ICOR: Improving codon optimization with recurrent neural networks

Rishab Jain[1], Aditya Jain[2], Elizabeth Mauro[1], Kevin LeShane[1], Douglas Densmore[1,3,4]

1. Lattice Automation, 709 E 5th St. #3, Boston, MA 02127
2. Harvard Medical School, 25 Shattuck St, Boston, MA 02115
3. Department of Electrical and Computer Engineering, Boston University, 8 Saint Mary's St. Boston MA, 02215
4. Biological Design Center, Boston University, 610 Commonwealth Ave., Boston MA, 02215

Abstract

In protein sequences—as there are 61 sense codons but only 20 standard amino acids—most amino acids are encoded by more than one codon. Although such synonymous codons do not alter the encoded amino acid sequence, their selection can dramatically affect the expression of the resulting protein. Codon optimization of synthetic DNA sequences is important for heterologous expression. However, existing solutions are primarily based on choosing high-frequency codons only, neglecting the important effects of rare codons. In this paper, we propose a novel recurrent-neural-network based codon optimization tool, ICOR, that aims to learn codon usage bias on a genomic dataset of *Escherichia coli*. We compile a dataset of over 7,000 non-redundant, high-expression, robust genes which are used for deep learning. The model uses a bidirectional long short-term memory-based architecture, allowing for the sequential context of codon usage in genes to be learned. Our tool can predict synonymous codons for synthetic genes toward optimal expression in *Escherichia coli*. We demonstrate that sequential context achieved via RNN may yield codon selection that is more similar to the host genome, therefore improving protein expression more than frequency-based approaches. ICOR is evaluated on 1,481 *Escherichia coli* genes as well as a benchmark set of 40 select DNA sequences whose heterologous expression has been previously characterized. ICOR's performance across five metrics is compared to that of five different codon optimization techniques. The codon adaptation index -- a metric indicative of high real-world expression -- was utilized as the primary benchmark in this study. ICOR is shown to improve the codon adaptation index by 41.69% and 17.25% compared to the original and Genscript's GenSmart-optimized sequences, respectively. Our tool is provided as an open-source software package that includes the benchmark set of sequences used in this study.

Introduction

Designing synthetic genes for heterologous expression is a keystone of synthetic biology(1). Expressing recombinant proteins in a heterologous host has applications from recombinant pharmaceuticals to vaccine manufacturing. For instance, producing malaria vaccine FALVAC-1(2) involves designing synthetic plasmids, transfection into the *Escherichia coli* (*E. coli*) host factory, growing up the cells, and harvesting the resulting protein(3). *E. coli* is well-established as a host factory for recombinant protein production, resulting in greater availability of codon optimization tools(4). To increase the efficacy of recombinant technology further, improving expression output is an area of particular interest.

Although the expression levels of synthetic genes when introduced recombinantly may be dependent on many factors, one important factor is the codon usage bias in comparison to the host(5). During translation, complimentary tRNAs are used to read codons from the mRNA strand(6). Relative abundance of tRNA leads to variable gene expression, where up to 1000-fold increases have been observed(6). The frequency of certain synonymous codons found in an organism's genome is positively correlated with its tRNAs(7)(8). Thus, by choosing synonymous codons that are more frequently found in the host organism's genome, there are more amino acids available during translation. Although such synonymous codons may code for the same amino acid, they are not redundant(9)(10). In a study by Gao et al., low expression levels of human immunodeficiency virus genes in mammalian cells are attributed to rare codon usage(11). *E. coli* and the Chinese Hamster Ovary (CHO) cell are a few of many host cells (chassis) that are used to produce recombinant technology-based biopharmaceuticals and vaccines(12). However, it is important to understand the underlying codon usage bias of the chassis to maximize protein expression.

Today, there are a range of FDA-approved recombinant DNA products from synthetic insulin to Hepatitis B therapeutics(12). Codon optimization tools can be used to increase protein expression and therefore the efficiency of manufacturing for such products(13). Industry-standard codon optimization techniques based on biological indexes replace synonymous codons with the most abundant codon found in the host organism's genome(14). Our review shows that many tools employ the aforementioned strategy, causing unintended consequences for the cell, such as creating an imbalanced tRNA pool(9)(15). If just one codon of a synonymous set is used, metabolic stress and translational error may be imposed on the cell(9). Recent research has shown that some rare codons play an important role in protein folding(16).

Understanding the context by which synonymous codons are used in a gene is essential to truly unlock the full potential that evolution has encoded in genes. By predicting synonymous codons based on their sequential context in the host organism's genome, protein expression can be increased while preventing translational error and plasmid toxicity. To best learn sequential and contextual patterns of the host, deep learning can be leveraged for its high level of abstraction on large datasets(17). Deep learning systems show promise in bioinformatics, potentially offering improvements over non-machine learning algorithms(18).

Recurrent neural networks (RNNs) are a class of deep neural networks that can grasp temporal data, thus demonstrating utility in applications with sequential information(19). For example, speech recognition models that utilize long short-term memory (LSTM) architectures(20)—a type of RNN—take advantage of the memory built into the LSTM module allowing it to interpret speech based on the surrounding context. For codon optimization, RNNs may offer improved synonymous codon selection because the RNN is designed to understand the characteristics and patterns of the data to inform subsequent synonymous codon prediction. By treating each amino acid as a timestep in the sequence, the RNN evaluates its prediction in the context of the amino acids before and after it.

In this study, a deep learning tool, ICOR, is trained on a large, robust, non-redundant dataset of *E. coli* genomes. This "big data" approach allows our model to learn codon usage across multitudinous genes of *E. coli* and develop a generalized, quantified model for rare codon usage and advantages instilled through evolution. ICOR adopts the Bidirectional Long-Short-Term Memory (BiLSTM) architecture(21) because of its ability to preserve temporal information from both the past and future.

40 reference synthetic genes as shown in Table S1 along with 1,481 *E. coli* genes were used for: testing and evaluation, allowing us to evaluate the model in two ways: its ability to replicate but optimize *E. coli* sequences, and its ability to optimize genes commonly used in synthetic plasmids for recombinant products. The resultant optimized sequences from the ICOR model were compared to five approaches (original, super naïve, naïve, brute-force, GenScript's GenSmart(22)) as outlined in S1 Appendix. GenSmart is defined as the industry-standard benchmark due to its recognition as in past studies(23), ease-of-use, and accessibility. Along with codon adaptation index (CAI) as a primary metric, GC content, codon frequency distribution (CFD), negative repeat elements, negative CIS elements, and algorithm run-time are quantified as secondary metrics to gauge the performance of the approaches. The ICOR tool is open-source and can be accessed at: https://doi.org/10.5281/zenodo.5529209.

Materials and Methods

Model Training Dataset

We use the National Center for Biotechnology Information's *E. coli* genome dataset(24) which included 6,877,000 genes. Genes that were under 90 amino acids in length were removed due to their hypothetical or specialized nature. Further, CD-HIT-EST(25) was utilized to cluster and remove similar nucleotide sequences by sequence identity of over 90%. After curation and removal of redundant genes, 42,266 sequences remained, of which 7,406 high-expression sequences were used as a model dataset.

Approximately 70% of the dataset was used for training (5,184 sequences), 10% for development or validation (741 sequences), and 20% for testing (1,481 sequences).

Synthetic Plasmid Benchmarks

40 DNA sequences were established as a benchmark set, extracted from studies conducted on codon optimization and evaluating the gene expression of plasmids in *E. coli.* The benchmark set serves as a validation for the effectiveness of the tool on genes whose heterologous expression has been studied. The resultant coding regions of the sequences can be accessed at https://doi.org/10.5281/zenodo.5529209 and their descriptions in Table S1.

Encoding

Encodings were created for our entire dataset using the "one-hot encoding" technique. Amino acid sequences were converted into integers and then placed into vectors that are 26 features long. At each timestep, the present amino acid is encoded into the vector as "1" while all other features are set to "0". For example, the amino acid Alanine can be represented by a 1x26 vector in which the first element is 1 and all other elements (features) are set to 0. Features were based on Table S2.

Model Building

Sequential information may yield synonymous codon selection that is more similar to the host organism. Deep learning is a technique that may be able to capture underlying patterns found in the host genome. Our model uses the BiLSTM architecture which predicts synonymous codons given the input amino acid sequence. The model hyperparameters were tuned iteratively when trained on the training and validation subsets of the dataset. L2 regularization and dropout were used to fine-tune our model and prevent overfitting. This model building overview along with a user workflow is depicted in Fig 1.

Fig 1. User workflow for sequence codon optimization using ICOR deep learning model with overview of model creation. (A) On the left-hand side, a user workflow towards creating a vector for heterologous expression is depicted. (B) On the right-hand side, expanding out of "sequence codon optimization" in the user workflow, the overview of the ICOR model creation is given. In a production setting, a trained and packaged model of ICOR is inferenced.

The ICOR model's architecture consists of a 12-layer recurrent neural network as visualized in Figure S1. Data is fed forward from the first layer—Sequence Input—to the last layer—Classification—from top to bottom. The model was trained in the MATLAB r2020b(26) on the Tesla V100 graphics card. The model was trained on 50 epochs and took 138 minutes to complete training.

Hyperparameter Tuning

We experimented with many variables, such as mini-batch size, gradient decay, and model architecture as displayed in Table 1.

Hyperparameter	**Value of Parameter**
Gradient Decay Factor (dictates the decay rate of gradient moving average)	0.95
Learning Rate (the frequency at which weights are updated during training, also known as step size)	5×10^{-5}
L2 Regularization (factor of weight decay)	1e-8
Shuffling (shuffles the order of training data)	never
Dropout (the factor by which neurons are temporarily ignored during training)	0.2
Padding (ensures a ragged input—sequences are not padded to a uniform length)	off
Sorting (sort input by ascending sequence length)	on
Optimizer (the solver algorithm used when	adam (adaptive moment estimation)

training the network)	

Table 1. Hyperparameter variables are used to finetune the results of the model. (A) In the left-hand column, 8 hyperparameters are listed. (B) In the right-hand column, their respective settings are given based on the testing and fine-tuning conducted in this research. Our iterative tuning methods reach the following hyperparameters, which prevent the possibility of over/underfitting while maintaining high performance.

The need to make trade-offs with respect to hidden units—corresponding to the quantity of information stored between network states —became evident. Initially, we found our model to be underfitting. In order to make it more complex, we made our model architecture more complex, and also increased the number of hidden units. Further, when the learning rate was too high, the model very quickly plateaued. In order to force the model to train for a longer period of time, this value was adjusted along with the epochs. Changing the value of the mini-batch-size was a matter of balancing memory for performance. The final model was trained across 50 epochs, with a minibatch size of 512, and 256 hidden units.

Software Architecture

The development of ICOR has two major software components for the user: ICORnet architecture and runtime scripts. The ICORnet architecture is the trained BiLSTM network. It serves as the brain for the codon optimization tool. By providing the amino acid sequence as an input, ICORnet can output a nucleotide codon sequence that would ideally match the codon biases of the host genome. The key steps of development are as follows:

1. Fastalator loads DNA sequences from FASTA files and converts them into structures containing amino acid sequences and the original DNA sequence. It can account for DNA sequences that do not have 100% confidence (occurs due to sequencing) by using IUPAC probability estimators.

2. CreateTrainingData and CreateTrainingDataNLF are two scripts that encode the amino acid data as vectors for the ICORnet RNN model. One uses One-Hot Encoding while the other uses Non-Linear Fisher Transform (NLFT).

3. trainNet trains ICORnet on the data imported. It provides options to adjust hyperparameterization and the network architecture. It outputs a network object which can then be used for classification using the classify function or through the ICORnet user application.

Given an input of sequences they would like to optimize, a user would receive optimized codon sequences for *E. coli* expression using the runtime scripts integrated with the ICOR model. The runtime scripts utilize the ONNX runtime to run inference on the trained MATLAB model. This system was designed modularly so improvements to the ICORnet model can be easily accessed by the user without the need to have a packaged execution. Rather, the ICORnet model can be downloaded and executed using the runtime workflow.

Statistical Analysis

We use the CAI as a primary metric, and GC content, CFD, negative repeat elements, and negative CIS elements as secondary metrics to conduct statistical analysis. The CAI is calculated using the formulae described in S1 Supporting Information. GenScript's rare codon analysis tool(27) is utilized to calculate the secondary metrics. The mutational rate is quantified by conducting optimization on the test dataset, converting the optimized codons back to amino acids, and then counting the number of amino acids that varied between them. Rare codon usage was qualitatively and quantitatively compared to reference

tables(28). During validation, codon accuracy, amino acid accuracy, and base errors were measured. By comparing the similarity of the optimized sequence to the input, these accuracies can be calculated.

Results

Codon Prediction with Deep Learning

We use the Codon Adaptation Index(29) primarily to quantify the performance of our tool. As noted in previous studies, CAI is highly correlated with real-world expression(30). On the test subset of 1,481 genes, we find that ICOR offered an improvement in CAI from 0.73 to 0.889 ± 0.012, or about 29.1% compared to the original sequences (Fig 2).

Fig 2. ICOR significantly improves CAI when compared to original sequences on a test subset of 1,481 genes. Box and Whisker Plot (n=1481) comparing CAI (left: original sequences, right: ICOR optimized sequences). The y-axis is the codon adaptation index on a scale from 0.55 to 0.95. The open points outside of the boxes are outliers that are beyond 1.5 times the interquartile range. The horizontal divisions present in each box (from top to bottom) are the upper whisker, 3rd quartile, median, 1st quartile, and lower whisker.

In order to properly contextualize the performance of the developed model, five algorithms from S1 Appendix were used to generate optimized sequences from the original benchmark sequences. These proteins came from a variety of origin organisms and had a mean CAI of 0.638 with a standard deviation of 0.0386. ICOR optimization resulted in a mean CAI of 0.904 with a standard deviation of about 0.016, signifying a ~41.692% increase. The super naive approach had a mean CAI of 0.602 and standard deviation of about 0.022. ICOR offered a ~50.21% increase compared to this approach. Finally, the naive approach offered a mean CAI of 0.699 and a standard deviation of 0.0158. ICOR offered a ~29.32%

increase in CAI compared to the naive approach. These comparisons were statistically significant ($p<0.0001$) using a two-sample t-test. The mean CAI for all approaches is shown in Fig 3.

Fig 3. ICOR significantly improves CAI when compared to the original, naïve, super naïve, brute force, and GenSmart techniques. Box and Whisker Plot (n=40) comparing CAI with legend indicating the color for each optimization method.

When extrapolating such improvements to findings by dos Reis et al. on the correlation between CAI and expression for group 1 (biased) genes, real-world mRNA expression could improve by an average of 236%(30).

The secondary endpoints were also quantified for each of the codon optimization tools on the benchmarking dataset. As depicted by Figure B in S2 Appendix, there was a statistically insignificant difference ($p=0.726$) between ICOR and GenSmart in the number of negative CIS elements in the optimized sequences using the Mann-Whitney U Test for non-parametric distributions. When computing the mean change in negative CIS elements between the optimization tool and the original sequence, there is a statistically insignificant difference between ICOR and GenSmart. This suggests that ICOR maintains equally low negative CIS elements as GenSmart while achieving a higher CAI. As depicted by Figure C in S2 Appendix, there was a trending difference ($p=0.1826$) between ICOR and GenSmart in the number of negative repeat elements in the optimized sequences using the Mann-Whitney U Test for non-parametric distributions. This suggests that although ICOR may have higher negative repeat elements, the difference is minimal as compared to GenSmart.

Optimization Run Time

The run time was calculated for the approaches where inference time could be isolated. Using a testing system as described in S1 Supporting Information, the algorithms were evaluated for run time on the

benchmark set with an average length of 1687.65 nucleotides per sequence. The scores normalized to the super naïve approach are displayed in Table 2.

	Average Time (n. seq=40) (n. trials=3)	Time Per Seq.	Time Per Codon	Time per NT
Brute Force	**275587%** (749597ms) (750.517s, 749.821s, 748.455s)	**18739.9ms**	**1332.50ms**	**444.166ms**
ICOR	**288%** (782ms) (0.772s, 0.807s, 0.766s)	**19.5ms**	**1.39ms**	**0.463ms**
Naive	**283%** (771ms) (0.761s, 0.764s, 0.787s)	**19.3ms**	**1.37ms**	**0.456ms**
Super Naive	**100%** (272ms) (0.271s, 0.274s, 0.272s)	**6.8ms**	**0.48ms**	**0.161ms**

Table 2. ICOR runtime comparable to naïve approaches and significantly faster than brute force approach. Three trials were conducted with the score being an average of these three times measured to the third significant figure in milliseconds. Optimization run time is displayed with a normalized percentage to the super naïve approach. Anecdotally, the GenSmart optimization took approximately 22 minutes, however, this measurement was not included in the table due to the tool being in a production environment with a queue of jobs.

Gene Mutations

Deep learning has the capability to learn, classify, and predict large amounts of data. However, as with any algorithm, it may not be perfect. In the case that the current amino acid (i.e., Glutamine) is replaced with a codon for an entirely different amino acid, this would result in at least one point mutation. When introduced inside of the host, detrimental effects to gene expression would ensue.

During our testing, it was found that encoding techniques made a significant difference. The One-Hot Encoding technique offered approximately a 10% improvement over NLFT. Theoretically, the One-Hot Encoding technique will be able to learn the amino acid context faster, because the input data is simpler: a 1x26 array with a "1" in the position of where the amino acid is. On the test dataset of 1,481 genes, our model yielded a 0.00% mutational rate and 100% amino acid accuracy. Thus, it was found that gene mutations are not present in our codon optimization technique.

Discussion

In this research, we propose ICOR, a novel codon optimization tool that uses recurrent neural networks towards improving heterologous expression for synthetic genes. We find that deep learning is a particularly effective method in the synonymous codon optimization area because it can contextualize a DNA sequence within the host genome to understand synonymous codon usage bias for high expression genes and specific subsequences. While previous research ranges from selecting high-frequency genes, to eliminating secondary structures, to machine learning models and convolutional neural network architectures, we use the RNN architecture which has the ability to analyze sequential data. By understanding the underlying patterns in the host genome through the use of an RNN, codon selection may be more similar. Using this approach, we built the ICOR model using a large dataset of 7,406 non-redundant genes from the *E. coli* genome. Having a non-redundant dataset was of vital importance in this study as many *E. coli* genomes will contain extremely similar genes. We used the CD-HIT-EST server to overcome this issue, as training a model on redundant data would yield codon selection that is biased to frequent genes only. Further, this helped reduce the necessary compute resources required for deep learning.

This research also demonstrates that encoding gene sequences using Natural Language Processing techniques is of particular interest. We use the one-hot encoding method in our final model. However, we also tested the NLFT based on the physicochemical properties of amino acids. One-hot encoding offered approximately a 10% improvement over NLFT. Theoretically, this is because when training a model using one-hot encoded data, it can immediately identify the amino acid labels. Thus, it can very quickly start to work on the problem of predicting codons from the simple amino acid labels. On the other hand, NLFT may take a longer time to do so due to its unique feature set.

The statistical data analysis primarily uses the Codon Adaptation Index as a way to directly compare ICOR to other solutions in addition to quantifying the accuracy of the tool on the test dataset. Although increases in CAI are strongly correlated with increased protein expression, it is not the only comprehensive statistic to quantify gene expression. Recent research has shown that creating models for measuring translation dynamics is possible, and such research can be applied to the results of our study. Modeling elongation rate, tRNA adaptation index, and other metrics may provide valuable insight into the results of the tool.

In this research, five codon optimization techniques were evaluated. Of these, the brute force algorithm attempts to illustrate the efficacy of an exhaustive optimization in which all potential sequences are generated. Due to computational limitations, a truly exhaustive search method in which every possible gene sequence is evaluated cannot be efficacious. Given an average sequence length of 562.55 amino acids and an average of 2.91 potential codons, over 1.01×10^8 possible sequences would exist, requiring an infeasible amount of time (over 524,000 hours) to calculate. Such a method would operate under the assumption that there is a single metric that requires optimization. However, biological context suggests that this assumption is erroneous, and therefore, codon optimization based on biological context is most effective.

ICOR codon optimization is competitive and can be applied directly in synthetic gene design. Synonymous codons can be optimized to increase resultant protein expression. Thus, the efficiency of production improves, potentially decreasing the cost of the product. Currently, the ICOR tool can be accessed through an open-source software package, however, we would like to build an API to improve accessibility in the future.

Although our model is based on the *E. coli* bacterium, it may be plausible to apply our methodology to other organisms such as yeast and mammalian cells in future research. A transfer learning approach may allow us to preserve our pre-trained model, and adapt it to other host cells. Additionally, we would like to add the ability for our model to optimize other regions of a gene such as promoter sequences. Research aimed at analyzing what sub-sequence properties are learned by the model to make predictions may be biologically relevant. Finally, to further verify confidence in the results, future research may consider testing our model in the lab setting.

References

1. Endy D. Foundations for engineering biology. Nat 2005 4387067 [Internet]. 2005 Nov 24 [cited 2021 Oct 31];438(7067):449–53. Available from: https://www.nature.com/articles/nature04342

2. Zhou Z, Schnake P, Xiao L, Lal AA. Enhanced expression of a recombinant malaria candidate vaccine in Escherichia coli by codon optimization. Protein Expr Purif. 2004 Mar 1;34(1):87–94.

3. Nascimento IP, Leite LCC. Recombinant vaccines and the development of new vaccine strategies [Internet]. Vol. 45, Brazilian Journal of Medical and Biological Research. Associação Brasileira de Divulgação Científica; 2012 [cited 2021 Mar 23]. p. 1102–11. Available from: /pmc/articles/PMC3854212/

4. Mitchell AM, Gogulancea V, Smith W, Wipat A, Ofiţeru ID. Recombinant Protein Production with Escherichia coli in Glucose and Glycerol Limited Chemostats. Appl Microbiol. 2021;1(2):239–54.

5. Zhoua Z, Danga Y, Zhou M, Li L, Yu CH, Fu J, et al. Codon usage is an important determinant of gene expression levels largely through its effects on transcription. Proc Natl Acad Sci U S A [Internet]. 2016 Oct 11 [cited 2021 Mar 23];113(41):E6117–25. Available from: www.pnas.org/cgi/doi/10.1073/pnas.1606724113

6. Gustafsson C, Govindarajan S, Minshull J. Codon bias and heterologous protein expression [Internet]. Vol. 22, Trends in Biotechnology. Elsevier; 2004 [cited 2021 Mar 23]. p. 346–53. Available from: http://www.cell.com/article/S0167779904001118/fulltext

7. Brule CE, Grayhack EJ. Synonymous Codons: Choose Wisely for Expression. Trends Genet [Internet]. 2017 Apr 1 [cited 2021 Mar 23];33(4):283–97. Available from: http://www.cell.com/article/S0168952517300227/fulltext

8. Ikemura T. Correlation between the abundance of Escherichia coli transfer RNAs and the occurrence of the respective codons in its protein genes: A proposal for a synonymous codon choice that is optimal for the E. coli translational system. J Mol Biol [Internet]. 1981 Sep 25 [cited 2021 Mar 23];151(3):389–409. Available from: https://pubmed.ncbi.nlm.nih.gov/6175758/

9. Villalobos A, Ness JE, Gustafsson C, Minshull J, Govindarajan S. Gene Designer: A synthetic biology tool for constructuring artificial DNA segments. BMC Bioinformatics. 2006 Jun 6;7.

10. Plotkin JB, Kudla G. Synonymous but not the same: The causes and consequences of codon bias [Internet]. Vol. 12, Nature Reviews Genetics. Nature Publishing Group; 2011 [cited 2021 Mar 23]. p. 32–42. Available from: www.nature.com/reviews/genetics

11. Gao W, Rzewski A, Sun H, Robbins PD, Gambotto A. UpGene: Application of a web-based dna

codon optimization algorithm. Biotechnol Prog [Internet]. 2004 Mar [cited 2021 Mar 23];20(2):443–8. Available from: https://pubmed.ncbi.nlm.nih.gov/15058988/

12. Sanchez-Garcia L, Martín L, Mangues R, Ferrer-Miralles N, Vázquez E, Villaverde A. Recombinant pharmaceuticals from microbial cells: A 2015 update [Internet]. Vol. 15, Microbial Cell Factories. BioMed Central Ltd.; 2016 [cited 2021 Mar 23]. p. 33. Available from: /pmc/articles/PMC4748523/

13. Mauro VP, Chappell SA. A critical analysis of codon optimization in human therapeutics [Internet]. Vol. 20, Trends in Molecular Medicine. Elsevier Ltd; 2014 [cited 2021 Mar 23]. p. 604–13. Available from: /pmc/articles/PMC4253638/

14. Tian J, Li Q, Chu X, Wu N. Presyncodon, a web server for gene design with the evolutionary information of the expression hosts. Int J Mol Sci. 2018;19(12).

15. Puigbò P, Guzmá E, Romeu A, Garcia-Vallvé S. OPTIMIZER: a web server for optimizing the codon usage of DNA sequences. Nucleic Acids Res [Internet]. 2007 [cited 2021 Mar 23];35. Available from: http://genomes.urv.es/OPTIMIZER.

16. Chaney JL, Steele A, Carmichael R, Rodriguez A, Specht AT, Ngo K, et al. Widespread position-specific conservation of synonymous rare codons within coding sequences. Wilke CO, editor. PLOS Comput Biol [Internet]. 2017 May 5 [cited 2021 Mar 23];13(5):e1005531. Available from: https://dx.plos.org/10.1371/journal.pcbi.1005531

17. Miotto R, Wang F, Wang S, Jiang X, Dudley JT. Deep learning for healthcare: Review, opportunities and challenges. Brief Bioinform [Internet]. 2017 May 30 [cited 2021 Mar 23];19(6):1236–46. Available from: /pmc/articles/PMC6455466/

18. Tang B, Pan Z, Yin K, Khateeb A. Recent Advances of Deep Learning in Bioinformatics and

Computational Biology. Front Genet. 2019;0(MAR):214.

19. Liu P, Qiu X, Huang X. Recurrent Neural Network for Text Classification with Multi-Task Learning.

20. Hochreiter S, Schmidhuber J. Long Short-Term Memory. Neural Comput. 1997;9(8):1735–80.

21. Schuster M, Paliwal KK. Bidirectional recurrent neural networks. IEEE Trans Signal Process. 1997;45(11):2673–81.

22. GenSmart™ Codon Optimization Tool-GenScript [Internet]. [cited 2021 Oct 3]. Available from: https://www.genscript.com/gensmart-free-gene-codon-optimization.html

23. Koblan LW, Doman JL, Wilson C, Levy JM, Tay T, Newby GA, et al. Improving cytidine and adenine base editors by expression optimization and ancestral reconstruction. Nat Biotechnol [Internet]. 2018 Oct 1 [cited 2021 Oct 3];36(9):843. Available from: /pmc/articles/PMC6126947/

24. National Center for Biotechnology Information. Genome Escherichia coli. Bethesda [Internet]. 2021. Available from: ftp://ftp.ncbi.nlm.nih.gov/genomes/refseq/bacteria/Escherichia_coli

25. Huang Y, Niu B, Gao Y, Fu L, Li W. CD-HIT Suite: a web server for clustering and comparing biological sequences. Bioinformatics [Internet]. 2010 Mar 1 [cited 2021 Mar 23];26(5):680–2. Available from: https://academic.oup.com/bioinformatics/article/26/5/680/212234

26. MATLAB. version 7.10.0 (R2010a). Natick, Massachusetts: The MathWorks Inc.; 2010.

27. Rare Codon Analysis Tool [Internet]. [cited 2021 Oct 3]. Available from: https://www.genscript.com/tools/rare-codon-analysis

28. Kane JF. Effects of rare codon clusters on high-level expression of heterologous proteins in

Escherichia coli. Curr Opin Biotechnol. 1995 Jan 1;6(5):494–500.

29. Sharp PM, Li WH. The codon adaptation index-a measure of directional synonymous codon usage bias, and its potential applications. Nucleic Acids Res [Internet]. 1987 Feb 11 [cited 2021 Mar 23];15(3):1281–95. Available from: https://academic.oup.com/nar/article/15/3/1281/1166844

30. dos Reis M, Wernisch L, Savva R. Unexpected correlations between gene expression and codon usage bias from microarray data for the whole Escherichia coli K-12 genome [Internet]. Vol. 31, Nucleic Acids Research. Oxford Academic; 2003 [cited 2021 Mar 23]. p. 6976–85. Available from: http://twod.med.harvard.edu/ExpressDB/

a) User Workflow

Identify Synthetic Gene Sequences
exons in FASTA format

Example Targets:

- FALVAC-1
- PTP4A3
- hPDF

Sequence Codon Optimization
with pre-trained ICOR deep learning model

Construct Optimized Sequence Into Expression Vector

b) Model Creation

Retrieve *E. coli* sequences from NCBI

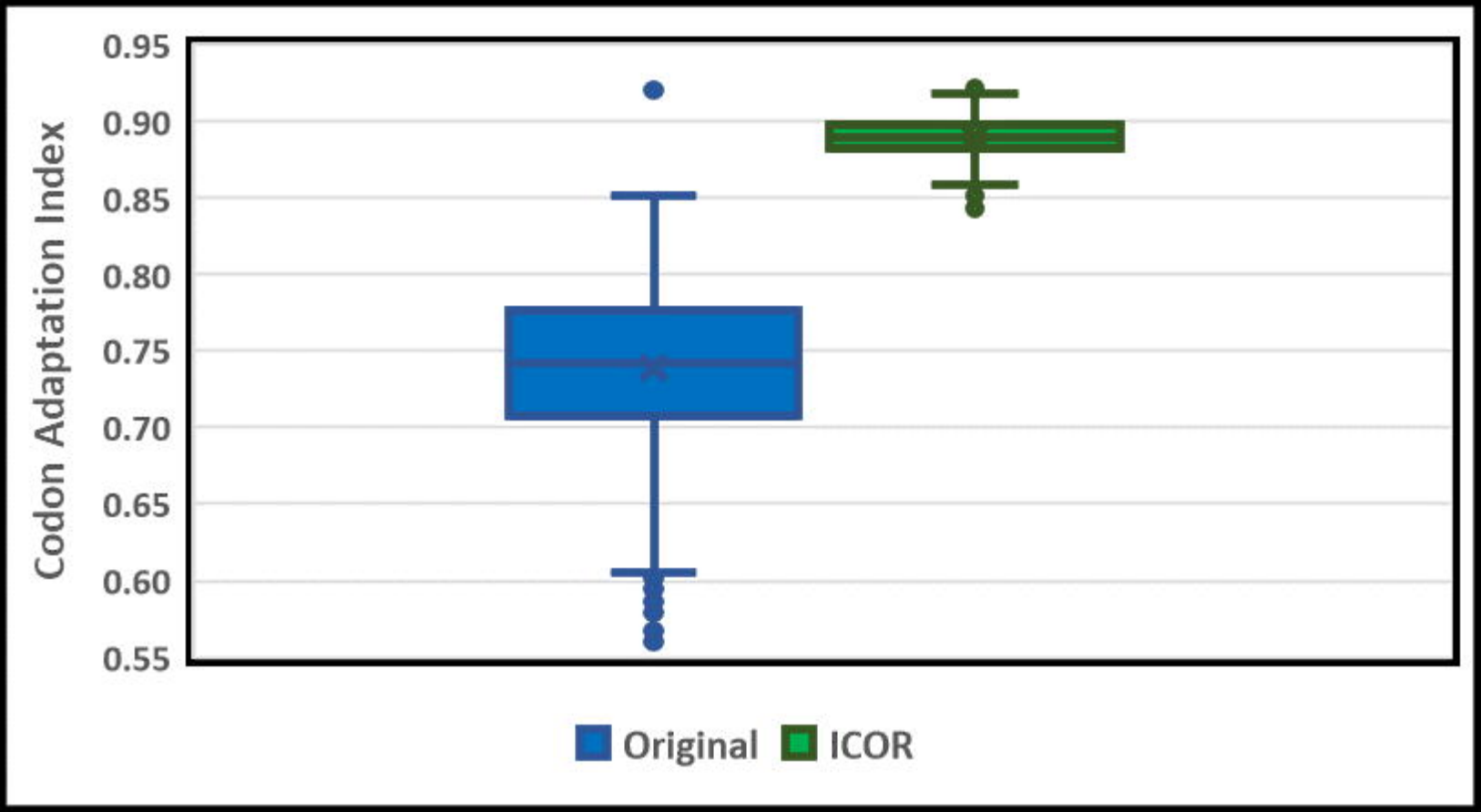
Data Pre-Processing and Pruning

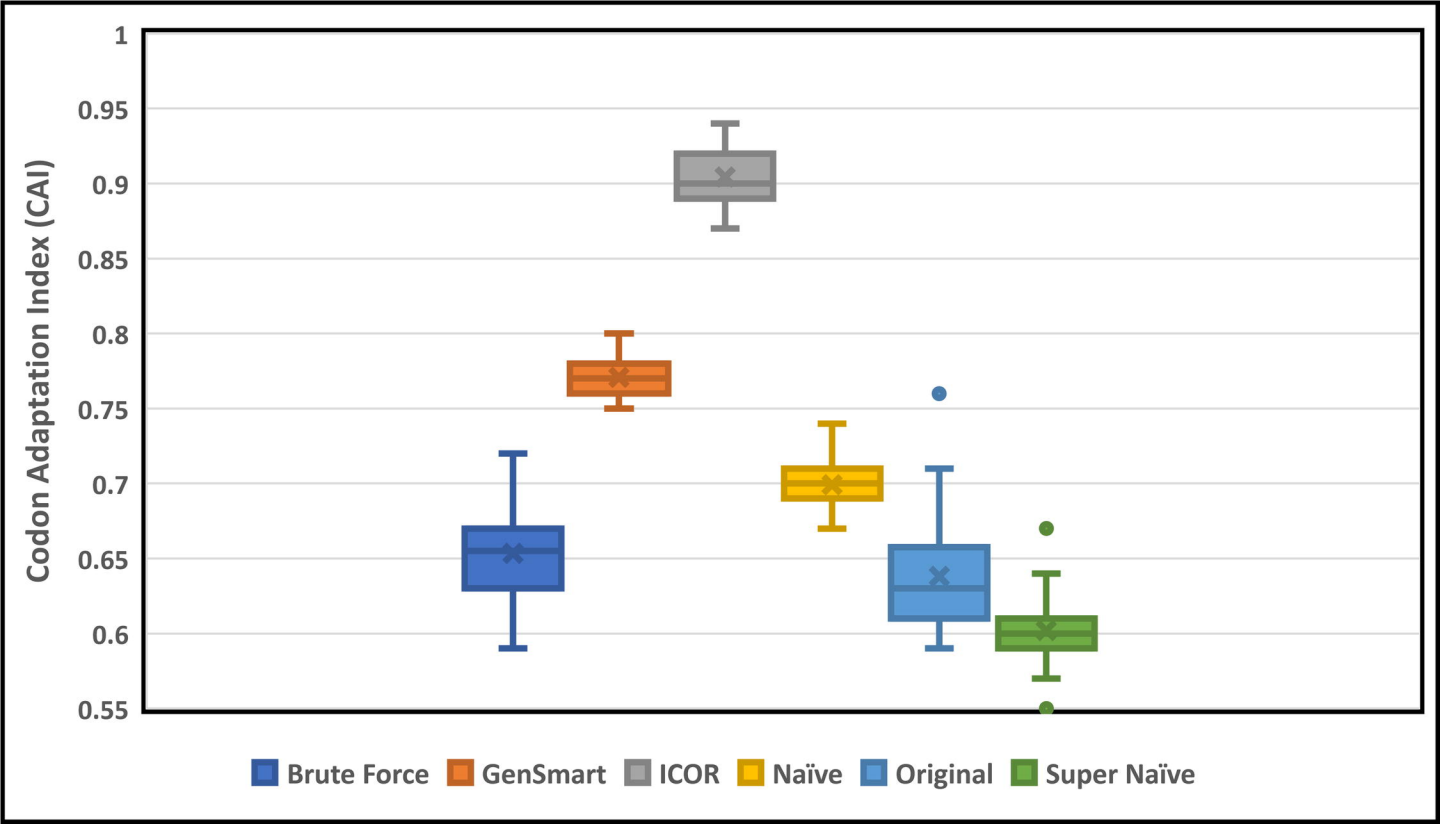
Data Encoding & Feature Creation

Model Training & Hyperparameter Optimization

Model Benchmarking

Packaged ONNX Model


Codon Adaptation Index
0.95
0.90
0.85
0.80
0.75
0.70
0.65
0.60
0.55
Original
ICOR


Codon Adaptation Index (CAI)
1
0.95
0.9
0.85
0.8
0.75
0.7
0.65
0.6
0.55
Brute Force
GenSmart
ICOR
Naïve
Original
Super Naïve