and FP is false positive. Recall (or Sensitivity) measures the proportion of correctly identified positive instances among all actual positive instances. It focuses on the model's ability to capture all positive instances.

$$Recall = \frac{TP}{TP + FN + UP} \tag{3}$$

where FN is false negative and UP is unclassified positive. F1-score is the harmonic mean of precision and recall, providing a balanced measure that considers both false positives and false negatives.

$$F1 - score = \frac{2 \times Precision \times Recall}{Precision + Recall} \tag{4}$$

It provides a single metric that balances precision and recall, especially useful in situa -tions with class imbalance. AUC represents the area under the Receiver Operating Characteristic (ROC) curve, measuring the ability of the classifier to distinguish betwe -en positive and negative instances. The ROC curve plots the True Positive Rate (TPR) against the False Positive Rate (FPR) at various threshold settings.

$$AUC = \int_0^1 TPR\,(t)\, dFPR\,(t) \tag{5}$$

AUC is unaffected by class imbalance and provides a comprehensive performance mea -sure of overall classification thresholds.

Results

Results on test dataset

On the test set, we used a model that combines Half-life Encoding with Convolutional Neural Network (CNN) architecture and achieved satisfactory results. Specifically, the model achieves an F1-score of 0.8199 and an AUC of 0.8965 on the test set, which indi -cates that the model has strong classification performance and high prediction accuracy. In addition, the accuracy, precision and recall of the model are 0.8212, 0.8275 and 0.8125, respectively, and these results further validate the stability and efficiency of the model.

By comparing with other models we constructed (Table. 1), it can be seen that the CNN model with half-life coding outperforms other models in all the indicators. This model not only achieves the best results in terms of performance, but also its training time is much shorter than the CNN model without half-life coding. Specifically, the training time for the CNN model with half-life coding was approximately 4 hours and 10 minutes (including the half-life coding step), while the CNN model without half-life coding took approximately 4 hours and 50 minutes to complete training. In contrast, the training time of the Autoencoder and XGBoost-based models also greatly exceeds

Table 1. Precision, Recall, F1-score, AUC obtained using four different models for the prediction of gene expression on test dataset.

model	Precision	Recall	F1 Score	AUC
CNNWithHFEncode[1]	0.8275	0.8125	0.8199	0.8965
CNNWithOnlyHF[2]	0.6718	0.6149	0.6421	0.7336
CNNWithOnlyDNA[3]	0.7977	0.8266	0.8119	0.8793
CNNOnly[4]	0.8004	0.8488	0.8239	0.8822
AE+XGBoost	0.7405	0.8226	0.7794	0.7666
LSTM	0.7545	0.7560	0.7533	0.8233

1. Using DNA promoter sequences and coding half-life data as inputs
2. Use non-half-life encoding for unique inputs
3. Use DNA promoter sequences as unique inputs
4. Using DNA promoter sequences and non-coding half-life data as inputs

the training time of our optimal model, further demonstrating the efficiency of the half-life coding + CNN architecture.

Although the F1-score of our half-life coding CNN model is slightly lower than that of the CNN model without coding on the test set, we printed the confusion matrix and compared the accuracy of both models. The accuracy of the CNN model with half-life coding was 0.8212, while the accuracy of the model without coding was 0.8182. This indicates that, despite a marginally lower F1-score, the half-life coding CNN model sli -ghtly outperforms the model without coding in terms of overall accuracy.

Meanwhile, in order to explore whether a single sequence input or half-life data input can lead to better prediction results, we also trained two independent CNN-based models using half-life data and sequence data as the single input. Specifically, the model using half-life data as the only input achieved an F1-score of 0.6660 and an AUC of 0.7291 on the test set, suggesting that although the half-life data provides valuable inf -ormation to the model, the model using these data alone has relatively limited effect and low performance. The model using sequence data as the only input performs much better on the test set, obtaining an F1-score of 0.8119 and an AUC of 0.8793, which indicates that the model with sequence data as the input is significantly better than the model using only half-life data in terms of prediction, especially in terms of classifica -tion accuracy and generalization ability.

These results suggest that although half-life data contribute to the predictive ability of the model to some extent, a single sequence data as input can provide richer informa -tion, which significantly improves the performance of the model. In order to further improve the prediction effect, we combined half-life coding and sequence data input. Finally, a hybrid model was constructed with even better performance, achieving the highest F1-score and AUC when considering both the validation and test sets, further validating the effectiveness of multiple data input methods.

Moreover, although the training time of the LSTM model is relatively short, taking only a few minutes, based on the various evaluation metrics we obtained, especially the F1-score, AUC, accuracy, precision, and recall, we believe that the model with half-life coding and CNN architecture not only outperforms the other models, but also excels in terms of training efficiency. Therefore, we can reasonably conclude that the half-life coding + CNN model is not only the best performing model, but also the shortest and most efficient solution in terms of training time.

Results on valid dataset

Table 2. Precision, Recall, F1-score, AUC obtained using four different models for the prediction of gene expression on valid dataset.

model	Precision	Recall	F1 Score	AUC
CNNWithHFEncode	0.8385	0.8182	0.8282	0.8999
CNNWithOnlyHF	0.6952	0.5899	0.6383	0.7245
CNNWithOnlyDNA	0.7591	0.8020	0.7800	0.8579
CNNOnly	0.7676	0.8606	0.8114	0.8662
AE+XGBoost	0.7442	0.8404	0.7894	0.7755
LSTM	0.7162	0.7596	0.7373	0.7861

We tuned the model using the validation set and finally obtained results that were consistent with the test set, both of which simultaneously verified the stability and validity of the model we built. The CNN model based on half-life coding performed equally well on the validation set, achieving the best results (Table. 2). This consistency suggests that our model consistently achieves better performance both on the training set, validation set, and test set.

By comparing the results of the validation and test sets, we can conclude that our model not only performs well on specific datasets, but also has good generalization ability. This is crucial to ensure that the model can run stably in real applications. Thus, the performance of the validation and test sets together demonstrate the adapt -ability and robustness of our CNN model based on half-life coding.

Model Interpretability Analysis

Analysis of First Fully Connected Layer Output We extracted feature results from the first fully-connected layer and visualized these features using the UMAP (Uniform Manifold Approximation and Projection) package (see Fig. 3).UMAP is an effective dimensionality reduction technique that helps us to map high-dimensional features to a low-dimensional space for a more intuitive understanding of data distri -bution and structure.

By analyzing the visualization results, the data distribution in the figure clearly shows that the data points in the feature space have been effectively partitioned after model training. The samples of each category form different clusters in the low-dimen -sional space, which indicates that the model has successfully extracted distinguishing features and achieved good separation of data in the feature space.

This clear separation indicates that the model has learned an effective feature repre -sentation during the training process, which lays a solid foundation for subsequent classification tasks. With these well-separated features, the classifier is able to recognize and distinguish different classes of samples more easily, thus improving the classification performance of the model.

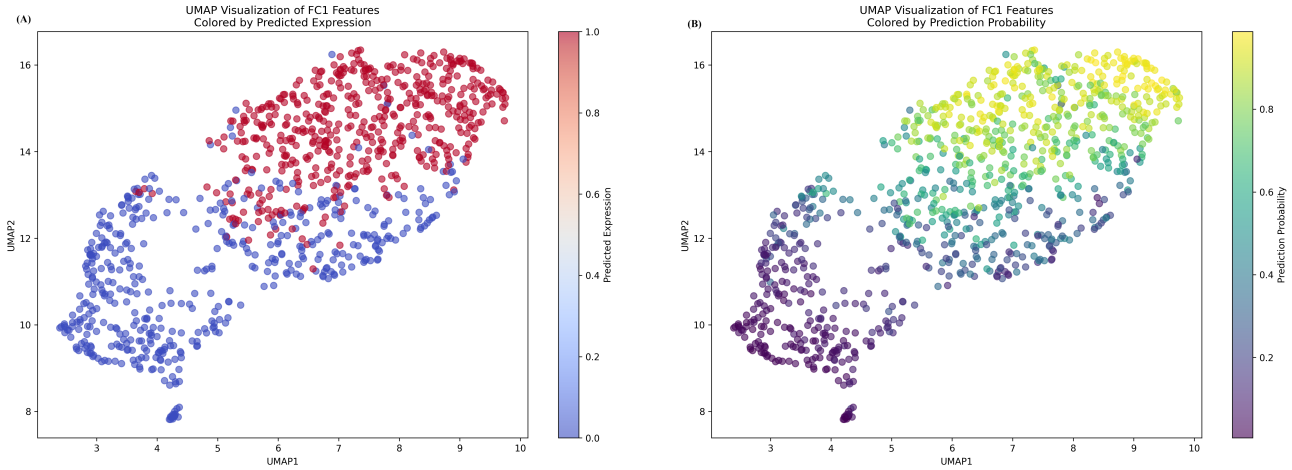

Figure 3. Visualizing First Fully Connected Layer Output with UMAP (A) Based on the final predictions, we labeled the points in the UMAP plot. Specifically, red points represent genes that were predicted to be highly expressed by the model, while purple points represent genes that were predicted to be lowly expressed. (B) Based on the probabilities after the final processing by the sigmoid function, we labeled the points in the UMAP plot. Specifically, the color shades of the points in the graph reflect the model's predicted probability for each sample, with low to high showing the probability that the sample is predicted to be a highly expressed gene. Points for lowly expressed genes have a lower predicted probability and are therefore presented as a lighter color in the graph. Points with highly expressed genes have a higher predicted probability and are usually shown as darker colors.

Analysis of Base Features Associated with High Expression Genes We calculated the feature importance for each nucleotide within the promoter sequences using a dataset of 990 samples from the test set. The feature importance was evaluated through a gradient-based approach, which quantifies the sensitivity of the model's predictions to individual features. Specifically, the importance of each base is measured by the gradient with respect to the model's output. A larger gradient value indicates a stronger influence of that particular feature on the model's predictions, suggesting that the corresponding nucleotide plays a more significant role in the overall prediction acc -uracy. This method provides insights into the contribution of each nucleotide in the promoter region to the expression levels predicted by the model, and can be used to identify key regulatory elements that influence gene expression.

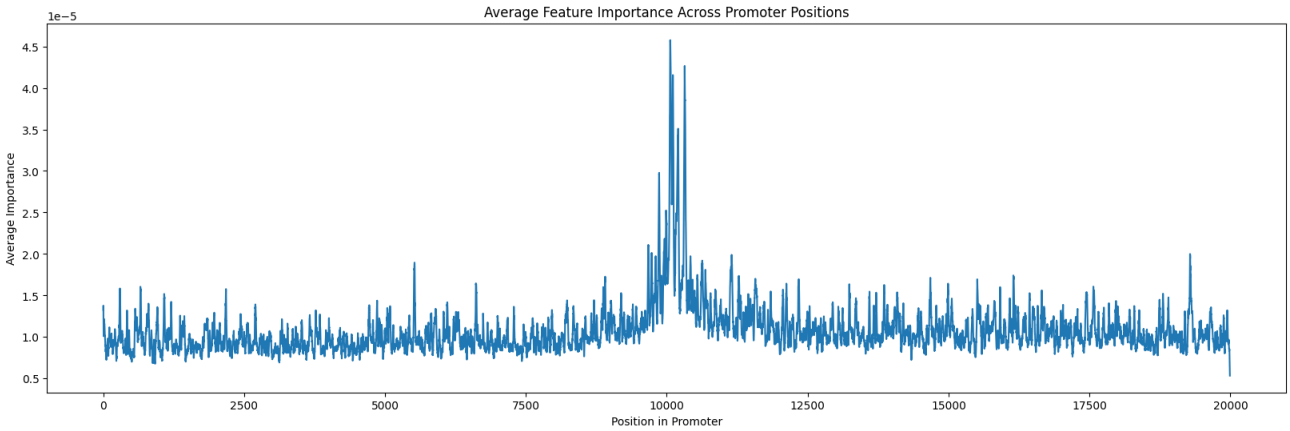

Figure 4. Calculating Feature Importance Using a Gradient-Based Approach

As can be seen in Figure 4, most of the bases that end up contributing significantly to highly expressed genes are concentrated in the middle part of the promoter region. This finding is consistent with the range of sequence data we used (TSS ± 10,000 bp). Specifically, bases located not far before and after the transcription start site (TSS) appear to play a more significant role in the high expression levels of genes. This sugg -ests that changes in bases in the promoter region, especially close to the transcrip -tion start site, may be more sensitive to the regulation of gene expression and may play a greater regulatory role.

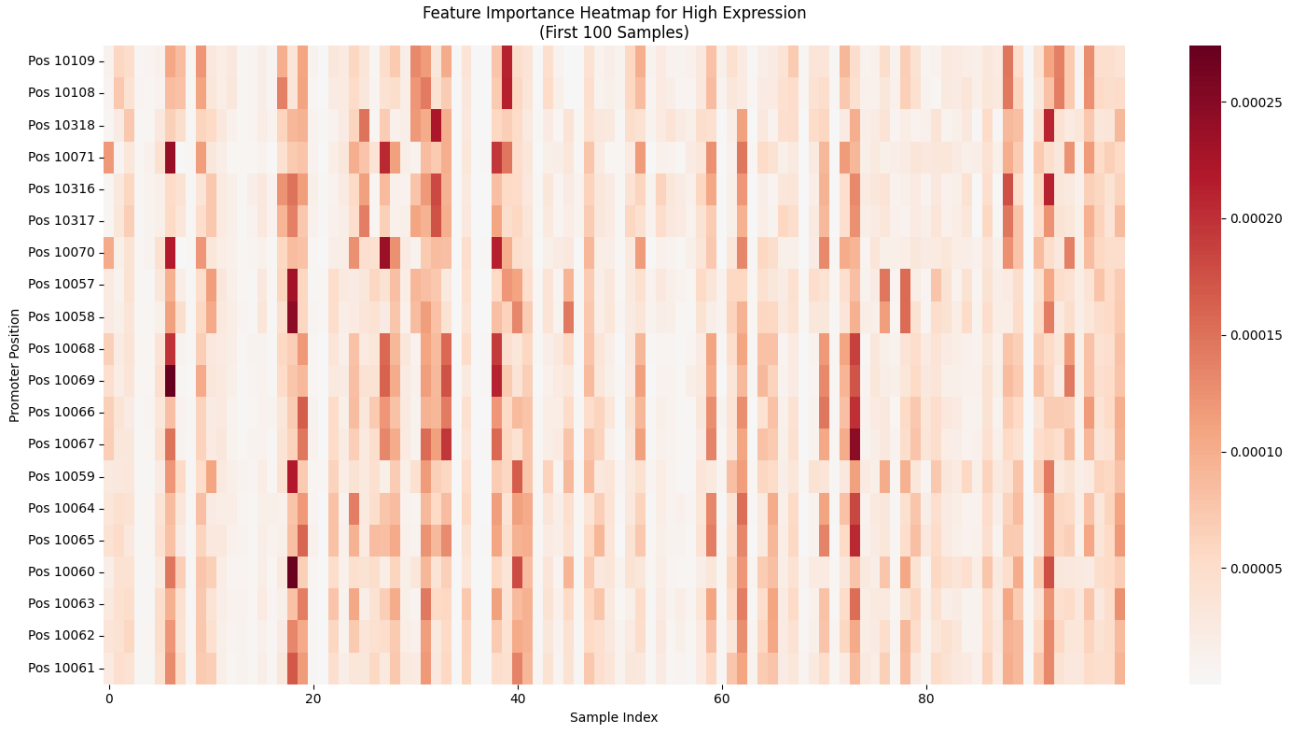

Figure 5. Positions of the top 20 bases contributing most to high expression

In Figure 5, we have extracted the top 20 base positions that contribute most to

high gene expression from 100 samples, showing their importance in each sample. The color shades in the heatmap indicate the degree of contribution of each base position, with the darker red color indicating the greater contribution of the base in a particular sample. This analysis helped us identify which base positions had the greatest impact on highly expressed genes between samples, further revealing the potential role and importance of these key positions in regulating gene expression.

Conclusion

In the competition, we propose a model based on half-life coding and convolutional neural network (CNN) architecture for predicting the high or low expression state of a gene and compare it with all other constructed models. The experimental results show that the model performs best in both accuracy and training efficiency, significantly out -performing other models. To verify the efficiency of the model, we extracted the output of the first fully connected layer and downscaled it using the UMAP (Unified Streamform Approximation and Projection) method. The results after dimensionality reduction show that most of the sample points are successfully separated in the low -dimensional space, further demonstrating that the model is able to efficiently learn and differentiate the differences between highly expressed and low-expressed genes.

In order to gain a deeper understanding of how the model works, we further visuali -zed the key bases identified by the model and analyzed their contribution to high gene expression. The results show that the model pays particular attention to the bases be -fore and after the transcription start site (TSS), and the contribution of these positi -ons is the largest, which is consistent with the biologically known laws of gene expres -sion regulation, suggesting that the model is able to efficiently identify the functional regions that are closely related to gene expression.

By combining half-life coding and deep learning architectures, our model demonstr -ates a strong capability in gene prediction tasks with high generalization ability and biological interpretability.

References

1. M. Poli, S. Massaroli, E. Nguyen, D. Y. Fu, T. Dao, S. Baccus, Y. Bengio, S. Ermon, and C. Ré. Hyena hierarchy: Towards larger convolutional language models. In *International Conference on Machine Learning*, pages 28043–28078. PMLR, 2023.

2. R. Yacouby and D. Axman. Probabilistic extension of precision, recall, and f1 score for more thorough evaluation of classification models. In *Proceedings of the first workshop on evaluation and comparison of NLP systems*, pages 79–91, 2020.

3. M. Zampieri. Multi-label classification of computed tomography scan reports. Master's thesis, SISSA, 2019.