DREAM Challenge 2022 Predicting gene expression using millions of random promoter sequences by Metformin-121

Abstract

The previous studies of prediction of gene expression usually used lots of methods of data preprocessing based on biological knowledge. However, these methodologies may cost lots of time to design and train.
Here, we aimed to propose an end-to-end solution of deep learning for predicting gene expression. We utilized bidirectional gated recurrent unit to develop a novel method to predict gene expression with minimal methods of data preprocessing.

1. Description of data usage

The provided data were divided into training set (n = 6,000,000) and validation set (n = 739,258) according to row order. By using the one-hot encoding, each character in the DNA sequence is represented by 4 channels ('T':[1,0,0,0], 'C':[0,1,0,0], 'G':[0,0,1,0], 'A':[0,0,0,1],'N':[1,1,1,1]). One-hot encoding in each DNA sequence, including the adapters that flank the promoter and itself, is stored in a list of length 142. The one-hot encoding [0, 0, 0, 0] represented to the missing value was filled in the front end of the list when the length of the DNA sequence was shorter than 142. The predicted value is obtained by dividing the original value by 2 times the median of the data (2 * np.median(Y)). There are no data generators or data augmentation strategies.

2. Description of the model

Our model was built on a combined architecture of 8 bidirectional gated recurrent unit (GRU), an attention layer [1,2], and finally a dense layer. Between attention layer and dense layer, to avoid overfitting, batch normalization is applied [2]. LeakyReLU activation function, a leaky version of Rectified Linear Unit, was used after batch normalization [3]. After LeakyReLU activation, we utilized dropout layer for generalization.

3. Training procedure

Our classification training was carried out using mean squared error loss function and ADAM optimizer [4]. Initial learning rate is 0.001.
For the best model, coefficient of determination of training set is 0.5728 and coefficient of determination of validation set is 0.5372.

4. Other important features

Models were evaluated on their performance on the validation set for 30 training epochs. The best model was saved based on the lowest loss.

5. Contributions and Acknowledgement

5.1 Contributions

Name	Affiliation	Email
Tsai-Min Chen	(1) Graduate Program of Data Science, National Taiwan University and Academia Sinica, Taipei, Taiwan (2) Research Center for Information Technology Innovation, Academia Sinica, Taipei, Taiwan	b99612040@ntu.edu.tw
Chih-Han Huang	ANIWARE, Taipei, Taiwan	robin.ch.huang@gmail.com
Hsuan-Kai Wang	Independent Researcher	wanghsuankai@gmail.com
Edward S.C. Shih	Institute of Biomedical Sciences, Academia Sinica, Taipei, Taiwan	shihds@gate.sinica.edu.tw
Sz-Hau Chen	Development Center for Biotechnology, Taipei, Taiwan	szhau.chentw@gmail.com
Chih-Hsun Wu	Artificial Intelligence and E-learning Center, National Chengchi	j20031214@gmail.com

	University, Taipei, Taiwan	
Jhih-Yu Chen	Graduate Institute of Biomedical Electronics and Bioinformatics, National Taiwan University, Taipei, Taiwan	a402250025@gmail.com
Kuei-Lin Huang	School of Medicine, China Medical University, Taichung, Taiwan	judyhuang4705@gmail.com

5.2 Acknowledgement

We thankfully acknowledge the DREAM Challenge 2022 Predicting gene expression using millions of random promoter sequences and Dr. Carl de Boer for their help and for making the data publicly available.

6. References

1. Schuster, M., and Paliwal, K.K. (1997). Bidirectional recurrent neural networks. Ieee Transactions on Signal Processing 45, 2673-2681.
2. Yang, Z., Yang, D., Dyer, C., He, X., Smola, A.J., and Hovy, E.H. (2016). Hierarchical Attention Networks for Document Classification. Paper presented at: HLT-NAACL.
3. Maas, A.L. (2013). Rectifier Nonlinearities Improve Neural Network Acoustic Models Proc. icml 30 (1), 3.
4. Kingma, D.P., and Ba, J. (2014). Adam: A Method for Stochastic Optimization. In ArXiv eprints.

7. Brief instructions on how to run the code

We suggest creating an empty environment and installing the packages by:

pip install -r requirements.txt

The codes of data preprocessing, training, and testing includes in Metformin-121.ipynb. First, put the training data “train_sequences.txt” and testing data “test_sequences.txt” into the same folder as Metformin-121.ipynb. After executing the codes from the top to the bottom, expression_f4_b32_full_gru_8_coeff_determination_loss would be the best model. “test_sequences_simon_22.txt” would be the prediction results.