journal, 2022, pp. 1–9

doi: https://doi.org/10.1101/2022.10.15.512354
Advance Access Publication Date: Day Month Year
Problem Solving Protocol

PROBLEM SOLVING PROTOCOL

EVMP: Enhancing machine learning models for synthetic promoter strength prediction by Extended Vision Mutant Priority framework

Weiqin Yang ,[2,†] Dexin Li [2,†] and Ranran Huang [1,*]

[1] Institute of Marine Science and Technology, Shandong University, 72 Binhai Road, 266237, Qingdao, China and [2] School of Computer Science and Technology, Shandong University, 72 Binhai Road, 266237, Qingdao, China
*Corresponding author: Ranran Huang, Email: huangrr@sdu.edu.cn
[†] The first two authors contribute equally to the paper

Abstract

In metabolic engineering and synthetic biology applications, promoters with appropriate strengths are critical. Promoter traits including excessive sequence length and restricted vocabulary size, are considered to impede the effect of natural language models on tasks involving genetic sequence. We proposed EVMP (Extended Vision Mutant Priority framework), which enhances various machine learning models. The synthetic promoter input to EVMP is split into base promoter and k-mer mutations, which are encoded by BaseEncoder and VarEncoder respectively. We used EVMP on various machine learning models for promoter strength prediction in Trc synthetic promoter library. The MAE (mean absolute error) of LSTM was reduced from 0.50 to 0.18, the MAE of the Transformer was reduced from 0.27 to 0.177, and the MAE of other models was also slightly reduced by EVMP. By virtual dataset expansion based on multiple different base promoters, EVMP utilizes ensemble learning and conclusively lowers MAE. Further investigation demonstrated that EVMP can improve performance of machine learning models by lowering the over-smoothing phenomenon caused by the sequence similarity of synthetic promoters. Additionally, only 56% of the original amount of synthetic promoter dataset is needed to obtain an identical previous result after applying EVMP, which reduces the cost of creating synthetic promoter libraries. Our experiments demonstrated that EVMP is versatile and robust for building synthetic sequence machine learning models. The source code is available at https://github.com/Tiny-Snow/EVMP.

Key words: EVMP, machine learning, promoter strength prediction, deep learning, synthetic promoter mutation library

Introduction

Promoters are the fundamental components of transcriptional regulation and have a direct impact on gene expression. In metabolic engineering and synthetic biology applications, promoters with desired strengths are critical [1]. However, it is challenging for them to meet the requirements of the logical design and optimization of metabolic pathway due to the insufficient number of well-characterized promoters. Therefore, a vast library with hundreds of promoters with better properties must be created and characterized [2].

The conventional methods for building promoter libraries rely on functional module combinations or random sequence mutation techniques Methods based on random sequence mutations, such as error-prone PCR, are regarded as a straightforward and effective mutagenesis technique and have been successfully used to synthesize artificial promoters. Alper et al. [3], for instance, mutagenized the bacteriophage PL-promoter using error-prone PCR methods. Finally, 22 promoters with an intensity distribution range of 196 times were selected from nearly 200 promoter mutants. Zhao et al. [4] constructed and characterized a mutant library of Trc promoters (P_{trc}) using 83 rounds of mutation-construction-screening-characterization engineering cycles, and established a synthetic promoter library that consisted of 3665 different variants, displaying an intensity range of more than two orders of magnitude. Despite the availability of experimental methods, obtaining a small number of useful promoters from a random library typically necessitates time-consuming and laborious screening, and given the enormous number of all possible sequence combinations, the effective identification of these useful promoters is largely constrained by a relatively small mutation library. For example, a 20-nucleotide sequence space represents $4^{20} = 1.1 \times 10^{12}$ possibilities, which exceeds even the largest bacterial libraries and most robust high-throughput screens. Therefore, obtaining the desired promoter through mutation, modification, or screening of existing promoters is challenging. In order to effectively guide how to search for new promoters in the huge potential sequence space, new methodologies should be developed to explore the relationship between promoter sequence and function more thoroughly [5].

© The Author 2022. Published by Oxford University Press. All rights reserved. For permissions, please e-mail: journals.permissions@oup.com

Fortunately, prediction of promoter transcriptional strength has been shown to be amenable to machine learning (ML) techniques, as comprehensively reviewed by de Jongh et al. [6]. Several recent studies have applied ML for promoter strength prediction. By varying a 50-nt region in the 5′ UTR of the HIS3 promoter, Cuperus et al. [7] generated 500000 variants for use as their training dataset for a convolutional neural network (CNN). This allowed them to predictively improve the expression of promoters with both random and native 5′ UTRs. They also demonstrated the advantage of using synthetic libraries as they found motifs that enhance transcription that are absent from the yeast genome. Additionally, ML has also enabled prediction of transcriptional outputs for constitutive, inducible, or synthetic promoters in *S. cerevisiae*. For instance, training a CNN with diversified libraries over 105 in size, created by precisely altering the P_{GPD} constitutive promoter and a P_{ZEV}-based inducible system, allowed prediction of promoter strengths with an accuracy of 79% [8, 9]. Similarly, de Boer et al. [10] created what is perhaps the largest synthetic library ever made for ML promoter engineering by synthesizing over 100 million random 80-bp upstream activating sequences (UASs). This library was used as a training dataset for a TF-motif-based model which was able to correctly predict the expression level of 94% random promoters. In an elegant combination of hybrid promoter engineering and ML in human cells, Wu et al. [11] created 129-bp UASs that contained tandem, human TF binding sites that represented over 6000 unique motifs taken from two TF databases, and then used the resulting dataset to train a generalized linear model with elastic net regularization (GLMNET) [11–13]. Using this model, it was possible to successfully predict differential expression of individual promoters across different cell lines, a difficultly for mammalian cell engineering. In order to extract high-order sequence features and create millions of entirely new promoters, Wang et al. [14] trained a generative adversarial network (GAN) off of natural E. coli promoters. These artificially created promoters shared characteristics with natural promoters like the k-mer frequency and the -10 and -35 regions with E. coli promoters. The promoter strength prediction model screened out these synthetic promoters, and more than 70.8% of them were experimentally demonstrated to be functional. Many of them were as strong as or stronger than the majority of natural promoters and their strongest mutants.

Despite substantial progress the aforementioned studies have made, they still have some limitations. The existing model architectures are simple, mainly focusing on using the DNA sequence compositional characteristics (base content, k-mer features, etc.) of promoters as a priori information to predict their strengths, while paid less attention to the influence of the promoter sequence mutations informations on the strength. As we all know, the effectiveness of popular natural language models will be constrained for long (100–1000) sequences with few words (only 5). To address this limitations, we established EVMP to extract important features related to sequence mutations information to enhance machine learning models, which performed synthetic promoter strength prediction.

Instead of taking synthetic promoter as input directly, EVMP regards the given base promoter and corresponding mutations as input. Base promoters are often specified as which are at the center of the distribution of synthetic promoters. Mutations here are characterized by k-mer mutant subsequences around each mutation site. In this paper, we applied EVMP to various well-known ML models. With the utilization of EVMP, the MAE (mean absolute error) of promoter strength (logarithmic Fluorescence/OD600) got a general descending. MAE (mean absolute error) of LSTM was reduced from 0.50 to 0.18, MAE of the Transformer was reduced from 0.27 to 0.19, and the state-of-the-art result was obtained. Further research showed that EVMP is enabled to reduce the *over-smoothing* phenomenon, which explained the effectiveness of EVMP. Moreover, by virtual dataset expansion based on selecting multiple different base promoters, EVMP formed ensemble learning and further reduced MAE to 0.177. Extensive experiments have shown that EVMP can significantly enhance machine learning models on synthetic promoter strength prediction task, which is expected to be extended to more tasks related to synthetic promoters and even to other tasks of synthetic biological sequences. As far as we know, this is the first research work that integrating mutation features to predict promoter strength.

Methods

EVMP Architecture

Given the base sequence $S = (x_1, \cdots, x_n)$ for each synthetic sequence $S' = (x'_1, \cdots, x'_n)$, where x is a *word* $\in \mathbb{R}^d$, EVMP receives both S and S' as input, which are independently processed by BaseEncoder and VarEncoder. EVMP framework structure is shown in Figure 1.A.

BaseEncoder

Similar to other sequence-based models, BaseEncoder deals with the base sequence directly. Moreover, BaseEncoder takes the base sequence S as input and outputs the embedding e_{base} of S.

$$e_{\mathrm{base}} := \mathrm{BaseEncoder}(S) \tag{1}$$

VarEncoder

Different from the common natural language models that deal with the whole synthetic sequence, VarEncoder deals with mutations in the synthetic sequence. Significantly, the *mutation* here is a general concept that can be depicted in many different ways, rather than just one specific mutation site. In Section 2.1, we present two different mutation representation paradigms. Here, we abstract the mutation representation method into a function M, which hides the base sequence S and synthetic sequence S'. VarEncoder takes the mutations $M(S, S')$ as input and outputs the embedding e_{var} of the mutations.

$$e_{\mathrm{var}} := \mathrm{VarEncoder}(M(S, S')) \tag{2}$$

Representation Decoder

Representation Decoder takes the base embedding e_{base} and variation embedding e_{var} as inputs, outputs the representation $r(S')$ of the synthetic sequence S'. The $r(S')$ can be used for various downstream tasks, such as promoter strength prediction, promoter classification, etc. The representation decoder is an *optional* module in EVMP, which is omitted in this work.

$$r(S') = \mathrm{RepDecoder}(e_{\mathrm{base}}, e_{\mathrm{var}}) \tag{3}$$

Objective Function

For common regression tasks, such as promoter strength prediction, the model is trained on dataset $D = \{(S_i, S'_i, y_i) | i =$

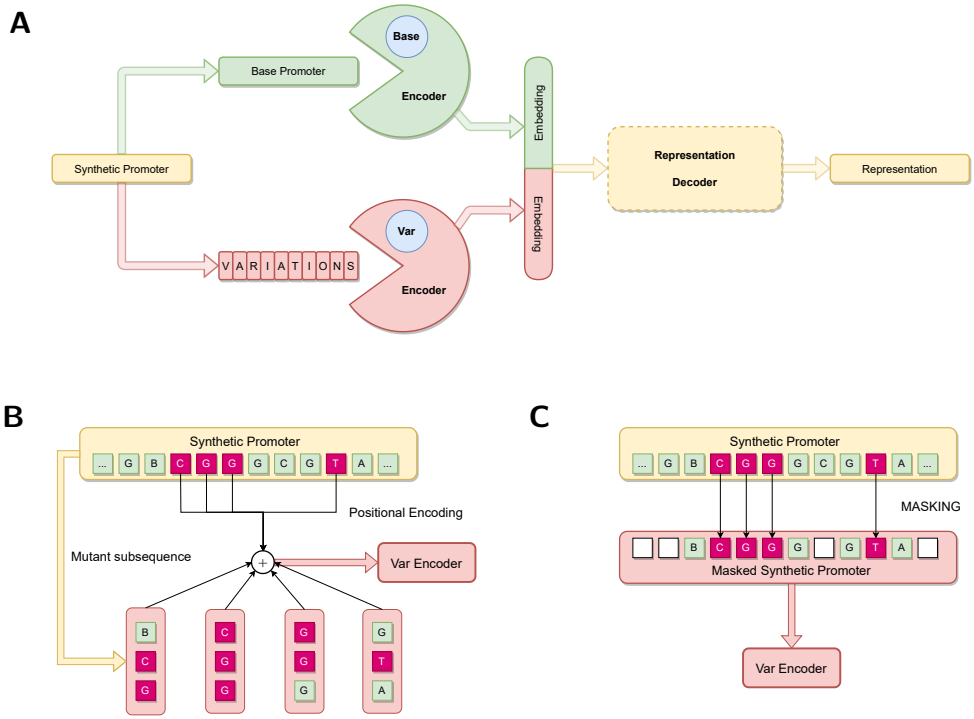

Fig. 1. Extended Vision Mutant Priority Framework. **(A)** Structure of EVMP framework. **(B, C)** two different ways of mutation representation, including vars+PE and mask, which are both based on k-mer mutant subsequences.

$1, \cdots, N\}$ with the objective function:

$$\min_{w} \sum_{i=1}^{N} |\hat{y}_i - y_i| + \lambda \|w\|^2 \tag{4}$$

where $\hat{y}_i$ is the output value of the input (S_i, S'_i), w is the weights of model and λ is the regularization constant.

Mutation Representation Paradigms

Mutation representation is a critical component of EVMP. In Equation (2), the implementation of M for mutation representation is not specified. Since there are various methods of mutation representation, the selection has a decisive influence. The key points of mutation representation are: (i) The extended vision of mutation region. (ii) The retention of mutation position information. Two different paradigms for mutation representation are shown in Figures 1.B and 1.C.

Mutant Subsequence with Positional Encoding

It is natural to use a sequence of mutations as the input of VarEncoder. Here, a mutation actually refers to a *mutant subsequence* in the synthetic sequence. A mutant subsequence are the k words (or k-mer, Liu and Wu [15]) surrounding a mutation position where k denotes the *receptive field* of each mutation, which satisfies the extended vision required in key point (i).

For different mutant subsequences, it is necessary to add position information. One way to add position information for mutations is Positional Encoding (PE, Vaswani et al. [16]), a function $PE : \mathbb{R} \to \mathbb{R}^d$. Positional Encoding is generally added to the sequence directly, thus we call this technique *vars+PE*. In this work, we use sin-cos Positional Encoding in Definition 1.4.

$$\begin{aligned} M_{\text{vars+PE}}(S, S') = \{(&x'_{i-\lfloor k/2 \rfloor} + PE(i), \cdots, \\ &x'_{i+k-\lfloor k/2 \rfloor -1} + PE(i))| \text{ if } x_i \neq x'_i, i = 1, \cdots, n\} \end{aligned} \tag{5}$$

Figure 1.B shows a simple example for vars+PE ($k = 3$). As shown in the example, BaseEncoder and Variation Encoder apply PE in different ways. In base sequence, each word adds a different PE based on its own position. In synthetic sequence, in order to distinguish between various mutant subsequences that might overlap with one another, each word in a mutant subsequence adds the same PE based on the same mutation site, as shown in Equation (5).

Masked Synthetic Sequence

In Section 2.2.1, PE is applied because the selection of mutant subsequences disrupted the initial order of words. If the sequence is kept in its original form, there is no need to provide additional positional information. The *mask* technique fits this need, which assigns words to zero if they never appear in any of the mutant subsequences to disguise the less significant terms.

$$\begin{aligned} M_{\text{mask}}(S, S') = S' \cdot (\mathbb{I}(&x_{i-k+\lfloor k/2 \rfloor +1} \neq x'_{i-k+\lfloor k/2 \rfloor +1} \vee \\ &\cdots \vee x_{i+\lfloor k/2 \rfloor} \neq x'_{i+\lfloor k/2 \rfloor})|i = 1, \cdots, n) \end{aligned} \tag{6}$$

Figure 1.C shows a simple example for mask ($k = 3$). Words that are never appear in any of the mutant subsequences will be masked, but their original positions will remain unchanged.

Comparison of Mutation Representation Paradigms

The length of mutation representation is the most significant difference between the above two paradigms. For n words sequence, assuming the average number of mutations is α. The mutation representation of vars+PE is αk words, while that of mask is n words. Generally, k should be set lower than n/α to prevent excessive mutant subsequence overlap.

The time complexity of both techniques is $O(nk)$. By setting k to a reasonable size, vars+PE will have lower space complexity than mask. Therefore, in terms of time complexity, vars+PE is superior to mask. Mask, on the other hand, is easier to implement. While VarEncoder using vars+PE needs a more complex design, VarEncoder using mask can directly reuse the structure of BaseEncoder.

Experiments

Dataset

Our work is mainly based on the synthetic promoter library created by Zhao et al. [4], which consisted of 3665 different synthetic promoters and used log Fluorescence/OD_{600} to indicate promoter strength. After sequence alignment with P_{trc} using MEGA [17], the synthetic promoters with large differences were discarded, and 3567 synthetic promoters were obtained. The distribution of the number of mutations for synthetic promoters in comparison to P_{trc} is shown in Figure 2.A. The probability that each position is included in the receptive field of VarEncoder is shown in Figure 2.B. When k equals 3 and 8, the receptive fields of VarEncoder turn out to be 10 and 23 respectively, i.e. 11.8% and 27.1% of the whole promoter length. The subsequent experiments demonstrate that EVMP still achieves superior results even with this level of information loss.

Model Implementations

For the promoter representation tasks, EVMP is applied to a large number of machine learning models. Deep learning models include LSTM [18, 19], Transformer [16], and 1D-CNN [20, 21]. Conventional machine learning models include decision tree models such as Random Forests [22], GBDT [23] and XGBoost [24], as well as linear classifiers such as SVM [25, 26]. The necessary mathematical foundations for the aforementioned machine learning methods are provided in Appendix 1.

In contrast to NLP models, which take S' as input, EVMP takes $\langle S, M(S, S')\rangle$ as input, where S is the specified base promoter and S' is the synthetic promoter. Each base-pair in the sequence is represented by a one-hot encoding. Both LSTM and Transformer in deep learning models utilize vars+PE technique to represent mutations, while 1D-CNN and all conventional machine learning models apply mask technique. It should be noted that instead of directly utilization of vars+PE technique, different mutant subsequences are mapped to unique vectors before being input into VarEncoder. The mutant receptive field $k = 8$ for all models, except $k = 3$ of EVMP-LSTM.

The art of implementing EVMP for a specific machine learning model is to reuse its original structure. BaseEncoder is structural identical to the original machine model. VarEncoder with vars+PE technique is similar to BaseEncoder in structure except that some hyperparameters are modified to fit the input of mutant subsequences, while VarEncoder with mask technique is identical to BaseEncoder, as mentioned in Section 2.2.3.

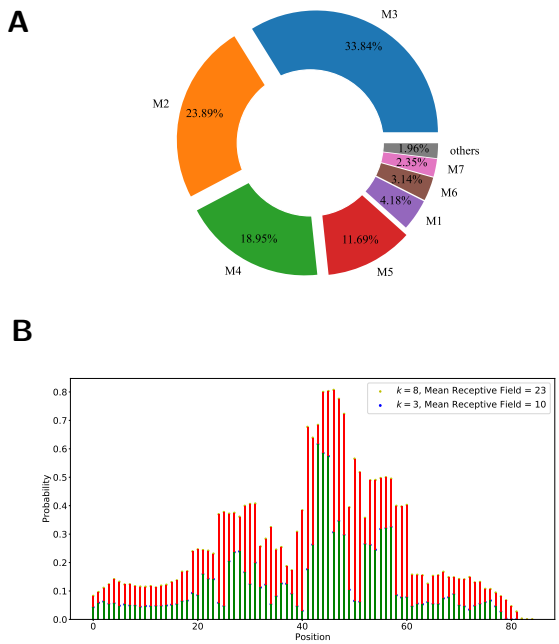

Fig. 2. Visualization of dataset. **(A)** The proportion of different mutation number of synthetic promoters in 3567 synthetic promoters. Mt stands for mutated synthetic promoters at t different positions. **(B)** The probability that each position is included in the receptive field of the VarEncoder. Each promoter is filled B to 85 length.

Figure 3 depicts the EVMP implementations of three deep learning models, including EVMP-LSTM, EVMP-Transformer and EVMP-1DCNN, and gives the visual interpretation of four conventional machine learning models. For more details on the model implementations, please refer to Appendix 2. Moreover, various learning rate adjustment strategies are activated for experiment, and the strategies are enabled in Appendix 1.6. The source code is available at https://github.com/Tiny-Snow/EVMP.

Synthetic Promoter Strength Prediction Task

The synthetic promoter strength prediction task is the primary experiment in this study. This regression task receives the input described in Section 3.2 and outputs the logarithmic strength of the promoter. In order to guarantee that all models have the same data set partition, we randomly split the dataset into training, validation and test sets with an 8:1:1 ratio.

The results of EVMP and non-EVMP models on synthetic promoter strength prediction task are listed in Table 1. MAE (mean absolute error) and R^2 (coefficient of determination) are considered to be the evaluation metrics of models. Visual results are available at Figure A3 in Appendix 3.

EVMP Enhances or Keeps the Effect of ML Models

Generally, the results of non-EVMP models in Table 1 were incredibly inconsistent with the deep learning era. Results in Figure A3.A indicated that LSTM could hardly output anything other than the average strength, revealing the fact that the length of promoter sequence seemed to be beyond the capacity of LSTM. Transformer performed inferior to any

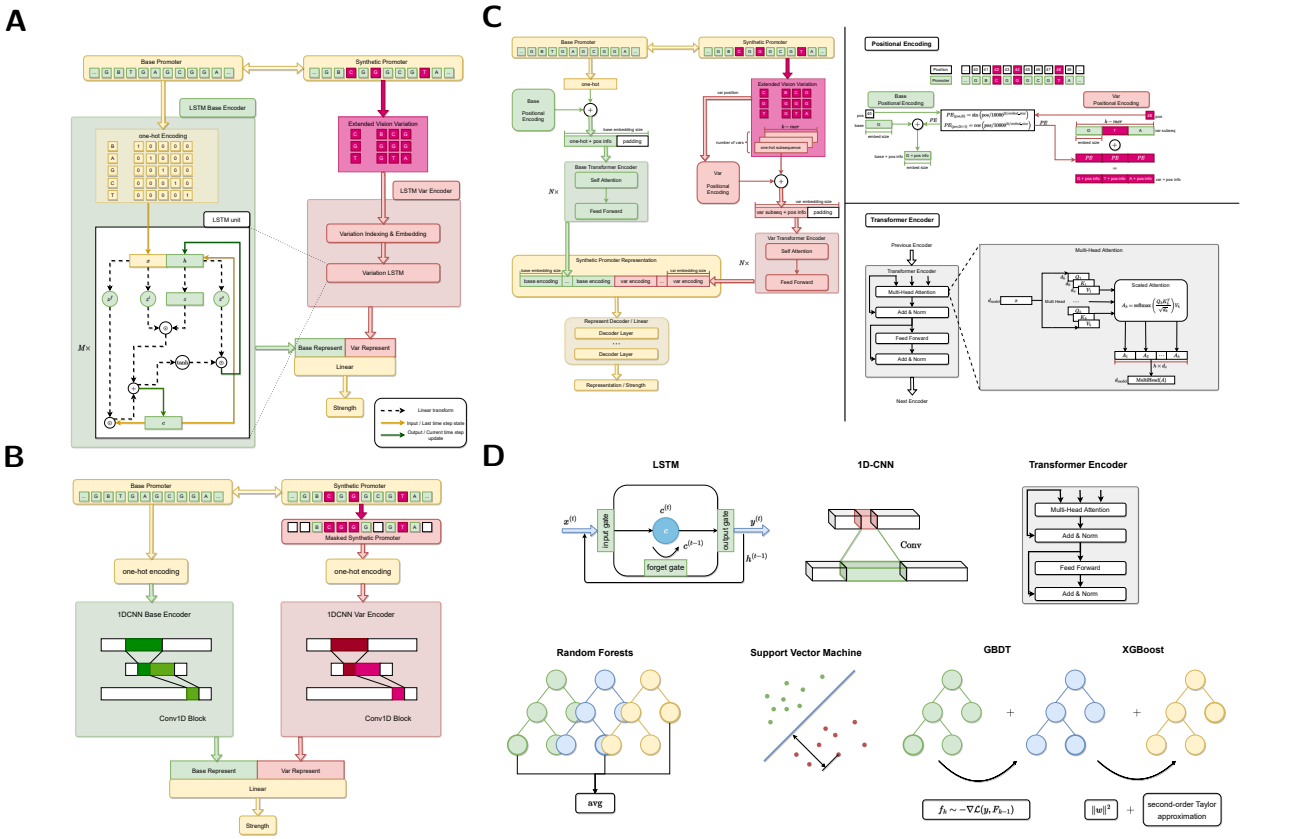

Fig. 3. Implementations of different EVMP and non-EVMP models. **(A, B, C)** Implementations of EVMP models, including EVMP-LSTM, EVMP-1DCNN and EVMP-Transformer. **(D)** Implementations of non-EVMP models, including conventional machine learing models, which are also the components used to construct the BaseEncoder and VarEncoder.

conventional machine learning models, which supposed to be unlikely. The result of 3-layer 1DCNN was remarkably similar to machine learning models. Among the conventional machine learning models, decision tree models outperformed SVM, probably for the reason of ensemble learning (Theorem 1.2 in Appendix 1.5). In general, the MAE of any non-EVMP model was not lower than 0.23.

Experiments showed that the performances of mentioned models were generally improved after applying EVMP. In stark contrast to the performance of LSTM, EVMP-LSTM achieved one of the best results, with MAE decreasing from 0.50 to 0.18. EVMP-Transformer also had a significant improvement over its original effect, with MAE decreasing from 0.27 to 0.19, which was far beyond the effects of any conventional machine learning models. 3-layer 1DCNN with EVMP kept the effect of that without EVMP. In addition, four conventional machine learning models performed better after the implementation of EVMP.

These experiments with various model implementations demonstrated the efficacy of EVMP, i.e. EVMP either enhance or maintain the effect of machine learning models on the task of predicting synthetic promoter strength.

Why EVMP: Experimental Perspective

The fact that EVMP reduces the phenomenon of *over-smoothing* representation contributes to its effectiveness. Over-smoothing phenomenon occurs when the outputs of different inputs appear to be almost the same. Appropriate smoothing seems to be regarded as a symbol of robustness in other tasks, such as image recognition [27]. However, considering the fact that the differences between synthetic promoters are much shallower than common natural language data, over-smoothing phenomenon is likely to lead to a poor performance.

We define *average embedding distance* in order to study over-smoothing phenomenon. Assume that synthetic promoters are $x_1, \cdots, x_N$, and x_0 is the base promoter (P_{trc} promoter). The output embedding vector $e_l(x) \in \mathbb{R}^{n_l}$ are calculated for each layer l and each input x. Denote the average embedding distance d_l of each layer as follows:

$$d_l = \frac{1}{N} \sum_{i=1}^{N} \frac{\|(e_l(x_i) - e_l(x_0)\|_2}{\sqrt{n_l}} \tag{7}$$

Average embedding distance measures the average distance of each dimension of the embedding at the l-th layer, thus it is capable to represent the dispersion of the output of each layer in neural network.

The average embedding distance of each layer was calculated for the EVMP and non-EVMP implementations of the 3-layer 1DCNN, LSTM and Transformer respectively, especially, LSTM calculated the average embedding distance of each time step, as shown in Figure 4. Figures 4.B, 4.C and 4.E showed that after applying EVMP, the average embedding distance of LSTM and Transformer made a significant increase, while the average embedding distance of 3-layer 1DCNN was also slightly ascending, which is consistent with our experimental results in the synthetic promoter strength prediction task. Experimental results demonstrated that EVMP can effectively alleviate over-smoothing phenomenon.

Table 1. Results of deep learning models and machine learning models on synthetic promoter strength prediction task. The entry *Mutant* is mutation representation method. k is the receptive field of each mutation. MAE is mean absolute error, R^2 is coefficient of determination.

Model	Mutant	k	Train		Valid		Test	
			MAE	R^2	MAE	R^2	MAE	R^2
LSTM	var+PE	3	0.51	-0.01	0.50	-0.01	0.50	0.00
EVMP-LSTM			0.16	0.78	**0.20**	**0.73**	**0.18**	**0.79**
Transformer		8	0.28	0.55	0.29	0.45	0.27	0.61
EVMP-Transformer			0.12	0.84	**0.19**	**0.74**	**0.19**	**0.75**
1DCNN (3 layers)	Mask		0.22	0.68	0.24	0.67	0.23	0.70
1DCNN (50 layers)			0.51	-0.01	0.50	-0.01	0.50	0.00
1DCNN (50 layers + ResNet)			0.20	0.71	**0.23**	**0.71**	**0.23**	**0.70**
EVMP-1DCNN (3 layers)			0.21	0.70	0.23	0.70	0.23	0.71
EVMP-1DCNN (50 layers)			0.51	-0.01	0.50	-0.01	0.50	0.00
EVMP-1DCNN (50 layers + ResNet)			0.21	0.70	**0.23**	**0.68**	**0.23**	**0.68**
SVM			0.23	0.68	0.26	0.62	0.25	0.63
GBDT			0.18	0.84	0.25	0.65	0.24	0.68
XGBoost			0.14	0.89	0.24	0.66	0.23	0.69
RF			0.08	0.95	0.23	0.63	0.23	0.64
EVMP-SVM			0.21	0.72	**0.24**	**0.66**	**0.23**	**0.67**
EVMP-GBDT			0.16	0.87	**0.22**	**0.70**	**0.23**	**0.69**
EVMP-XGBoost			0.13	0.91	**0.22**	**0.70**	**0.23**	**0.69**
EVMP-RF			0.08	0.95	**0.22**	**0.68**	**0.22**	**0.68**

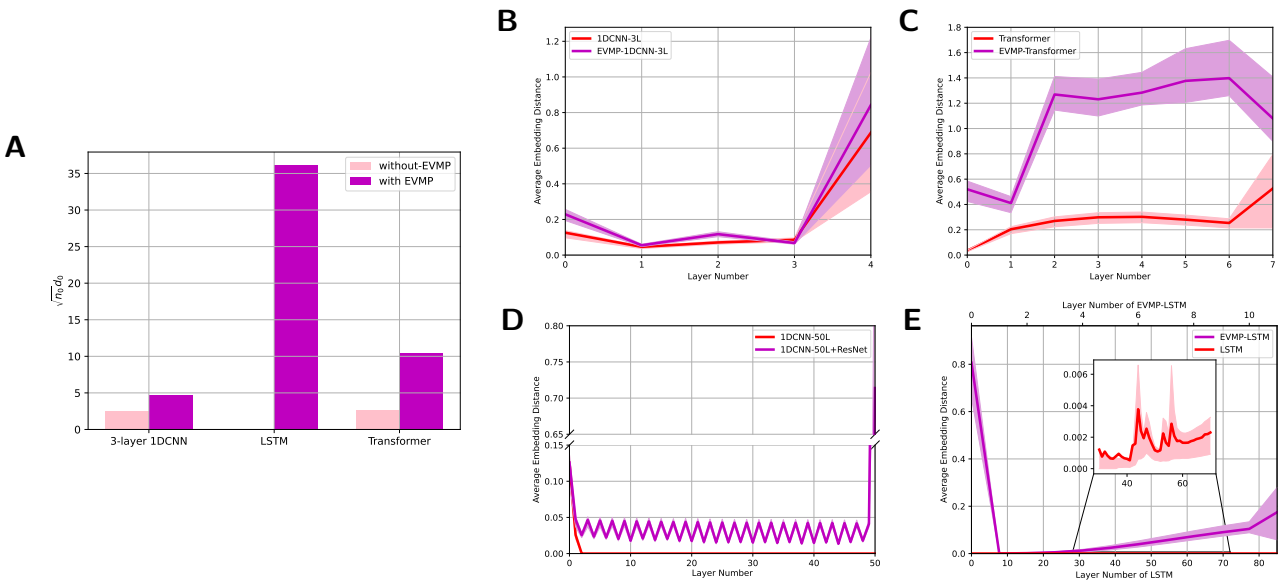

Fig. 4. Average embedding distance for each layer of deep learning models, with the 0-th layer serving as the input. Since BaseEncoder receives the same base promoter P_{trc}, non-EVMP models calculate the average embedding distance of each layer directly, while EVMP models calculate the average embedding distance of VarEncoder. **(A)** non-norm d_0 ($\sqrt{n_0}d_0$) for 1DCNN-3L, LSTM and Transformer. **(B)** Average embedding distance for each layer of 1DCNN-3L and EVMP-1DCNN-3L. **(C)** Average embedding distance for each layer of Transformer and EVMP-Transformer. **(D)** Average embedding distance for each layer of 1DCNN-50L and 1DCNN-50L+ResNet. **(E)** Average embedding distance for each layer of LSTM and EVMP-LSTM, where LSTM calculates the average embedding distance for each time step.

Why EVMP: Theoretical Perspective

Section 3.3.2 provides a comparison of the average embedding distance between neural network layers, in order to intuitively verify whether EVMP addresses the problem of over-smoothing representation and to elaborate the effectiveness of EVMP. Here, a theoretical illustration is provided to the rationality of this perspective.

Zou et al. [28] proved that when the width of each layer in deep ReLU networks is at least $\widetilde{\Omega}\left(n^{14}L^{16}/\phi^4\right)$, gradient descent can achieve zero training error within $\widetilde{O}\left(n^5L^3/\phi\right)$ iterations, where n is the number of training examples, L is the number of hidden layers, and ϕ is the maximum lower bound of the L_2-norm between any two training samples.

Here, we regard $\sqrt{n_0}d_0$ as an approximation of ϕ, where n_0 is the dimension of input. Figure 4.A illustrates ϕ of the three deep learning models increased significantly after utilizing EVMP. Since the size of the training set n and the number of model layers L are remained unchanged, the

overparameterizations of layer width and training iterations are reduced, and the model fitting effect tends to be improved. This statement is also enabled to be applied to RNN and CNN [28].

In the case of conventional machine learning models, as we proved in Appendix 1.3, decision tree models and SVM with mask EVMP have smaller regularization penalties, which may lead to better fitting results. Although we only demonstrated the case of $k = 1$, this claim still holds for the case of $k > 1$.

Why EVMP: Not a Panacea is NOT a Weakness

We have provided explanations for the superior performance of EVMP in machine learning from both experimental and theoretical perspectives. Theoretically, EVMP model are not likely to produce worse outcomes than the original model, but it is also not guaranteed that the results will be better. From the results of 50-layer 1DCNN and 50-layer 1DCNN+ResNet in Table 1, EVMP appeared to fail to significantly enhance the extremely deep models. Zou et al. [28] pointed out that as for deep ReLU networks, the overparameterization of layer width W is polynomial to the number of layers L, while Frei et al. [29] proved that for deep ReLU networks with residual structure, the overparameterization of W is logarithmic to L. This indicates that underfitting phenomenon is caused by excessive depth, which should be solved by skip-connection structure rather than EVMP, shown in Figure 4.D. However, EVMP can still keep the non-EVMP effect. Therefore, even though it does not solve all the harsh problems, EVMP can still be regarded as a competitive candidate of synthetic biology machine learning paradigm.

Virtual Dataset Expansion by Multiple Base Promoters Forms Ensemble Learning

In previous discussion, we only specify P_{trc} as the unique base promoter. However, in a realistic scenario, synthetic promoter dataset may be mutated from multiple base promoters, which is a basic requirement of our framework. Instead of finding multiple synthetic promoter libraries which are mutated from different base promoters, we chose to select several sufficiently different synthetic promoters as base promoters, which can be approximately substituted for the previous case. In Table A5, the first 10 synthetic promoters were selected as base promoters. The performance of EVMP-Transformer on validation and test proportions are shown in Table 2.

Table 2. Results of EVMP-Transformer in synthetic promoter strength prediction task with multiple base promoters and virtual dataset expansion. The entries *Train*, *Valid* and *Test* refer to the dataset processing method, where *Fix* refers to using P_{trc} as base promoter, *Rand* refers to randomly selecting one of the 10 base promoters as base promoter, and $\times 10$ refers to generating 10 copies of all 10 base promoters. For visualization of the results, please refer to Figure A4 in Appendix 3.

ID	EVMP	Train	Valid	Test	Valid MAE	Test MAE
E1		Fix	Fix	Fix	0.281	0.290
E2	✓	Rand	Rand	Rand	0.238	0.250
E3	✓	Fix	Fix	Fix	0.194	0.189
E4	✓	×10	Fix	Fix	0.185	0.191
E5	✓	×10	Rand	Rand	**0.177**	**0.177**

In the five experiments, E1 and E3 were the EVMP/non-EVMP results obtained before. In E2, each promoter from both the training and validation/test sets randomly used one of the 10 base promoters. This means for a particular base promoter, its training set shrank by 9/10. Therefore, the effect of E3 was lower than that of E2, but it was still higher than that of E1, which indicated that EVMP is sufficiently robust to improve the effect of the model.

In E4 and E5, training set used 10-fold *virtual dataset expansion*, which means each promoter from training set used all 10 base promoters to produce 10 different inputs. In this case, the training set for each base promoter was the original size. The difference between E4 and E5 is that E4 used the validation/test set when P_{trc} was used as the base promoter, while E5 handled the validation/test set in the same way as E2. Experimental results showed that E3 and E4 achieved similar effects, and E5 achieved the best effect in all experiments, with MAE as low as 0.177. The reason why E5 works better than E2 should be attributed to the size of the training set, which shows that 10-fold expansion does work. The improved effect of E5 over E4 indicated that P_{trc} is not the only suitable base promoter. Based on the similar results for E3 and E4, we can consider that the model forms a de facto ensemble learning for different base promoters.

On the basis of the above ideas, multiple base promoters should be applied to EVMP. Interestingly, the 10-fold expansion of the training set is exactly *virtual* – the size of original synthetic promoter library does not change throughout. In addition, considering that the differences between the base promoters used in the above experiments were large enough, the conclusions can also be reasonably extended to the general case.

Discussion

Enhancing machine learning model performance is just one aspect of EVMP. For instance, building a sizable mutation-based synthetic promoter library necessitates a lot of resources, and a key issue is how sizable a dataset must be to maintain a previous error rate following the application of EVMP? Here, given a fixed validation proportion (1/10 of the original dataset), we randomly selected 1/9, 2/9, $\cdots$, 9/9 of the remaining 9/10 dataset as training proportion and trained EVMP-Transformer in the previous environment. The MAE and R^2 of the training and validation sets for each model are shown in Figure 5.

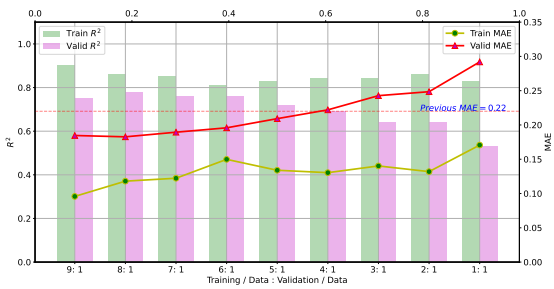

Fig. 5. As the proportion of the training set in the original dataset decreases, the MAE of the validation set increases and the R^2 decreases. When the proportion of training set to validation set is higher than 5:1, EVMP-Transformer achieves lower MAE than non-EVMP machine learning models.

The experiments show a pattern where the MAE ascending speed increases in accordance with the decreasing of the training set proportion. In general, when the validation set MAE is better than 0.22, which is the MAE of earlier conventional machine learning models trained on 9:1 dataset, it is reasonable to assume that the proportion of training sets is sufficient. According to Figure 5, the ratio appears to be 5:1, showing that EVMP can effectively improve machine learning models while lowering the cost of creating a synthetic promoter library with only 56% of its original size.

Several studies have demonstrated that EVMP typically yields better outcomes for the prediction of synthetic promoter strength. The EVMP may also be used for other tasks that call for the representation of synthetic promoters and could even be applied to other synthetic biological sequences. We demonstrate a self-supervised AutoEncoder implemented by EVMP in Appendix 1.7, and a 99.67% accuracy shows that EVMP is capable of producing effective representations of synthetic promoters.

Conclusion

Synthetic promoter strength prediction is a critical issue in synthetic biology. In this work, we proposed EVMP as a unified framework enhancing machine learning models. The core idea of EVMP lies in highlighting mutations with k-mer extended vision. A series of comparative experiments demonstrated that EVMP models are superior to the existing state-of-the-art non-EVMP models. The excellent performance of EVMP is attributed to the alleviation of over-smoothing phenomenon and the virtual dataset expansion by selecting multiple base promoters. The source code for EVMP implementations of various ML models is given in https://github.com/Tiny-Snow/EVMP.

We expect our method to act as a useful tool in the area of synthetic promoter representation. It should be pointed out that EVMP is a methodology, which needs to be combined with a more refined design of neural network to obtain better results. Therefore, future work may target further improvements in the mutation representation. In addition, information interaction between BaseEncoder and VarEncoder can also be attempted while ensuring stable training.

Supplementary Material

Appendix is available online at https://www.biorxiv.org.

Funding

This work was supported by Grants 31970113 and 32170065 from the National Natural Science Foundation of China.

Key Points

- We propose EVMP (Extended Vision Mutant Priority) framework, which enhance various machine learning models for synthetic promoter strength predication by splitting synthetic promoter into base promoter and k-mer mutations.
- EVMP enhances the effect of existing machine learning models and substantially improves the prediction accuracy.
- EVMP alleviates the over-smoothing phenomenon that occurs when NLP models process synthetic promoters.
- EVMP forms ensemble learning by selecting multiple base promoters and virtual dataset expansion.

References

1. J. Gao, L. Jiang, J. Lian, Development of synthetic biology tools to engineer pichia pastoris as a chassis for the production of natural products, Synthetic and systems biotechnology 6 (2021) 110–119.
2. H. Tang, Y. Wu, J. Deng, N. Chen, Z. Zheng, Y. Wei, X. Luo, J. D. Keasling, Promoter architecture and promoter engineering in saccharomyces cerevisiae, Metabolites 10 (2020) 320.
3. H. Alper, C. Fischer, E. Nevoigt, G. Stephanopoulos, Tuning genetic control through promoter engineering, Proceedings of the National Academy of Sciences 102 (2005) 12678–12683.
4. M. Zhao, Z. Yuan, L. Wu, S. Zhou, Y. Deng, Precise prediction of promoter strength based on a de novo synthetic promoter library coupled with machine learning, ACS Synthetic Biology 11 (2021) 92–102.
5. A. P. Cazier, J. Blazeck, Advances in promoter engineering: novel applications and predefined transcriptional control, Biotechnology Journal 16 (2021) 2100239.
6. R. P. de Jongh, A. D. van Dijk, M. K. Julsing, P. J. Schaap, D. de Ridder, Designing eukaryotic gene expression regulation using machine learning, Trends in biotechnology 38 (2020) 191–201.
7. J. T. Cuperus, B. Groves, A. Kuchina, A. B. Rosenberg, N. Jojic, S. Fields, G. Seelig, Deep learning of the regulatory grammar of yeast 5' untranslated regions from 500,000 random sequences, Genome research 27 (2017) 2015–2024.
8. R. S. McIsaac, P. A. Gibney, S. S. Chandran, K. R. Benjamin, D. Botstein, Synthetic biology tools for programming gene expression without nutritional perturbations in saccharomyces cerevisiae, Nucleic acids research 42 (2014) e48–e48.
9. B. J. Kotopka, C. D. Smolke, Model-driven generation of artificial yeast promoters, Nature communications 11 (2020) 1–13.
10. C. G. de Boer, E. D. Vaishnav, R. Sadeh, E. L. Abeyta, N. Friedman, A. Regev, Deciphering eukaryotic gene-regulatory logic with 100 million random promoters, Nature biotechnology 38 (2020) 56–65.
11. M.-R. Wu, L. Nissim, D. Stupp, E. Pery, A. Binder-Nissim, K. Weisinger, C. Enghuus, S. R. Palacios, M. Humphrey, Z. Zhang, et al., A high-throughput screening and computation platform for identifying synthetic promoters with enhanced cell-state specificity (specs), Nature communications 10 (2019) 1–10.

12. P. Kheradpour, M. Kellis, Systematic discovery and characterization of regulatory motifs in encode tf binding experiments, Nucleic acids research 42 (2014) 2976–2987.
13. M. T. Weirauch, A. Yang, M. Albu, A. G. Cote, A. Montenegro-Montero, P. Drewe, H. S. Najafabadi, S. A. Lambert, I. Mann, K. Cook, et al., Determination and inference of eukaryotic transcription factor sequence specificity, Cell 158 (2014) 1431–1443.
14. Y. Wang, H. Wang, L. Wei, S. Li, L. Liu, X. Wang, Synthetic promoter design in escherichia coli based on a deep generative network, Nucleic Acids Research 48 (2020) 6403–6412.
15. W.-l. Liu, Q.-b. Wu, Analysis method and algorithm design of biological sequence problem based on generalized k-mer vector, Applied Mathematics-A Journal of Chinese Universities 36 (2021) 114–127.
16. A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, I. Polosukhin, Attention is all you need, Advances in neural information processing systems 30 (2017).
17. S. Kumar, M. Nei, J. Dudley, K. Tamura, MEGA: A biologist-centric software for evolutionary analysis of DNA and protein sequences, Briefings in Bioinformatics 9 (2008) 299–306.
18. S. Hochreiter, J. Schmidhuber, Long short-term memory, Neural computation 9 (1997) 1735–1780.
19. F. A. Gers, J. Schmidhuber, F. Cummins, Learning to forget: Continual prediction with lstm, Neural computation 12 (2000) 2451–2471.
20. Y. LeCun, B. Boser, J. Denker, D. Henderson, R. Howard, W. Hubbard, L. Jackel, Handwritten digit recognition with a back-propagation network, Advances in neural information processing systems 2 (1989).
21. S. Kiranyaz, O. Avci, O. Abdeljaber, T. Ince, M. Gabbouj, D. J. Inman, 1d convolutional neural networks and applications: A survey, Mechanical systems and signal processing 151 (2021) 107398.
22. L. Breiman, Random forests, Machine learning 45 (2001) 5–32.
23. J. H. Friedman, Greedy function approximation: a gradient boosting machine, Annals of statistics (2001) 1189–1232.
24. T. Chen, C. Guestrin, Xgboost: A scalable tree boosting system, in: Proceedings of the 22nd acm sigkdd international conference on knowledge discovery and data mining, 2016, pp. 785–794.
25. C. Cortes, V. Vapnik, Support-vector networks, Machine learning 20 (1995) 273–297.
26. B. E. Boser, I. M. Guyon, V. N. Vapnik, A training algorithm for optimal margin classifiers, in: Proceedings of the fifth annual workshop on Computational learning theory, 1992, pp. 144–152.
27. A. Ruderman, N. C. Rabinowitz, A. S. Morcos, D. Zoran, Pooling is neither necessary nor sufficient for appropriate deformation stability in cnns, arXiv preprint arXiv:1804.04438 (2018).
28. D. Zou, Y. Cao, D. Zhou, Q. Gu, Gradient descent optimizes over-parameterized deep relu networks, Machine learning 109 (2020) 467–492.
29. S. Frei, Y. Cao, Q. Gu, Algorithm-dependent generalization bounds for overparameterized deep residual networks, Advances in neural information processing systems 32 (2019).

Wei-Qin Yang. is an undergraduate student at the School of Computer Science and Technology, Shandong University. His research interests include machine learning and bioinformatics.

De-Xin Li. is an undergraduate student at the School of Computer Science and Technology, Shandong University. His research interests include natural language processing and bioinformatics.

Ran-Ran Huang. is an associate professor at the Institute of Marine Science and Technology, Shandong University. His research interests include bioinformatics and Bacterial posttranscriptional regulation