Sequence analysis

pysster: classification of biological sequences by learning sequence and structure motifs with convolutional neural networks

Stefan Budach\textsuperscript{1,*} and Annalisa Marsico\textsuperscript{1,2,*}

\textsuperscript{1}Otto-Warburg-Laboratory, RNA Bioinformatics, Max Planck Institute for Molecular Genetics, Berlin 14195, Germany and \textsuperscript{2}Department of Mathematics and Computer Science, Free University Berlin, Berlin 14195, Germany

\*To whom correspondence should be addressed.

Associate Editor: John Hancock

Received on December 7, 2017; revised on March 21, 2018; editorial decision on April 3, 2018; accepted on April 5, 2018

Abstract

Summary: Convolutional neural networks (CNNs) have been shown to perform exceptionally well in a variety of tasks, including biological sequence classification. Available implementations, however, are usually optimized for a particular task and difficult to reuse. To enable researchers to utilize these networks more easily, we implemented pysster, a Python package for training CNNs on biological sequence data. Sequences are classified by learning sequence and structure motifs and the package offers an automated hyper-parameter optimization procedure and options to visualize learned motifs along with information about their positional and class enrichment. The package runs seamlessly on CPU and GPU and provides a simple interface to train and evaluate a network with a handful lines of code. Using an RNA A-to-I editing dataset and cross-linking immunoprecipitation (CLIP)-seq binding site sequences, we demonstrate that pysster classifies sequences with higher accuracy than previous methods, such as GraphProt or ssHMM, and is able to recover known sequence and structure motifs.

Availability and implementation: pysster is freely available at https://github.com/budach/pysster.

Contact: budach@molgen.mpg.de or marsico@molgen.mpg.de

Supplementary information: Supplementary data are available at Bioinformatics online.

1 Introduction

In recent years, deep convolutional neural networks (CNNs) have been shown to be an accurate method for biological sequence classification and sequence motif detection (Alipanahi et al., 2015; Angermueller et al., 2017; Kelley et al., 2016). The increasing amount of sequence data and the rise of general-purpose computing on graphics processing units (GPUs) have enabled CNNs to outperform other machine learning methods, such as random forests and support vector machines, in terms of both classification performance and runtime performance (Angermueller et al., 2017; Kelley et al., 2016). While a number of publications have made use of CNNs on biological data, these implementations are usually hard to reuse (Alipanahi et al., 2015) or tailored to a specific problem, such as prediction of DNA CpG methylation from single-cell data (Angermueller et al., 2017). Basset (Kelley et al., 2016) and iDeep (Pan and Shen, 2017) represent more general frameworks for training of CNNs on DNA and RNA data, respectively. However, they don’t provide detailed motif interpretations, such as motif locations and class enrichment of motifs, and they are not able to learn structure motifs in the RNA case. For the specific task of classifying RNA by learning sequence and structure motifs other tools not based on neural networks are available, e.g. RNAcontext (Kazan et al., 2010) and GraphProt (Maticzka et al., 2014). These tools usually learn only a single motif and similar to CNN-based methods provide no additional information such as motif locations.

To address these issues and to enable researchers to easily utilize CNNs, we implemented pysster, a python package for training CNN classifiers on biological sequences. Supervised classification is enabled by the automatic detection of sequence motifs. Our package focuses on interpretability and extends previous implementations by...
providing information about the positional and class enrichment of learned motifs. Moreover, by incorporating structure information, e.g. in the form of secondary structure predictions for RNA sequences, it is possible to learn structure motifs corresponding to the sequence motifs. We demonstrate that our tool is able to learn well-known motifs for an RNA A-to-I editing dataset and multiple CLIP-seq datasets and that it outperforms GraphProt, a state-of-the-art classifier for RNA sequences and structures, both in terms of classification and runtime performance. Providing a simple programming interface we hope that our package enables more researchers to make effective use of CNNs in classifying and interpreting large sets of biological sequences.

2 Implementation and features

We implemented an established network architecture and multiple interpretation options as an easy-to-use python package. The basic architecture of the network consists of a variable number of convolutional and max-pooling layers followed by a variable number of dense layers interspersed by dropout layers. The network can be tuned via an extensive list of hyper-parameters. The network inputs are one-hot encoded sequences and the network outputs predicted probabilities, indicating class membership. (A) The basic network architecture consists of a variable number of convolutional/max-pooling stacks followed by a variable number of dense layers interspersed by dropout layers. The network can be tuned via an extensive list of hyper-parameters. The network inputs are one-hot encoded sequences and the network outputs predicted probabilities, indicating class membership. (B) RNA sequence and structure input strings are encoded into a single string by combining the sequence alphabet and the secondary structure alphabet into an extended alphabet consisting of arbitrary characters. Subsequently, this string is one-hot encoded and used as the network input. (C) For the motif interpretation the string over the arbitrary alphabet can be decoded into the two original strings to construct sequence logos for the original alphabets. The shown example motif corresponds to an ALU repeat motif found in the classification task of RNA A-to-I editing sites (see the tutorial workflow on github for detailed information).

optional use of structure information, handcrafted features and recurrent layers

visualization of all network layers using visualization by optimization (Olah et al., 2017)

seamless CPU or GPU computation by building on top of TensorFlow (Abadi et al., 2016)

Structure information (e.g. for RNA in the form of dot-bracket strings or annotated dot-bracket strings) is incorporated into the network by encoding the given sequence string over an alphabet of size $N$ and the corresponding structure string over an alphabet of size $M$ into a single new string using an extended alphabet of size $N \times M$ (Fig. 1B). The network is then trained on these new strings, which can be decoded back into the original strings after the training to enable the visualization of two motifs as position-weight matrices (Fig. 1C). Detailed descriptions of the model and visualization options can be found in the tutorials and documentation.

3 Case studies

pysster is freely available at https://github.com/budach/pysster. Its documentation includes a workflow tutorial that showcases the main functionality on an RNA A-to-I editing dataset (Picardi et al., 2017). Editing sites are known to be enriched in repetitive ALU sequences and we show that we are able to classify the editing location with high accuracy and that we learn known ALU motifs (Fig. 1C). Finally, we have trained our tool on sequences derived from CLIP-seq data for a number of RNA binding proteins with known binding site motifs (Supplementary Table S1). Similar to GraphProt and ssHMM (Heller et al., 2017), an unsupervised hidden Markov model-based approach for learning sequence/structure motifs, pysster can recover the known motifs but outperforms GraphProt both in terms of classification and runtime performance.

Conflict of Interest: none declared.
References