Bioinformatics, 35(19), 2019, 3831–3833
doi: 10.1093/bioinformatics/btz165
Advance Access Publication Date: 8 March 2019
Applications Note

OXFORD

Sequence analysis

PyFeat: a Python-based effective feature generation tool for DNA, RNA and protein sequences

Rafsanjani Muhammod[1,†], Sajid Ahmed[1,†], Dewan Md Farid[1], Swakkhar Shatabda[1,*], Alok Sharma [2,3,4,*] and Abdollah Dehzangi[5,*]

[1]Department of Computer Science and Engineering, United International University, Dhaka, Bangladesh, [2]School of Engineering and Physics, University of the South Pacific, Private Mail Bag, Laucala Campus, Suva, Fiji, [3]RIKEN Center for Integrative Medical Sciences, Yokohama, Kanagawa, Japan, [4]Institite for Integrated and Intelligent Systems, Griffith University, Brisbane, Queensland, Australia and [5]Department of Computer Science, Morgan State University, Baltimore, MD, USA

*To whom correspondence should be addressed.

[†]The authors wish it to be known that, in their opinion, the first two authors should be regarded as Joint First Authors.

Associate Editor: John Hancock

Received on September 4, 2018; revised on February 11, 2019; editorial decision on March 2, 2019; accepted on March 6, 2019

Abstract

Motivation: Extracting useful feature set which contains significant discriminatory information is a critical step in effectively presenting sequence data to predict structural, functional, interaction and expression of proteins, DNAs and RNAs. Also, being able to filter features with significant information and avoid sparsity in the extracted features require the employment of efficient feature selection techniques. Here we present PyFeat as a practical and easy to use toolkit implemented in Python for extracting various features from proteins, DNAs and RNAs. To build PyFeat we mainly focused on extracting features that capture information about the interaction of neighboring residues to be able to provide more local information. We then employ AdaBoost technique to select features with maximum discriminatory information. In this way, we can significantly reduce the number of extracted features and enable PyFeat to represent the combination of effective features from large neighboring residues. As a result, PyFeat is able to extract features from 13 different techniques and represent context free combination of effective features. The source code for PyFeat standalone toolkit and employed benchmarks with a comprehensive user manual explaining its system and workflow in a step by step manner are publicly available.
Results: https://github.com/mrzResearchArena/PyFeat/blob/master/RESULTS.md.
Availability and implementation: Toolkit, source code and manual to use PyFeat: https://github.com/mrzResearchArena/PyFeat/
Contact: abdollah.dehzangi@morgan.edu
Supplementary information: Supplementary data are available at *Bioinformatics* online.

1 Introduction

Extracting effective features from sequence data which contains significant discriminatory information is considered the most important step in developing accurate computational methods to predict structural, functional, interaction or expression levels. Most of the methods that have been previously proposed produce a large number of features that in most of the cases do not provide sufficient discriminatory information and at the same time introduce noise and case sparsity in the feature set. Therefore, in addition to feature, extracting informative ones is also a challenging task in

Table 1. Features description

Feature name	Number of features	Applicable for
zCurve	3 features for DNA/RNA	DNA/RNA
gcContent	1 feature for DNA/RNA	DNA/RNA
ATGC ratio	1 feature for DNA/RNA	DNA/RNA
Cumulative Skew	2 features for DNA/RNA	DNA/RNA
Chou's Pseudo composition	when –kGap = 1, 84 features for DNA/RNA and 8420 for protein	DNA/RNA/Protein
monoMonoKGap	when –kGap = 1, 16 features for DNA/RNA and 400 for protein	DNA/RNA/Protein
monoDiKGap	when –kGap = 1, 64 features for DNA/RNA and 8000 for protein	DNA/RNA/Protein
monoTriKGap	when –kGap = 1, 256 features for DNA/RNA and 160 000 for protein	DNA/RNA/Protein
diMonoKGap	when –kGap = 1, 64 features for DNA/RNA and 8000 for protein	DNA/RNA/Protein
diDiKGap	when –kGap = 1, 256 features for DNA/RNA and 160 000 for protein	DNA/RNA/Protein
diTriKGap	when –kGap = 1, 1024 features for DNA/RNA and 3 200 000 for protein	DNA/RNA/Protein
triMonoKGap	when –kGap = 1, 256 features for DNA/RNA and 160 000 for protein	DNA/RNA/Protein
triDiKGap	when –kGap = 1, 1024 features for DNA/RNA and 3 200 000 for protein	DNA/RNA/Protein

computational biology. Over the past decade, several studies have proposed such toolkits that extract different feature sets for different sequence data such as protein and other peptides, RNAs, or DNAs (Cao *et al.*, 2013; Chen *et al.*, 2018; Liu, 2017; Liu *et al.*, 2015, 2017). Despite all the efforts have been made so far, still accessibility to the available methods is limited. Especially toolkits that provide features for prediction task involving proteins, DNAs and RNAs are crucial to develop due to involvement of data from various platform. Here we propose PyFeat as a comprehensive toolkit implemented in Python for generating various numerical feature presentation schemes from DNA, RNA and protein sequences. As a result, it can be widely used in different applications in bioinformatics and biological science. This tool is also able to select the best features among a large number of features that are generated mainly based on the Gap based feature extraction to provide useful local discriminatory information. Feature generation methods implemented in PyFeat aimed at capturing the frequency distributions of various permutations of the base nucleotides/amino acids in the sequences which in turn are able to represent the sequences in the model training process, efficiently. Here we also demonstrate the effectiveness of features generated by PyFeat to tackle challenging problems using different sequence data (DNAs, RNAs and Proteins) which for all three cases outperforms previously proposed models found in the literature (Liu, 2017).

2 Implementations

K-mer frequency is considered as an important method to extract local features. However, as the K (length of sub-sequences) increases, the number of features dramatical increases as well. Therefore, a small K can only be used to avoid sparsity (Ghandi *et al.*, 2014). For example, for DNA or RNA sequences K less than 5 is used. The case for proteins is even more dramatical as the number of produced features are by far more extensive due to the number of alphabets. To deal with this limitation, we focused on the concept of kGap to implement PyFeat (Cao *et al.*, 2013; Liu *et al.*, 2017). We have considered gaps in the nucleotide/amino acid sub-sequences and frequency of these sub-sequences are regarded as features for the prediction models. In PyFeat, the number of gaps is an argument that can be adjusted by users. When K is too large in kGap, it can produce large number of features. Therefore, implementing a feature reduction method is necessary to filter out informative and discriminatory features, and discard redundancy and noisy data to avoid sparsity and the curse of dimensionality. At the same time, it should be considered that discarding informative features may cause in loss of long-range sequence-order information resulting in underfitting or high bias.

In our experiments, we have incorporated features with kGap values ranging from 1 to 5 for DNA and RNA sequences, and values ranging from 1 to 10 for protein sequences. When the kGap value is small, length of the generated feature set is also small, and the occurrence frequency of these features retain local or short-range sequence order information while features with moderately large kGap values preserve global or long-range sequence-order information. Based on this analysis, we extract features from 13 different methods which are listed in Table 1. A full description of these feature extraction methods and the number of produced features is provided in a user-friendly manual that is available online as a supplementary material.

For feature selection and to reduce the impact of the curse of dimensionality and at the same time maintain informative features (Keogh and Mueen, 2017) we have employed AdaBoost classification model to calculate the average impurity-curtailment achieved by splitting upon each of the features in all of the trees trained on different weight distributions of the instances. We then select n features with the maximum score for model training. It is to be noted that this selection mechanism is much more cost-effective compared to the wrapper-based methods since only one run of the AdaBoost model is sufficient for the selection process. Moreover, it is more effective compared to the other methods as different trees incorporate different instance weight distributions into the impurity measure which in turn adds diversity to the way features are selected for node splitting in different trees, thus making the selection process less likely to be adversely affected by the presence correlated features having equally high predictive capability. It makes the choice of features diverse and robust in the presence of high feature multicollinearity (Wang, 2012). As a result, although the number of extracted features using each method can be very large, PyFeat is able to reduce dimensionality quite dramatically. The explanation of how to use feature extraction, adjusting K in KGap technique and model selection are available online.

3 Results and discussions

To demonstrate the effectiveness of features extracted using PyFeat, we used three different case studies for DNA, RNA and protein-based problems. We employed extracted features from PyFeat to predict three different tasks. For DNA, to predict Sigma70 promoter region (Lin *et al.*, 2017); for RNA, to predict adenosine (A) to inosine (I) site which direct the transcription of majority of the genes

(Chen *et al.*, 2016); and for proteins, to predict DNA binding proteins (Chowdhury *et al.*, 2017). Extracted features by PyFeat used for different classifiers to build the models. As a result, for all three cases, models built based on features extracted from PyFeat outperformed previous studies found in the literature. For Sigma70 promoter prediction task we achieved 92.88% prediction accuracy, for prediction of Adenosine (A) to Inosine (I) site task we achieved 88.50% prediction accuracy, and for prediction of DNA binding proteins we achieved 83.33% prediction accuracy which for all cases significantly outperform previous results found in the literature (Chen *et al.*, 2016; Liu *et al.*, 2017; Lin *et al.*, 2017; Chowdhury *et al.*, 2017). The full table and description of the problems are provided in supplementary material as well as PyFeat repository in GitHub.

These results demonstrate the effectiveness of extracted features as well as dimensionality reduction scheme provided in PyFeat. To the best of our knowledge, the collection of these feature extractions and dimensionality reduction methods have not been presented in such an integrated toolkit, before. In addition, the conducted experimentation for all three DNA, RNA and protein-based problems lead to results better than those reported in previous studies (Jani *et al.*, 2018).

In the future, we will integrate more feature extraction, feature reduction and analysis techniques to expand PyFeat to enable interactive analysis and machine learning-based modeling. PyFeat is expected to be widely used as a powerful tool in bioinformatics, computational biology and proteome research. PyFeat and its source code and manual are publicly available at: https://github.com/mrzResearchArena/PyFeat/.

Funding

Research reported in this publication was supported by the National Institute of General Medical Sciences of the National Institutes of Health under Award Number UL1GM118973.

Conflict of Interest: none declared.

References

Cao,D.S. *et al.* (2013) ProPy: a tool to generate various modes of chou's pseaac. *Bioinformatics*, **29**, 960–962.

Chowdhury,S.Y. *et al.* (2017) Idnaprot-es: identification of DNA-binding proteins using evolutionary and structural features. *Sci. Rep.*, **7**, 14938.

Chen,Z. *et al.* (2018) iFeature: a python package and web server for features extraction and selection from protein and peptide sequences. *Bioinformatics*, **1**, 2499–2502.

Chen,W. *et al.* (2016) Predicting adenosine to inosine editing sites by using pseudo nucleotide compositions. *Sci. Rep.*, **6**. https://www.nature.com/articles/srep35123.

Ghandi,M. *et al.* (2014) Enhanced regulatory sequence prediction using gapped k-mer features. *PLoS Comput. Biol.*, **10**, e1003711.

Jani,M.R. *et al.* (2018) iRecSpot-EF: effective sequence based features for recombination hotspot prediction. *Comput. Biol. Med.*, **103**, 17–23.

Keogh,E. and Mueen,A. (2017) Curse of dimensionality. *Encyclopedia Mach. Learn. Data Min.*, 314–315.

Liu,B. (2017) BioSeq-analysis: a platform for DNA, RNA and protein sequence analysis based on machine learning approaches. *Brief. Bioinf.* bbx165.

Liu,B. *et al.* (2015) Pse-in-one: a web server for generating various modes of pseudo components of DNA, RNA, and protein sequences. *Nucleic Acids Res.*, **43**, W65–W71.

Liu,B. *et al.* (2017) Pse-analysis: a python package for DNA/RNA and protein/peptide sequence analysis based on pseudo components and kernel methods. *Oncotarget*, **8**, 13338.

Lin,H. *et al.* (2017) Identifying sigma70 promoters with novel pseudo nucleotide composition. *IEEE/ACM Trans. Comput. Biol. Bioinf.* https://ieeexplore.ieee.org/document/7847385.

Wang,R. (2012) AdaBoost for feature selection, classification and its relation with SVM, a review. *Phys. Proc.*, **25**, 800–807.