Improving Codon Optimization with RNN

ADITYA JAIN, EC552 FINAL PROJECT, 2020

PROJECT SUMMARY

Synthetic plasmid creation and inter-organism plasmid transfer is a cornerstone of synthetic biology. In creating protein sequences, there are 64 potential codons but only 22 amino acids. This is because many codons are synonymous, meaning they code for the same amino acid. Recent research has shown that synonymous codons are not redundant; using specific codons has major effects on protein expression due to effects on the transcription & translation stages, causing up to a 100-fold change in expression. Further, different organisms have a different codon bias for which synonymous codons they use. Yet, the industry standard codon optimization tools typically select only high frequency codons, discounting the effects of rare codons. In the design and manufacturing of synthetic plasmids, a tool for codon optimization which can better match the host genome would be valuable in time & costs saved by creating more efficient plasmids. I developed ICOR, a novel codon optimization tool which uses recurrent neural networks (RNN) to learn information about the sequence in which amino acids appear with the hypothesis that this sequence affects which synonymous codon to use. I created a robust database of over 35,000 non-redundant proteins from the NCBI E. Coli Genome archives, generalized across all species of E. coli and trained ICOR to an accuracy of around 54%, 2.5x higher than random selection of a synonymous codon. This research provides compelling evidence that the sequence of amino acids can be learned and this context can yield codon selection that is more similar to the host genome, improving efficacy, reducing likelihood of plasmid toxicity, and reducing costs.

SOFTWARE FUNCTIONALITY

The development of ICOR has two major software components for the user: ICORnet architecture and user application. The ICORnet architecture is a Long Short Term Memory (LSTM) type of RNN. It was trained & validated on the 35,000+ non-redundant protein database and serves as the brain for the codon optimization tool. By providing the amino acid sequence as an input, ICORnet is able to output a nucleotide codon sequence which would ideally match the codon biases of the host genome. The user application provides a GUI for the user to input the sequences they would like to optimize and receive optimized codon sequences. The user application is lightweight and is able to process codon sequences at a rate which matches/beats many of the leading industry applications. The application was designed so improvements to the ICORnet model can be easily accessed by the user without having to update their local application of ICOR. There are many scripts which were written to process the database and train ICOR, but the key steps are as follows:

1. Fastalator.m loads DNA sequences from FASTA files and converts them into structures containing Amino Acid sequence and the original DNA sequence. It is able to account for DNA sequences which do not have 100% confidence (occurs due to sequencing) by using IUPAC probability estimators.

2. CreateTrainingData.m & CreateTrainingDataNLF.m are two scripts which encode the amino acid data into vectors for the ICORnet RNN model. One uses One Hot Encoding while the other uses Non-Linear Fischer Transform. The current version of ICORnet uses One Hot Encoding.
3. trainNet trains ICORnet on the data imported. It provides options to adjust hyper-parameterization and the network architecture. It outputs a Matlab network object which can then be used for classification using the classify function or through the ICORnet application.

These scripts were used for development. In a real user setting, the user would simply install the ICOR program, download the latest version of ICORnet, and select the sequence they would like to process through the GUI.

SPECIAL INSTRUCTIONS

The code uploaded to the Github repository for this project contain all the necessary files to run ICORnet on new data and even to compile statistics on this new data. The repository also contains the files needed to train the ICORnet on new data. All instructions for how to go through this process are included in the README.md file on the Github.

The data used to train ICOR for this project were not included due to the restrictive file size. Further, the database is being thoroughly verified towards potential publication. It is recommended that any user who wants to replicate the project compile the dataset from the NCBI Archives according to the potential publication.