Briefings in Bioinformatics, 00(00), 2020, 1–7

doi: 10.1093/bib/bbaa254
Problem Solving Protocol

Predicting enhancer-promoter interactions by deep learning and matching heuristic

Xiaoping Min †, Congmin Ye†, Xiangrong Liu and Xiangxiang Zeng

Corresponding author: Xiangxiang Zeng, Department of Computer Science, Hunan University, 2 Lushan S Rd, Yuelu District, 410086 Changsha, China. E-mail: xzeng@hnu.edu.cn
†These authors are joint first authors in this work.

Abstract

Enhancer-promoter interactions (EPIs) play an important role in transcriptional regulation. Recently, machine learning-based methods have been widely used in the genome-scale identification of EPIs due to their promising predictive performance. In this paper, we propose a novel method, termed EPI-DLMH, for predicting EPIs with the use of DNA sequences only. EPI-DLMH consists of three major steps. First, a two-layer convolutional neural network is used to learn local features, and an bidirectional gated recurrent unit network is used to capture long-range dependencies on the sequences of promoters and enhancers. Second, an attention mechanism is used for focusing on relatively important features. Finally, a matching heuristic mechanism is introduced for the exploration of the interaction between enhancers and promoters. We use benchmark datasets in evaluating and comparing the proposed method with existing methods. Comparative results show that our model is superior to currently existing models in multiple cell lines. Specifically, we found that the matching heuristic mechanism introduced into the proposed model mainly contributes to the improvement of performance in terms of overall accuracy. Additionally, compared with existing models, our model is more efficient with regard to computational speed.

Key words: enhancer-promoter interactions; pretraining; deep learning; matching heuristic; DNA sequence

Introduction

The three-dimensional (3D) chromatin organization itself was recently found to play an important role in transcription regulation [1, 2]. Recognizing enhancer-promoter interactions (EPIs) is an effective means for understanding gene regulation and grasping the mechanism of a disease [3]. For example, EPIs may be useful in linking risk loci from genome-wide association studies with their target genes and understanding the pathogenesis of complex diseases. Thus, the identification of true 3D genome organization, especially EPIs across different cell lines, is an important step toward in-depth understanding of gene regulation and disease mechanisms.

In the last few decades, the recognition of EPIs relies on high-throughput experimental techniques. For example, Hi-C [4] has made the genome-wide detection of EPIs possible, but the major problem of the technique is that the resolution is sufficiently high for precisely capturing individual EPIs. ChIA-PET [5], also used for the genome-scale identification of genomic interactions, is restricted to the detection of interactions that are mediated by a preselected protein of interest. Additionally, these high-throughput experimental methods have some intrinsic shortcomings, that is, they are time-consuming and expensive.

For these problems, machine learning methods [6] have been extensively in the identification of EPIs on the basis of genomic sequences, and significant progress has been made. The use of deep learning in making EPI predictions with the use of DNA sequences only has been proposed, and this method has shown

Xiaoping Min is an associate professor in Xiamen University. His research interests include bioinformatics and data mining.
Congmin Ye is a graduate student in Xiamen University. Her research interest includes bioinformatics.
Xiangrong Liu is a professor in Xiamen University. His research interest includes in the classification of proteins in bioinformatics.
Xiangxiang Zeng is a professor in Hunan University. His research interests include biocomputing and bioinformatics.
Submitted: 13 July 2020; **Received (in revised form):** 8 September 2020

great potential in the high-throughput prediction of large-scale genomes. For instance, SPEID [7] is the first deep learning architecture for training the hybrid models of convolution neural networks (CNNs) and recurrent neural networks for the prediction of EPIs. SIMCNN [8], a CNN-based predictive model, can have a similar level of predictive performance as SPEID but has a considerably higher training efficiency. EPIANN [9] introduces an attention mechanism for constructing a network, further improving predictive performance. It provides single and pairwise importance scores for the positions of enhancer and promoter regions, making a detailed analysis of feature importance feasible.

Although these deep learning models have achieved substantial progress, they still have some limitations [10–12]. First, most of the deep learning models use one-hot representations to convert query promoters and enhancers into the inputs of networks, and this type of representation of sequential characteristics is flawed with the potential for the curse of dimensionality [13]. Second, existing model architectures are extremely simple and thus cannot learn discriminative features during feature extraction. Third, in most of these models, the learned features of promoters and enhancers are directly concatenated for subsequent prediction, and thus potential interaction information is lost. Lastly, training efficiency is relatively low because of the numerous parameters in existing deep learning models.

To address these limitations, we proposed EPI-DLMH, a novel deep learning predictive framework. The main contributions of this paper are summarized as follows:

(i) We proposed a novel deep network model to capture latent structural information between promoters and enhancers. The experimental results showed that our model is superior to existing models in terms of overall predictive performance and training speed.
(ii) We used matching heuristics to capture interaction information between promoters and enhancers. Our results showed that interaction information is useful in improving predictive performance.

Methods

Data

In this study, we used the benchmark dataset derived from EPIANN. The dataset contains six different cell lines, namely, GM12878, HeLa-S3, IMR90, K562, HUVEC and NHEK. Each cell line roughly has 1500 positive samples (true EPIs) and 30 000 negative samples (non-EPIs) on average. Information on the dataset of each cell line is presented in Table 1. The dataset of each cell line is highly imbalanced, and the ratio of positive to negative samples is approximately 1:20. Imbalance data would influence the traditional classifier and make the performance poor [14–20]. To solve the data imbalanced problem, EPIANN augmented the number of positive samples by using the upstream and downstream regions of enhancers (positives). Relevant regions around enhancers and promoters were included by setting the length of the extended window at 3 kbps for enhancers and 2 kbps for promoters.

Model structure

We proposed a novel predictive framework that can automatically predict EPIs with the use of their DNA sequences only. The procedure of the proposed predictive framework is illustrated in Figure 1. The four main steps in the framework are sequence embedding, feature extraction, matching heuristic and prediction. First, given that the enhancers and promoters are inputs, they are embedded as feature matrixes that are then fed into a hybrid multilayer neural network for the learning of high-latent features. The two learned feature vectors are connected using different matching heuristics and ultimately placed into a prediction layer. This procedure is performed for determining whether query enhancers interact with promoters. In the following subsections, we introduce the proposed framework step by step. The source code is available at https://github.com/Xzenglab/EPI-DLMH.

Table 1. Summary of the original dataset in six cell lines

Cell lines	Positive samples	Negative samples
GM12878	2113	42 200
HUVEC	1524	30 400
HeLa-S3	1740	34 800
IMR90	1254	25 000
K562	1977	39 500
NHEK	1291	25 600

Sequence embedding

The usage of k-mer representation is an effective method in long DNA fragment analysis. In this work, according to k-mer representation, we split the promoters and enhancers by using a k-bp window with s as the sliding step size. It is mentioned in the literature [21] that PEP-WORD verified in the experiment that when $k = 6$, the computational efficiency of the model and the information complexity of the vectors are most suitable. Likewise, we set k at 6 and s at 1 in our study. For example, 'AGCTGTTC' is accordingly split into 'AGCTGT,' 'GCTGTT' and 'CTGTTC' through k-mer representation. Obviously, the k-mer representation is simple to understand and compute. Unfortunately, its straightforward vector encoding as a one-hot vector (i.e. bit vector that consists of zeros except in a single dimension) is vulnerable to the curse of dimensionality. Specifically, its one-hot vector has a dimension exponential to the length of k. A 6-mer needs a bit vector of dimension 4^6 of 4096. However, this is problematic when applying the latest machine learning algorithms to solve problems in biological sequence analysis, due to the fact that most of these tools prefer lower-dimensional continuous vectors as input [22–24]. Further, using a one-hot vector makes the distances among different k-mer words equal, and thus correlation information among k-mer words can be ignored.

Therefore, we represent the sequences with k-mer words through dna2vec embedding [13, 25]. Dna2vec embedding is based on the popular word embedding model word2vec [26], which is trained on a shallow two-layer neural network. Dna2vec can obtain low-dimensional and high-quality vectors to represent k-mer words. In our experimental setting, dna2vec was first pretrained with hg38 human assembly chr1 to chr22 and then fine-tuned using our datasets to adapt to our prediction tasks. By doing so, we were able to obtain a 100-dimensional word vector that represented all 6-mer words. To this end, the promoter and enhancer sequences in our datasets were encoded as 100*2000-D matrix and 100*3000-D matrix.

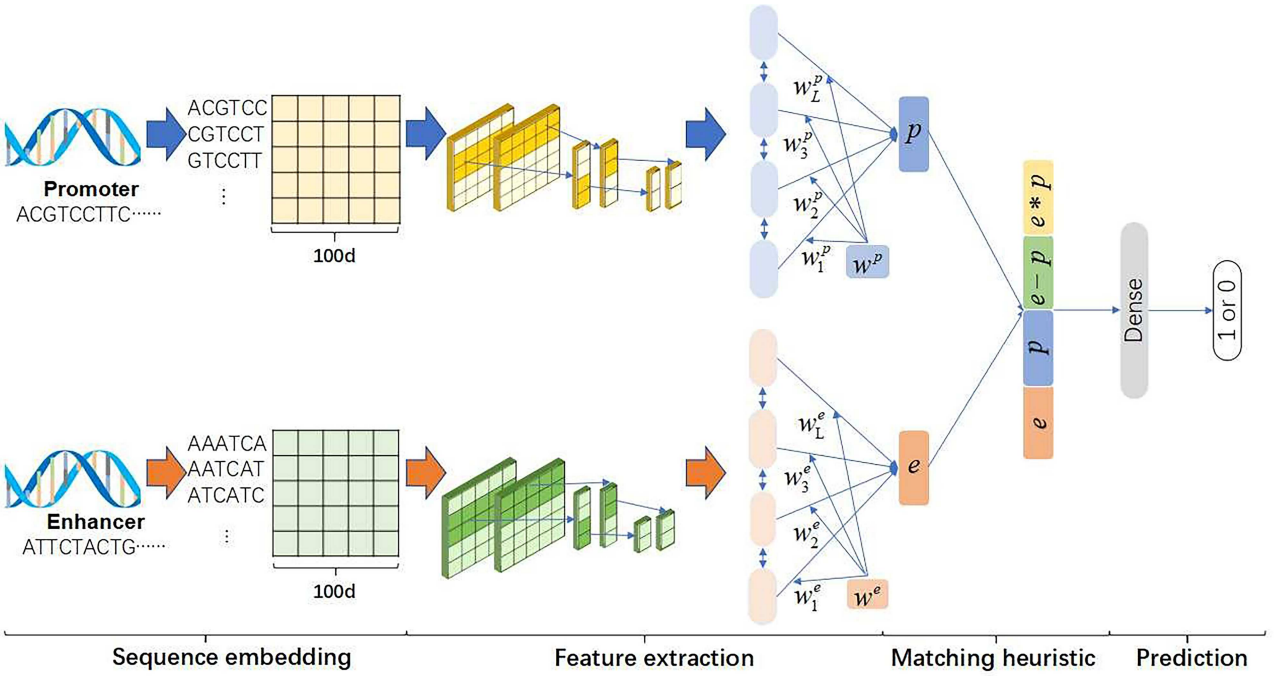

Figure 1. Architecture of the model. It consists of four steps, including sequence embedding, feature extraction, matching heuristic and prediction.

Feature extraction

In our model, we used a hybrid network structure containing a CNN [27, 28] and bidirectional gated recurrent unit (Bi-GRU). CNN was used to learn the local features of promoters and enhancers, whereas Bi-GRU was used to capture the long-term dependencies of local features. Moreover, an attention layer was added for the calculation of important features that were given a larger weight to represent feature vectors.

Convolution neural network. We first constructed a two-layer CNN consisting of a convolution layer and a max-pooling layer. The convolution layer is mainly used to learn the local features of enhancers and promoters, and the max-pooling layer reduces feature dimension. In our experimental setting, we built a CNN for enhancers and another for promoters. For the enhancers, we set the filter length of the convolution layer at 60, the number of filters at 64, the pooling length of max-pooling layer at 30, and the stride at 30. For the promoters, we set the filter length of the convolution layer at 40, the number of filters at 64, the pooling length of max-pooling layer at 20, and the stride at 20. The hyperparameter setting is consistent with the comparison model (SPEID [7] and SIMCNN [8]).

Bidirectional gated recurrent unit. The resulting local features yielded by the CNN above were then fed to Bi-GRU, a bidirectional network that can capture bidirectional semantic dependencies. Here, a Bi-GRU layer was used to learn the long-term dependencies of local features. Specifically, the Bi-GRU layer had two parts, which simultaneously read the features from the forward and reverse directions. The update process of GRU is as follows:

$$r_t = \sigma\left(W_r x_t + U_r h_{t-1} + b_r\right) \tag{1}$$

$$z_t = \sigma\left(w_z x_t + U_z h_{t-1} + b_z\right) \tag{2}$$

$$\overline{h}_t = \tanh\left(W_h x_t + U_h\left(r_t \cdot h_{t-1}\right) + b_h\right) \tag{3}$$

$$h_t = (1 - z_t) \cdot \overline{h}_t + z_t \cdot h_{t-1} \tag{4}$$

where tanh is the hyperbolic tangent function, σ is the sigmoid function, U is the weight matrixes for the previous hidden state vector h_{t-1}, x_t is the input at t time, $\overline{h}_t$ is a candidate activation, $\cdot$ is an element-wise multiplication, r_t and z_t denote the reset gate and renew gate vector, respectively. Bi-GRU generates the forward sequence $\left\{\overrightarrow{h_1}, \overrightarrow{h_2}, \cdots, \overrightarrow{h_t}, \cdots, \overrightarrow{h_L}\right\}$ from left to right, and the reverse sequence $\left\{\overleftarrow{h_1}, \overleftarrow{h_2}, \cdots, \overleftarrow{h_t}, \cdots, \overleftarrow{h_L}\right\}$ is generated from right to left. Finally, the forward and reverse states are combined.

$$h_t = \left[\overrightarrow{h_t}, \overleftarrow{h_t}\right] \tag{5}$$

We used different Bi-GRU layers with 50 output units for enhancers and promoters.

Attention layer. Attention layer was used to learn the importance of features, providing more important features with more weights to represent the sequence.

$$w_k^p = \frac{\exp\left(\rho_{kp}^T b_\varepsilon\right)}{\sum_{k=1}^{L} \exp\left(\rho_{kp}^T b_\varepsilon\right)} \tag{6}$$

$$w_k^e = \frac{\exp\left(\rho_{ke}^T b_\varepsilon\right)}{\sum_{k=1}^{L} \exp\left(\rho_{ke}^T b_\varepsilon\right)} \tag{7}$$

$$p = \sum_{k=1}^{L} w_k^p * h_k^p \tag{8}$$

$$e = \sum_{k=1}^{L} w_k^e * h_k^e \tag{9}$$

where ρ_{kp}^T and ρ_{ke}^T are hidden representations of the kth feature of promoter and enhancer, respectively. b_ε is stands for context vector. w_k^p and w_k^e are the importance of h_k^p and h_k^e, h_k^p and h_k^e are the outputs of the promoter and enhancer at k time in the Bi-GRU layer, and e and p represent the feature vectors of the enhancer

and promoter. Here, we used different attention layers with 50 output units for the enhancers and promoters.

Matching heuristics

To capture more interaction information between promoters and enhancers, we introduced an additional matching heuristics, which has been widely used in various advanced models in Natural Language Inference [29–31]. In the prediction of EPIs, we used matching heuristics to capture the relationship between promoters and enhancers. In our model, we introduced three matching heuristics: (1) concatenation of enhancers and promoters, (2) element-wise difference and (3) element-wise product.

The first matching heuristics were used to concatenate the features of the promoters and enhancers, and the others were used to calculate the 'closeness' and 'similarity' of the features between promoters and enhancers. The three matching heuristics are formulated by

$$m = [\, e, p, e - p, e \; o \; p \,] \tag{8}$$

where e and p are the feature vectors of the promoters and enhancers, respectively, - denotes element-wise difference, o denotes element-wise product, and m is the final feature vector.

Model training setup

To compare our model with other existing models (SPEID, SIMCNN, EPIANN), we used the same dataset and followed the same training procedure in constructing our deep learning model. For any given cell line, the training procedure was described as follows.

(i) Start from the imbalanced data D.
(ii) Split D into a training set D_{train} (90% of D) and a test set D_{test} (10% of D) by stratified sampling.
(iii) Augment D_{train} to get a balanced dataset D_{aug}.
(iv) Train the model on D_{aug}.
(v) Evaluate the model on D_{test}.

We used the Glorot uniform initializer to initialize weights for each CNN model [32]. To prevent overfitting, we added dropout and L2 regularization to the model. Our models were trained in mini-batches 32 samples through back-propagation with binary cross-entropy loss and minimized using the Adam algorithm [33]. In each cell line, our model trains 90 epochs, consistent with several existing methods, such as SPEID, SIMCNN and EPIANN.

Results and discussions

Matching heuristics improve predictive performance

In this section, we determined whether the additional matching heuristics layer can improve performance. We evaluated and compared our predictive models with different matching heuristics. Figures 2–4 present the AUROCs, AUPRCs and F1 score of the models based on different matching heuristics, respectively. The additional matching heuristics are effective in improving predictive performance in most cases (multiple cell lines). For example, in the GM12878 cell line, the AUROC of EPI-DLMH (cat, -, o) is 1.1% higher than that of EPI-DLMH (cat). In the HUVEC cell lines, the AUPRC value of EPI-DLMH (cat, -) is roughly 1% higher than that of EPI-DLMH (cat). In the NHEK cell lines, the F1 score of EPI-DLMH (cat, -) is 6.4% higher than that of EPI-DLMH (cat). However, in some of the other cell lines, the performance of matching heuristics has no improvement. For example, in the K562 cell line, the AUPRC of EPI-DLMH (cat, -, o) is lower than that of EPI-DLMH (cat) and the F1 score of EPI-DLMH (cat, -) is lower than that of EPI-DLMH (cat). According to our experiment, matching heuristics (cat, -) or (cat, -, o) are the best in most cases. For a specific cell line, different matching heuristics generally exhibit different performance and can outperform a model trained with concatenated features. Thus, the matching heuristics we introduced is effective in capturing interactive information, thereby improving performance.

Performance comparison of our proposed method and existing methods

To evaluate the effectiveness of our predictive method, we compared it with three state-of-the-art deep learning models, namely, SPEID, SIMCNN and EPIANN, and used a single benchmark dataset with six cell lines. We presented a predictive performance in terms of AUROC, AUPRC and F1 score in Tables 2–4, respectively.

Table 2 shows the performance of our model compared with those of other exiting models on AUROC. EPI-DLMH performs better than the other models (SPEID, SIMCNN and EPIANN) in each cell line. Specifically, the AUROCs of our model are 1.2, 2.4, 1.1, 0.2, 0.4 and 0.8% higher than the runner-up predictors on GM12878, HUVEC, HeLa-S3, IMR90, K562 and NHEK, respectively. Our model shows greater improvement in GM12878, HUVEC and HeLa-S3 and slight improvement in IMR90, K562 and NHEK.

Table 3 shows the performance of our model compared with those of the other models in terms of AUPRC. Specifically, our model is 2.3% for GM12878, 1.8% for HOVEC, 3.3% for Hela-S3, 4.5% for IMR90, 2.1% for K562 and 0.4% for NHEK higher than the runner-up predictors. Although our model shows a slight improvement in AUROC (0.2% and 0.4%) in the IMR90 and K562 lines, the improvement in AUPRC (4.5% and 2.1%) was greater in these two cell lines. Table 4 shows the performance of our model compared with those of the other models in terms of F1 score. Specifically, our model is 0.2% for GM12878, 1% for HOVEC, 6.2% for Hela-S3, 4.7% for IMR90, 6.3% for K562 and 1.1% for NHEK higher than the runner-up predictors. Our EPI-DLMH exhibits better performance than any of the other models in each cell line. In summary, our model outperforms state-of-the-art predictors robustly in different cell lines. The possible reason is that the sequence features that our model learn are more informative than those of the other models, especially when a matching heuristic mechanism is applied, which improves the prediction of EPIs.

Moreover, we further compared the running times of existing models with the running time of our proposed method. In our experiment, we used a 1080 GPU to train all the models. In training, times were roughly 180 h (SPEID), 120 h (SIMCNN) and 240 h (EPIANN), which are considerably longer than the running time of our model (72 h). The reason is that we used a lower parameter number for our deep learning model training. The results demonstrate that our model is more efficient. Overall, our model is more effective and efficient than the other existing models.

Conclusion

We proposed a novel deep learning model for predicting EPIs with DNA sequences only. Benchmarking comparison results indicate that in multiple human cell lines, our proposed method

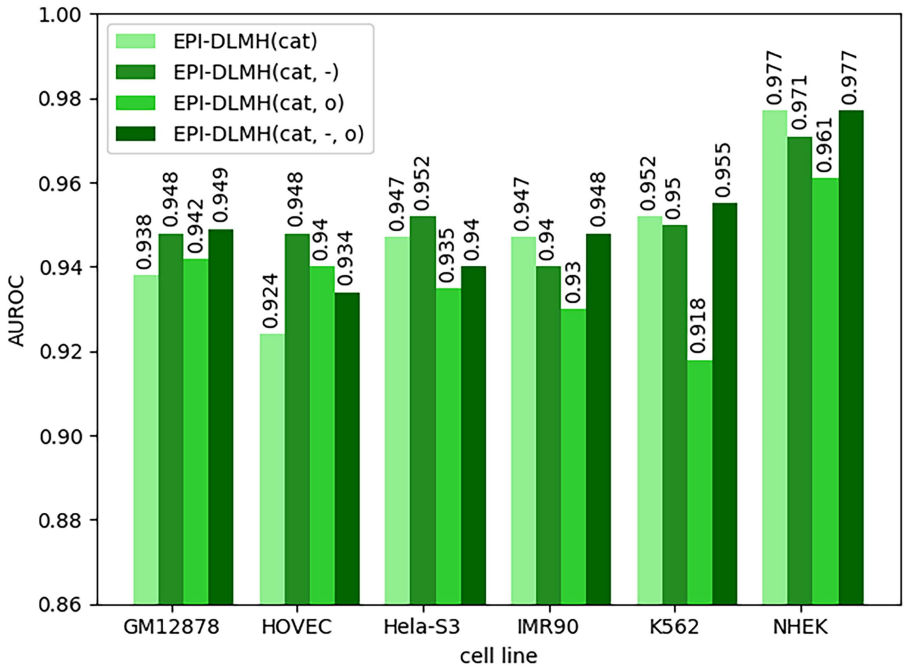

Figure 2. AUROC of the variants of our model using different matching heuristics. 'Cat' refers to the feature concatenation of enhancers and promoters; '-' and 'o' denote element-wise difference and product, respectively.

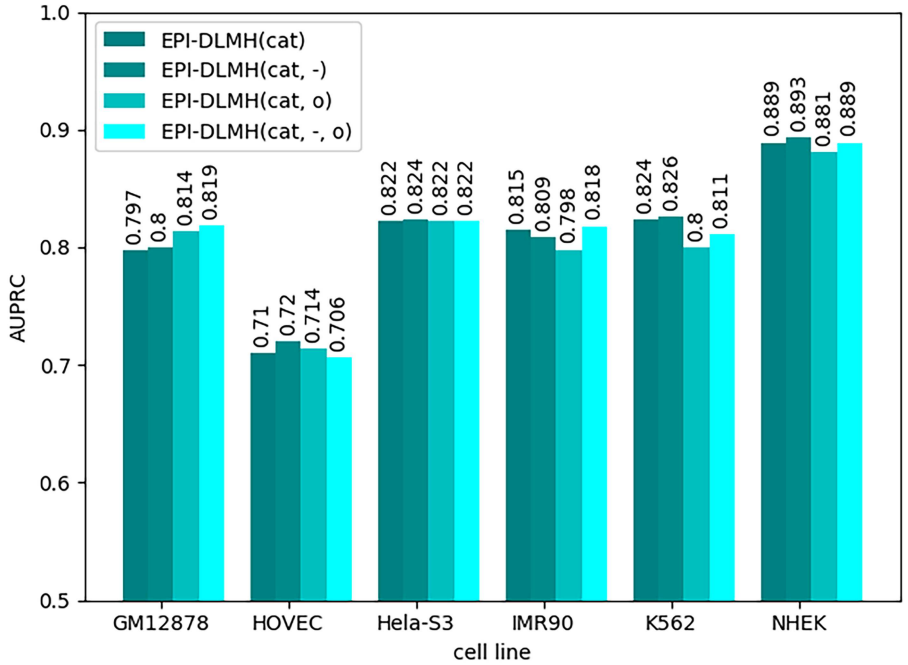

Figure 3. AUPRC of the variants of our model using different matching heuristics. 'Cat' refers to the feature concatenation of enhancers and promoters; '-' and 'o' denote element-wise difference and product, respectively.

Table 2. AUROC of different EPI prediction methods for six cell lines

	GM12878	HOVEC	Hela-S3	IMR90	K562	NHEK
SPEID	0.937	0.855	0.847	0.879	0.937	0.969
SIMCNN	0.937	0.924	0.941	0.946	0.951	0.967
EPIANN	0.919	0.924	0.918	0.945	0.943	0.959
EPI-DLMH(best)	**0.949**	**0.948**	**0.952**	**0.948**	**0.955**	**0.977**

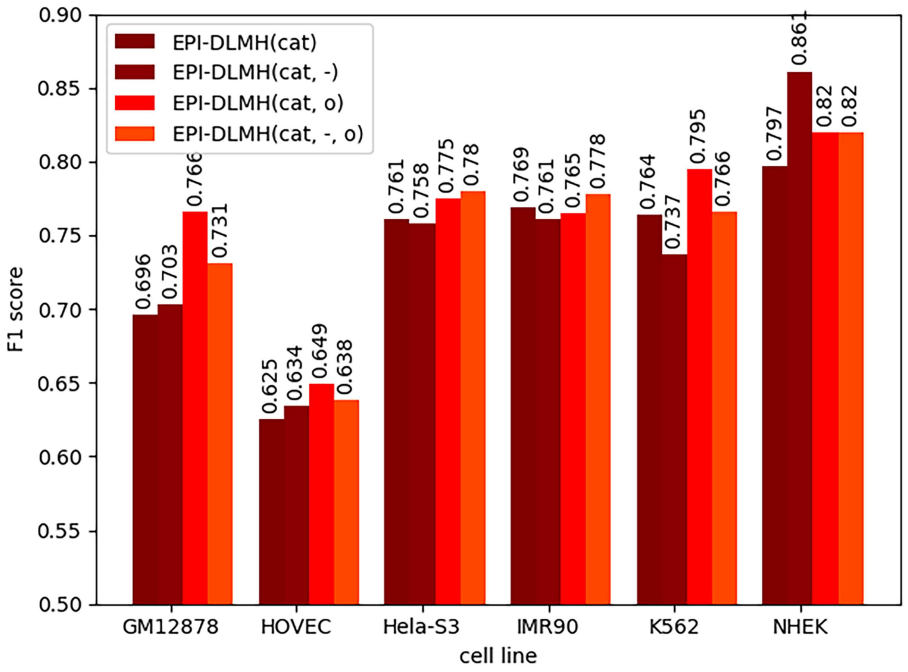

Figure 4. F1 score of the variants of our model using different matching heuristics.
'cat' refers to the feature concatenation of enhancers and promoters; '-' and 'o' denote element-wise difference and product, respectively.

Table 3. AUPRC of different EPI prediction methods for six cell lines

	GM12878	HOVEC	Hela-S3	IMR90	K562	NHEK
SPEID	0.796	0.606	0.733	0.753	0.780	0.889
SIMCNN	0.786	0.642	0.791	0.773	0.805	0.887
EPIANN	0.723	0.702	0.616	0.770	0.673	0.861
EPI-DLMH(best)	**0.819**	**0.720**	**0.824**	**0.818**	**0.826**	**0.893**

Table 4. F1 score of different EPI prediction methods for six cell lines

	GM12878	HOVEC	Hela-S3	IMR90	K562	NHEK
SPEID	0.764	0.535	0.718	0.731	0.692	0.734
SIMCNN	0.726	0.541	0.652	0.709	0.732	0.85
EPIANN	0.699	0.639	0.59	0.711	0.626	0.797
EPI-DLMH(best)	**0.766**	**0.649**	**0.78**	**0.778**	**0.795**	**0.861**

improves performance in terms of AUROC, AUPRC and F1 score compared with existing predictive methods. Through further analysis, we found that our matching heuristics mainly contribute to performance improvement. The matching heuristics we introduced to our model is capable of capturing the intrinsic interaction information of promoters and enhancers. Moreover, compared with existing deep learning models, our model has a considerably faster training speed and thus has potential as a useful tool for fast EPI prediction in genome-wide applications.

Key Point

- Propose a novel method for predicting EPIs with the use of DNA sequences only.
- Introduce matching heuristic mechanism to improve the effectiveness of the model.
- Compared with existing models, our model is more efficient with regard to computational speed.

Funding

The work was supported by the National Natural Science Foundation of China (Grant Nos. 61472333, 61772441, 61472335, 61272152, and 41476118); Project of Marine Economic In-Novation and Development in Xiamen (No. 16PFW034SF02); Natural Science Foundation of the Higher Education Institutions of Fujian Province (No. JZ160400); Natural Science Foundation of Fujian Province (No. 2017 J01099); and President Fund of Xiamen University (No. 20720170054). X. Zeng is supported by Juan de la Cierva position (code: IJCI-2015-26991).

References

1. Zhang W, Li W, Zhang J, *et al*. Data integration of hybrid microarray and single cell expression data to enhance gene network inference. *Curr Bioinf* 2019;**14**(3):255–68.
2. Krivega I, Dean A. Enhancer and promoter interactions—long distance calls. *Curr Opin Genet Dev* 2012;**22**(2):79–85.
3. Li G. X Ruan, RK Auerbach, et al. extensive promoter-centered chromatin interactions provide a topological basis for transcription regulation. *Cell* 2012;**148**(1–2):84–98.
4. Dryden NH, Broome LR, Dudbridge F, *et al*. Unbiased analysis of potential targets of breast cancer susceptibility loci by capture hi-C. *Genome Res* 2014;**24**(11):1854–68.
5. Rao SS, Huntley MH, Durand NC, *et al*. A 3D map of the human genome at kilobase resolution reveals principles of chromatin looping. *Cell* 2014;**159**(7):1665–80.
6. Bu HD, Hao J, Guan J, *et al*. Predicting enhancers from multiple cell lines and tissues across different developmental stages based on SVM method. *Curr Bioinf* 2018;**13**(6):655–60.
7. Singh S, Yang Y, Póczos B, *et al*. Predicting enhancer-promoter interaction from genomic sequence with deep neural networks. *Quant Biol* 2019;**7**(2):122–37.
8. Zhuang Z, Shen X, Pan W. A simple convolutional neural network for prediction of enhancer–promoter interactions with DNA sequence data. *Bioinformatics* 2019;**35**(17):2899–906.
9. Mao W, Kostka D, Chikina M. Modeling enhancer-promoter interactions with attention-based neural networks. *bioRxiv* 2017;219667.
10. Lv ZB, Ao CY, Zou Q. Protein function prediction: from traditional classifier to deep learning. *Proteomics* 2019;**19**(14):2.
11. Wu B, Zhang H, Lin L, *et al*. A similarity searching system for biological phenotype images using deep convolutional encoder-decoder architecture. *Curr Bioinf* 2019;**14**(7):628–39.
12. Liu B, Li C, Yan K. DeepSVM-fold: protein fold recognition by combining support vector machines and pairwise sequence similarity scores generated by deep learning networks. *Brief Bioinform* 2020;**21**(5):1733–41.
13. Ng P. dna2vec: consistent vector representations of variable-length k-mers. arXiv preprint arXiv. 2017;**1701**:06279.
14. Zou Q, Xie S, Lin Z, *et al*. Finding the best classification threshold in imbalanced classification. *Big Data Res* 2016;**5**:2–8.
15. Fdez-Glez J, Ruano-Ordas D, Mendez JR, *et al*. Determining the inf luence of class imbalance for the triage of biomedical documents. *Curr Bioinf* 2018;**13**(6):592–605.
16. Wan S, Duan Y, Zou Q. HPSLPred: an ensemble multi-label classifier for human protein subcellular location prediction with imbalanced source. *Proteomics* 2017;**17**(17–18):1700262.
17. Song L, Li D, Zeng X, *et al*. nDNA-prot: identification of DNA-binding proteins based on unbalanced classification. *Bmc Bioinf* 2014;**15**(1):298.
18. Shen Y, Ding Y, Tang J, *et al*. Critical evaluation of web-based prediction tools for human protein subcellular localization. *Brief Bioinform* 2020;**21**(5):1628–40.
19. Wang Z, He W, Tang J, *et al*. Identification of highest-affinity binding sites of yeast transcription factor families. *J Chem Inf Model* 2020;**60**(3):1876–83.
20. Liu B, Gao X, Zhang H. BioSeq-Analysis2.0: an updated platform for analyzing DNA, RNA, and protein sequences at sequence level and residue level based on machine learning approaches. *Nucleic Acids Res* 2019;**47**(20):e127.
21. Yang Y, Zhang R, Singh S, *et al*. Exploiting sequence-based features for predicting enhancer–promoter interactions. *Bioinformatics* 2017;**33**(14):i252–60.
22. Suykens JA, Vandewalle JJN. Least squares support vector machine classifiers. *Neural processing letters* 1999;**9**(3):293–300.
23. Angermueller C, Pärnamaa T, Parts L, *et al*. Deep learning for computational biology. *Mol Syst Biol* 2016;**12**(7):878.
24. Turian J, Ratinov L, Bengio Y. Word representations: a simple and general method for semi-supervised learning. in *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics* 2010: 384–94.
25. Zou Q, Xing P, Wei L, *et al*. Gene2vec: gene subsequence embedding for prediction of mammalian N6-Methyladenosine sites from mRNA. *RNA* 2019;**25**(2):205–18.
26. Mikolov T, Chen K, Corrado G, *et al*. Efficient estimation of word representations in vector spacearXiv preprint arXiv:1301.3781. 2013.
27. Ren F, Yang C, Qiu Q, *et al*. Exploiting discriminative regions of brain slices based on 2D CNNs for Alzheimer's disease classification. *IEEE Access* 2019;**7**:181423–33.
28. Li C-C, Liu B. MotifCNN-fold: protein fold recognition based on fold-specific features extracted by motif-based convolutional neural networks. *Brief Bioinform* 2019. doi: 10.1093/bib/bbz133.
29. Conneau A, Kiela D, Schwenk H, *et al*. Supervised learning of universal sentence representations fromnatural language inference dataarXiv preprint arXiv:1705.02364. 2017.
30. Liu Y, Sun C, Lin L, *et al*. Learning natural language inference using bidirectional LSTM model and inner-attentionarXiv preprint arXiv:1605.09090. 2016.
31. Nie Y, Bansal M. Shortcut-stacked sentence encoders for multi-domain inference. arXiv preprint arXiv:1708.02312. 2017.
32. Glorot, X. and Y. Bengio. Understanding the difficulty of training deep feedforward neural networks. in *Proceedings of the Thirteenth International Conference on Artificial Intelligence and Statistics*. 2010: 249–56.
33. Kingma DP, Ba J. Adam: a method for stochastic optimization. arXiv preprint arXiv:1412.6980. 2014.