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]

*Corresponding author

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

Background: 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*.

Results: We demonstrate that sequential context achieved via RNN may yield codon selection that is more similar to the host genome. Based on computational metrics that predict protein expression, ICOR theoretically optimizes 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 is measured across five metrics: the Codon Adaptation Index, GC-content, negative repeat elements, negative cis-regulatory elements, and codon frequency distribution.

Conclusions: The results, based on in silico metrics, indicate that ICOR codon optimization is theoretically more effective in enhancing recombinant expression of proteins over other established codon optimization techniques. Our tool is provided as an open-source software package that includes the benchmark set of sequences used in this study.

Keywords: codon optimization, genetic design, synthetic biology, machine learning

Background

Designing synthetic genes for heterologous expression is a keystone of synthetic biology[1]. The expression of recombinant proteins in a heterologous host has applications from manufacturing pharmaceuticals to vaccines. For instance, producing malaria vaccine FALVAC-1[2] involves designing synthetic plasmids, transfection into the *Escherichia coli* (*E. coli*) host factory, growing the cells, and harvesting the resulting protein[3]. *E. coli* is well-established as a host for heterologous expression[4], however, codon bias limits the use of *E. coli* as an expression platform[5]. To increase the efficiency of recombinant expression in *E. coli*, improving codon optimization is an area of particular interest.

Expression levels of synthetic genes in heterologous hosts are dependent on multiple factors including codon usage bias[5–7]. During the process of translation, complimentary tRNAs are used to read codons from an mRNA strand. In *E. coli*, the frequency of a certain codon in its genome is positively correlated with the presence of tRNAs for that codon[5, 8, 9]. Thus, choosing synonymous codons that are more frequently found in a host genome may improve heterologous expression, with 2 to 15-fold increases typically measured in a *E. coli* chassis[7]. Although synonymous codons may code for the same amino acid, they are not redundant[10, 11]. In a study by Gao et al., low expression levels of human immunodeficiency virus genes in mammalian cells were attributed to rare codon usage[12]. Other studies observe that codon bias affects gene expression and even protein folding and solubility[13–15]. Therefore, 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[16]. Codon optimization tools can be used to increase protein expression towards improving the efficiency of manufacturing such products[15]. Codon optimization techniques that are based on biological indexes replace synonymous codons with the most abundant codon found in the host organism's genome[17]. Our review shows that many industry-standard tools employ the aforementioned strategy, causing unintended consequences for the cell, such as an imbalanced tRNA pool[10, 18]. If just one codon of a synonymous set is used throughout an entire synthetic gene, when expressed in the host, metabolic stress and translational error may be imposed[19]. Research has shown that using high-frequency codons only during codon optimization leads to incorrect protein folding and the formation of insoluble proteins[20]. Further, rare codons have been found to play an important role in protein folding[21], thereby raising the interest for understanding subpatterns of codon usage along with surrounding context in which codons are used rather than synonymous codon frequency alone.

Understanding the context by which synonymous codons are used in a gene may be essential to unlock the full evolutionary-instilled potential of a cell factory towards heterologous expression. By utilizing synonymous codons based patterns and sequential context of their prevalence in the host genome, protein expression could be increased while preventing 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[22]. Deep learning systems show promise in bioinformatics, potentially offering improvements over non-machine learning algorithms[23].

Recurrent neural networks (RNNs) are a class of deep neural networks that can grasp temporal data, thus demonstrating utility in applications that require an understanding of sequential information[24]. For example, speech recognition models that utilize long short-term memory (LSTM) architectures[25]—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 if designed to understand underlying patterns of synonymous codon usage to inform subsequent codon prediction. By treating each amino acid as a timestep in a sequence, the RNN evaluates its prediction in the context of surrounding amino acids.

In this study, a deep learning tool, ICOR – Improving Codon Optimization with RNNs – 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 model to improve codon optimization by understanding context. ICOR adopts the Bidirectional Long-Short-Term Memory (BiLSTM) architecture[26] because of its ability to preserve temporal information from both the past and future. In a gene, the BiLSTM would theoretically use surrounding synonymous codons to make a prediction.

40 benchmark genes as shown in Table S1 along with 1,481 *E. coli* genes were used for testing and data analysis, allowing us to evaluate the model in two ways: its ability to optimize genes used in past recombinant expression studies, and its ability to replicate but still optimize sequences from *E. coli*. The resultant optimized sequences from the ICOR model were compared to six approaches (original, uniform random choice (URC), background frequency choice (BFC), extended random choice (ERC), highest frequency choice (HFC), GenScript's GenSmart[27]) as outlined in S1 Appendix. GenSmart is accepted as the industry-standard benchmark due to recognition in past studies[28], ease-of-use, and accessibility. To gauge performance of the codon optimization approaches, the Codon Adaptation Index (CAI), GC-content, codon frequency distribution (CFD), negative repeat elements, negative cis-regulatory elements, and algorithm run-time are measured on the 40 benchmark and 1,481 *E. coli* genes. These in silico metrics demonstrate the effectiveness of this approach as an algorithm. The ICOR tool is open-source and can be accessed at: https://doi.org/10.5281/zenodo.5529209.

Implementation

Model Training Dataset

We use the National Center for Biotechnology Information's (NCBI) GenBank database[29] which includes 6,877,000 genes reported for many *E. coli* strains. *E. coli* genes that were shorter than 90 amino acids in length were removed due to their hypothetical or specialized nature. Then, CD-HIT-EST[30] was utilized to cluster and remove similar nucleotide sequences that had sequence identities of over 90%. With a high filter, we still maintain some similar genes and the CD-HIT-EST tool creates small clusters. After removal of such redundant genes, the remaining 42,266 sequences were sorted in descending order based on their CAI as depicted in S2 Appendix. Of these, 7,406 sequences with the highest CAI were selected to serve as the model dataset. Approximately 70% of the dataset was used for training (5,184 sequences), 10% for validation (741 sequences), and 20% for testing (1,481 sequences).

Synthetic Plasmid Benchmarks

40 DNA sequences were established as a benchmark set, extracted from both studies conducted on codon optimization and gene expression evaluation 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. In addition, we experimented with encodings based on a Non-Linear Fisher Transform (NLFT) technique that has coded 18 features per amino acid[32].

Model Building

Predicting an optimal synonymous codon with sequential information – the sequence of codons that surround the prediction – 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[26] which predicts synonymous codons given the input amino acid sequence. The model hyperparameters as shown in Table S3 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

The ICOR model's architecture consists of a 12-layer recurrent neural network as visualized in Fig. 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[31] on the Tesla V100 graphics card. The model was trained on 50 epochs and took 138 minutes to complete training.

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 specifics about development are detailed in S1 Supporting Information.

With an input of sequences to be optimized, a user receives codon sequences optimized for *E. coli* expression using the ICOR runtime scripts. The runtime scripts utilize the ONNX runtime to inference the trained model.

Statistical Analysis

The CAI is calculated using the formulae described in S1 Supporting Information. GenScript's rare codon analysis tool[33] is utilized to calculate GC-content, CFD, negative repeat elements, and negative cis-regulatory elements. 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[34].

Results

We use multiple previously established metrics such as the Codon Adaptation Index[35], GC-content, CFD, number of negative repeat elements, and negative cis-regulatory elements to quantify the performance of our tool. Formal definitions of these metrics are given in S1 Supporting Information.

Codon Adaptation Index

As noted in previous studies, CAI is highly correlated with real-world expression[36]. 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

In order to properly contextualize the performance of the developed model, six algorithms from S1 Appendix were used to optimize the benchmark set of 40 genes. These 40 genes came from a variety of origin organisms and had a mean CAI of 0.638 with a standard deviation of 0.0386. ICOR optimization yielded a mean CAI of 0.904 with a standard deviation of about 0.016, signifying a ~41.692% increase in CAI. The URC approach had a mean CAI of 0.602 and standard deviation of about 0.022. ICOR offered a ~50.21% increase in CAI compared to this approach. Finally, the BFC 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 BFC 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

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 estimated 236%[36].

Secondary Endpoints

The secondary endpoints were quantified for each of the six codon optimization approaches on the benchmarking dataset sequences (n=40).

Ideal GC-content for recombinant genes is known to be between 30% and 70%; peaks outside of this range adversely affect transcriptional and translational efficiency[33]. ICOR, along with the other optimization techniques were all found to optimize genes within this range. The mean GC-content for each optimization technique is depicted in Fig. 4.

Fig. 4

Genes that employ low frequency (<30% usage in the host genome) can cause a disengagement of translational machinery and reduce the efficiency of translation[33]. This rare codon frequency distribution is measured as a percentage and minimizing this value is ideal. ICOR offered a significant improvement in CFD, outperforming all the other optimization techniques tested. ICOR reduced CFD by 93.55% and 97.69% compared to the GenSmart and original sequences respectively. The improvement over GenSmart and original sequences were both statistically significant with p-values less than 0.0001. The mean CFD for each optimization technique is depicted in Fig. 5.

Fig. 5

There was a difference (p=0.726) that was not found to be significant between ICOR and GenSmart in the number of negative cis-regulatory elements using a two-sided Mann-Whitney U Test for non-parametric distributions. When computing the mean change in negative cis-regulatory elements between the optimization tool and the original sequence, there is also a statistically insignificant difference between ICOR and GenSmart. This suggests that ICOR maintains equally low negative cis-regulatory elements as GenSmart while achieving a higher CAI. The mean number of negative cis-regulatory elements is depicted in Fig. 6.

Fig. 6

There was a trending difference (p=0.1826) between ICOR and GenSmart in the number of negative repeat elements in the optimized sequences using a two-sided 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. The mean number of negative repeat elements is depicted in Fig. 7.

Fig. 7

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 URC approach are displayed in Table 3.

	Average Time (n. seq=40) (n. trials=3)	**Time Per Seq.**	**Time Per Codon**	**Time per NT**
ERC	**275,587%** (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**
BFC	**283%** (771ms)	**19.3ms**	**1.37ms**	**0.456ms**

	(0.761s, 0.764s, 0.787s)			
HFC	**84%** (229ms) (0.233s, 0.226s, 0.228s)	**5.7ms**	**0.40ms**	**0.136ms**
URC	**100%** (272ms) (0.271s, 0.274s, 0.272s)	**6.8ms**	**0.48ms**	**0.161ms**

Table 3. ICOR runtime comparable to the BFC, HFC, and URC approaches and significantly faster than ERC 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 URC approach because it represents the most naïve codon selection.

GenSmart optimization was not comparable due to its production environment with a queue of jobs.

Gene Mutations

The deep learning model in this study does not have strict rules in place regarding codon usage: it is not explicitly given a “codon to amino acid dictionary.” At least one point mutation would arise if a codon prediction is replaced with a non-synonymous choice (e.g., CAA to GAA). On the test dataset of 1,481 genes, our model yielded a 0.00% mutational rate and did not predict non-synonymous codons. Thus, it was found that gene mutations are not present in the ICOR codon optimization technique.

During our testing, it was found that encoding techniques made a significant difference in learning these initial codons to amino acid pairings. The One-Hot Encoding technique offered approximately a 10% improvement in matching prediction of the host codon over NLFT during model training.

Conclusions

In this paper, we introduce ICOR, a codon optimization tool that uses recurrent neural networks towards improving heterologous expression for synthetic genes. We find that deep learning is a particularly effective codon optimization method because it learns codon usage bias in tandem with a codon’s surrounding context, thus allowing it to make predictions using the patterns and subsequences in which synonymous codons are used in genes. While previous research ranges from selecting high-frequency codons, to eliminating secondary structures, to machine learning models and convolutional neural network architectures, we use the RNN architecture which has the ability to take sequential information into account. The RNN can thereby use underlying patterns in genes to inform codon selection that may be more similar to that of the host genome.

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 across various strains contain similar genes. We used the CD-HIT-EST server to overcome this issue because a model trained on redundant data would yield codon selection that is biased to common *E. coli* genes only. Further, this helped reduce the necessary compute resources required for deep learning. The ICOR model encodes gene sequences using Natural Language Processing techniques, demonstrating their effectiveness in this task.

The statistical data analysis uses the CAI, GC-content, CFD, negative repeat elements, and negative cis-regulatory elements as metrics to compare ICOR to other solutions. Although ICOR demonstrates improvements in these metrics, signifying increased protein expression, they are not the only comprehensive statistics to predict gene expression. Recent research has shown that creating models for measuring translation dynamics is possible[37], and such research can be applied to the results of our study. Modeling elongation rate, tRNA adaptation index, and other metrics may provide further valuable insight into the results of our tool.

In this research, six codon optimization techniques were evaluated. Of these, the ERC 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 acid 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, our review suggests that optimization may instead require a genome-wide understanding of codon usage to yield optimal expression. The HFC algorithm illustrates the implications of a "CAI = 1.0" optimization strategy which the results point towards increased negative cis-regulatory elements and negative repeat elements. In addition, our review suggests that this method may result in plasmid toxicity.

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 *E. coli* recombinantly-produced products. Currently, the ICOR tool can be accessed through an open-source software package at https://doi.org/10.5281/zenodo.5529209, however, we would like to build an API to improve accessibility in the future.

Although our model is based on *E. coli* genomes, it may be possible 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, experimental results for our method are not included. This is relevant. As a contribution to bioinformatics and machine learning, biological results would only be useful to demonstrate our ability to synthesize specific DNA sequences, which is outside the scope of this paper. The efficacy of these sequences would only feedback into our machine learning workflow and not fundamentally change the process as outlined.

Availability and requirements

Project name: ICOR: Improving Codon Optimization with Recurrent neural networks

Project home page: https://github.com/Lattice-Automation/icor-codon-optimization

Operating system(s): Platform independent

Programming language: Python

Other requirements: Python 3.9.4 or newer

License: MIT

Any restrictions to use by non-academics: license needed

Declarations

Abbreviations

E. coli	*Escherichia coli*
RNNs	Recurrent neural networks
LSTM	Long short-term memory
BiLSTM	Bidirectional LSTM
CAI	Codon adaptation index
CFD	Codon frequency distribution
NCBI	National Center for Biotechnology Information
NLFT	Nonlinear Fisher Transformation
ERC	Extended Random Choice
BFC	Background Frequency Choice
URC	Uniform Random Choice

Ethics approval and consent to participate

Not Applicable

Consent for publication

Not Applicable

Availability of data and material

The training dataset was derived from the National Center for Biotechnology Information's (NCBI) GenBank database [https://www.ncbi.nlm.nih.gov/assembly/?term=escherichia+coli]

The benchmark sequence datasets used to compare codon optimization approaches are available in the ICOR repository [doi:10.5281/zenodo.5529209] and see Reference [29]. Please see S1_Table for public references used to derive the training dataset.

Competing interests

Aditya Jain has declared that no competing interests exist.

Rishab Jain has declared that no competing interests exist.

Douglas Densmore has read the journal's policy and has declared the following competing interests: commercial interests at Lattice Automation and BioSens8, Professorship at Boston University, and co-founder of Asimov, Inc.

Kevin LeShane has read the journal's policy and has declared the following competing interests: I have financial competing interests at Lattice Automation and Asimov Inc.

Elizabeth Mauro has declared that no competing interests exist.

Funding

Not Applicable.

Authors' contributions

RJ, AJ, and DD were involved in determining the datasets and methods of comparison for codon optimization tools. RJ, AJ, DD, KL were involved in the writing of the manuscript and in the interpretation of the results. RJ, KL, and EM were involved in administration. All authors edited, revised, read and approved the final manuscript.

Acknowledgements

Not Applicable

References

1. Endy D. Foundations for engineering biology. Nature 2005 438:7067. 2005;438:449–53.

2. Zhou Z, Schnake P, Xiao L, Lal AA. Enhanced expression of a recombinant malaria candidate vaccine in Escherichia coli by codon optimization. Protein Expression and Purification. 2004;34:87–94.

3. Nascimento IP, Leite LCC. Recombinant vaccines and the development of new vaccine strategies. Brazilian Journal of Medical and Biological Research. 2012;45:1102–11.

4. Mitchell AM, Gogulancea V, Smith W, Wipat A, Ofițeru ID. Recombinant Protein Production with Escherichia coli in Glucose and Glycerol Limited Chemostats. Applied Microbiology. 2021;1:239–54.

5. Lipinszki Z, Vernyik V, Farago N, Sari T, Puskas LG, Blattner FR, et al. Enhancing the Translational Capacity of E. coli by Resolving the Codon Bias. ACS Synthetic Biology. 2018;7:2656–64.

6. 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. 2016;113:E6117–25.

7. Gustafsson C, Govindarajan S, Minshull J. Codon bias and heterologous protein expression. Trends in Biotechnology. 2004;22:346–53.

8. Brule CE, Grayhack EJ. Synonymous Codons: Choose Wisely for Expression. Trends in Genetics. 2017;33:283–97.

9. 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. Journal of Molecular Biology. 1981;151:389–409.

10. Villalobos A, Ness JE, Gustafsson C, Minshull J, Govindarajan S. Gene Designer: A synthetic biology tool for constructuring artificial DNA segments. BMC Bioinformatics. 2006;7:285.

11. Plotkin JB, Kudla G. Synonymous but not the same: The causes and consequences of codon bias. Nature Reviews Genetics. 2011;12:32–42.

12. Gao W, Rzewski A, Sun H, Robbins PD, Gambotto A. UpGene: Application of a web-based dna codon optimization algorithm. Biotechnology Progress. 2004;20:443–8.

13. Kudla G, Murray AW, Tollervey D, Plotkin JB. Coding-sequence determinants of expression in escherichia coli. Science (1979). 2009;324:255–8.

14. Rosano GL, Ceccarelli EA. Rare codon content affects the solubility of recombinant proteins in a codon bias-adjusted Escherichia coli strain. Microbial Cell Factories. 2009;8:1–9.

15. Mauro VP, Chappell SA. A critical analysis of codon optimization in human therapeutics. Trends in Molecular Medicine. 2014;20:604–13.

16. Sanchez-Garcia L, Martín L, Mangues R, Ferrer-Miralles N, Vázquez E, Villaverde A. Recombinant pharmaceuticals from microbial cells: A 2015 update. Microbial Cell Factories. 2016;15:33.

17. Tian J, Li Q, Chu X, Wu N. Presyncodon, a web server for gene design with the evolutionary information of the expression hosts. International Journal of Molecular Sciences. 2018;19.

18. Puigbò P, Guzmá E, Romeu A, Garcia-Vallvé S. OPTIMIZER: a web server for optimizing the codon usage of DNA sequences. Nucleic Acids Research. 2007;35.

19. Angov E. Codon usage: Nature's roadmap to expression and folding of proteins. Biotechnology Journal. 2011;6:650.

20. Hurley JM, Dunlap JC. A fable of too much too fast. Nature 2013 495:7439. 2013;495:57–8.

21. 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. PLOS Computational Biology. 2017;13:e1005531.

22. Miotto R, Wang F, Wang S, Jiang X, Dudley JT. Deep learning for healthcare: Review, opportunities and challenges. Briefings in Bioinformatics. 2017;19:1236–46.

23. Tang B, Pan Z, Yin K, Khateeb A. Recent Advances of Deep Learning in Bioinformatics and Computational Biology. Frontiers in Genetics. 2019;0 MAR:214.

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

25. Hochreiter S, Schmidhuber J. Long Short-Term Memory. Neural Computation. 1997;9:1735–80.

26. Schuster M, Paliwal KK. Bidirectional recurrent neural networks. IEEE Transactions on Signal Processing. 1997;45:2673–81.

27. GenSmart^{TM} Codon Optimization Tool-GenScript. https://www.genscript.com/gensmart-free-gene-codon-optimization.html. Accessed 2 Oct 2021.

28. 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. 2018;36:843.

29. National Center for Biotechnology Information. Genome Escherichia coli. Bethesda. 2021.

30. Huang Y, Niu B, Gao Y, Fu L, Li W. CD-HIT Suite: a web server for clustering and comparing biological sequences. Bioinformatics. 2010;26:680–2.

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

32. Nanni L, Lumini A. A new encoding technique for peptide classification. Expert Systems with Applications. 2011;38:3185–91.

33. Rare Codon Analysis Tool. https://www.genscript.com/tools/rare-codon-analysis. Accessed 2 Oct 2021.

34. Kane JF. Effects of rare codon clusters on high-level expression of heterologous proteins in Escherichia coli. Current Opinion in Biotechnology. 1995;6:494–500.

35. Sharp PM, Li WH. The codon adaptation index-a measure of directional synonymous codon usage bias, and its potential applications. Nucleic Acids Research. 1987;15:1281–95.

36. 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. Nucleic Acids Research. 2003;31:6976–85.

37. Trösemeier JH, Rudorf S, Loessner H, Hofner B, Reuter A, Schulenborg T, et al. Optimizing the dynamics of protein expression. Scientific Reports 2019 9:1. 2019;9:1–15.

Figure Legends

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.

Fig. 2. ICOR significantly improves Codon Adaptation Index (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.

Fig. 3. ICOR significantly improves Codon Adaptation Index (CAI) when compared to the original, BFC, URC, ERC, and GenSmart techniques. Box and Whisker Plot (n=40) comparing CAI with legend indicating the color for each optimization method.

Fig. 4. ICOR and other codon optimization techniques all maintain within optimal GC-content range. Box and whisker plot (n=40) comparing GC-content with legend indicative of each optimization method.

Fig. 5. ICOR codon optimization significantly improves mean Codon Frequency Distribution (CFD). Box and whisker plot (n=40) comparing CFD with legend indicating each optimization method.

Fig. 6. ICOR codon optimization technique yields statistically insignificant change in negative cis-regulatory elements. Box and whisker plot (n=40) comparing negative cis-regulatory elements with legend indicative of each optimization method.

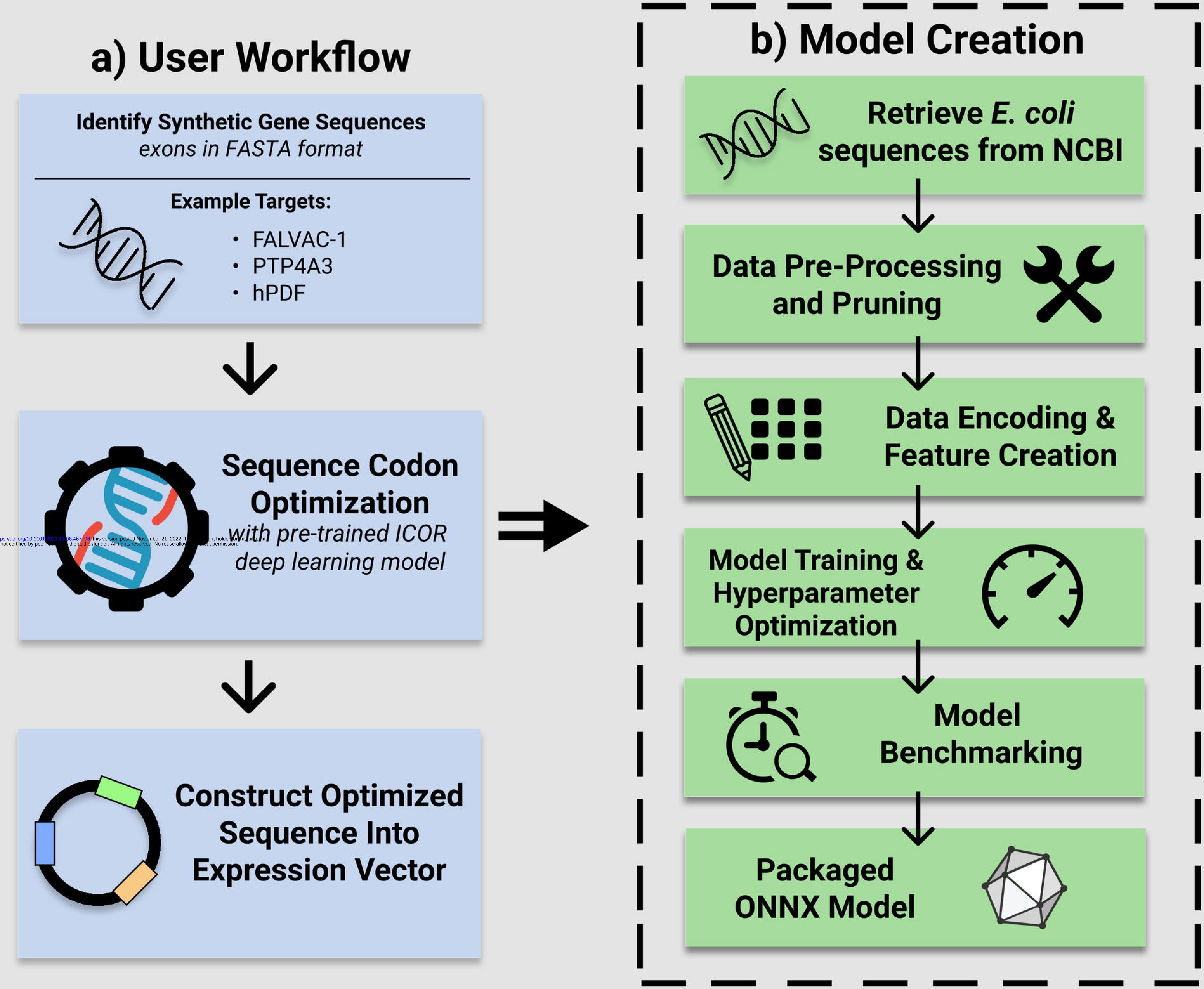
Fig. 7. Trending difference between ICOR optimization in number of negative repeat elements. Box and whisker plot (n=40) comparing negative repeat elements with legend indicative of each optimization method.

Additional Files

1. File name: S1_Fig.pdf
    a. Title: ICOR deep learning model architecture
    b. Description: An overview of the 12-layer recurrent neural network model trained in this study.
2. File name: S1_Appendix.docx
    a. Title: Codon optimization methods overview
    b. Description: An overview of the four codon optimization methods built and defined in this study.
3. File name: S1_SupportingInformation.docx
    a. Title: Supporting details for testing methods and computational metrics
    b. Description: Specifications for the system used in the benchmarking of the software tool along with supporting formulae used in the validation of the software tool performance.
4. File name: S1_Table.docx
    a. Title: Description of 40 genes used to benchmark codon optimization methods
    b. Description: Average amino acids/codons in sequence: 562.55. Average length/nucleotides of sequence: 1687.65. Genes are sorted by the number of base pairs from smallest to greatest.
5. File name: S2_Appendix.docx
    a. Title: Training dataset specifications

   b. Description: A histogram displaying the distribution of the codon adaptation index of genes utilized in the ICOR training dataset.

6. File name: S2_Table.docx
   a. Title: Amino acid integer lookup table
   b. Description: A list of the encoding used for converting amino acids to numerical, integer values.
7. File name: S3_Table.docx
   a. Title: Hyperparameter variables are used to finetune the results of the model
   b. Description: (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.



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

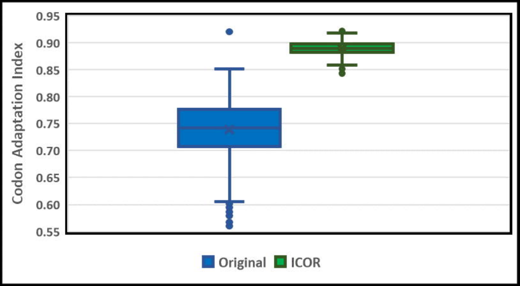
Data Pre-Processing and Pruning

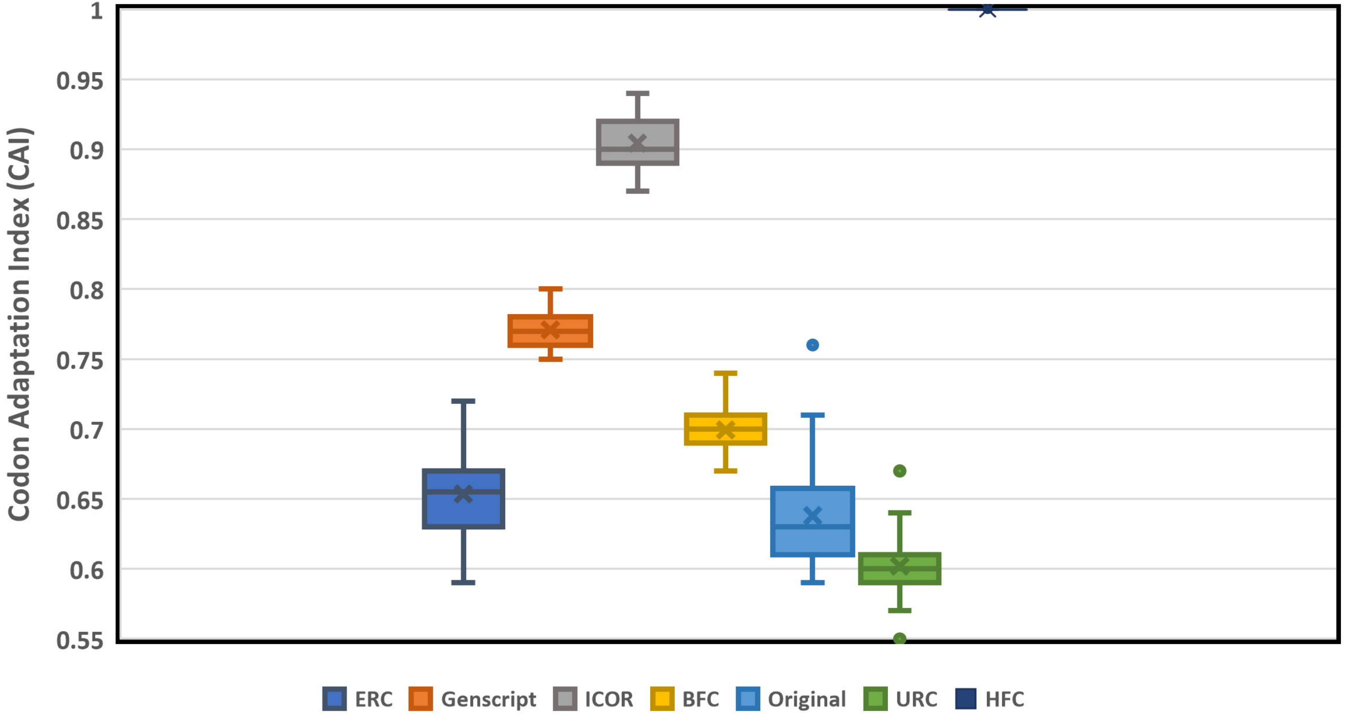
Data Encoding & Feature Creation

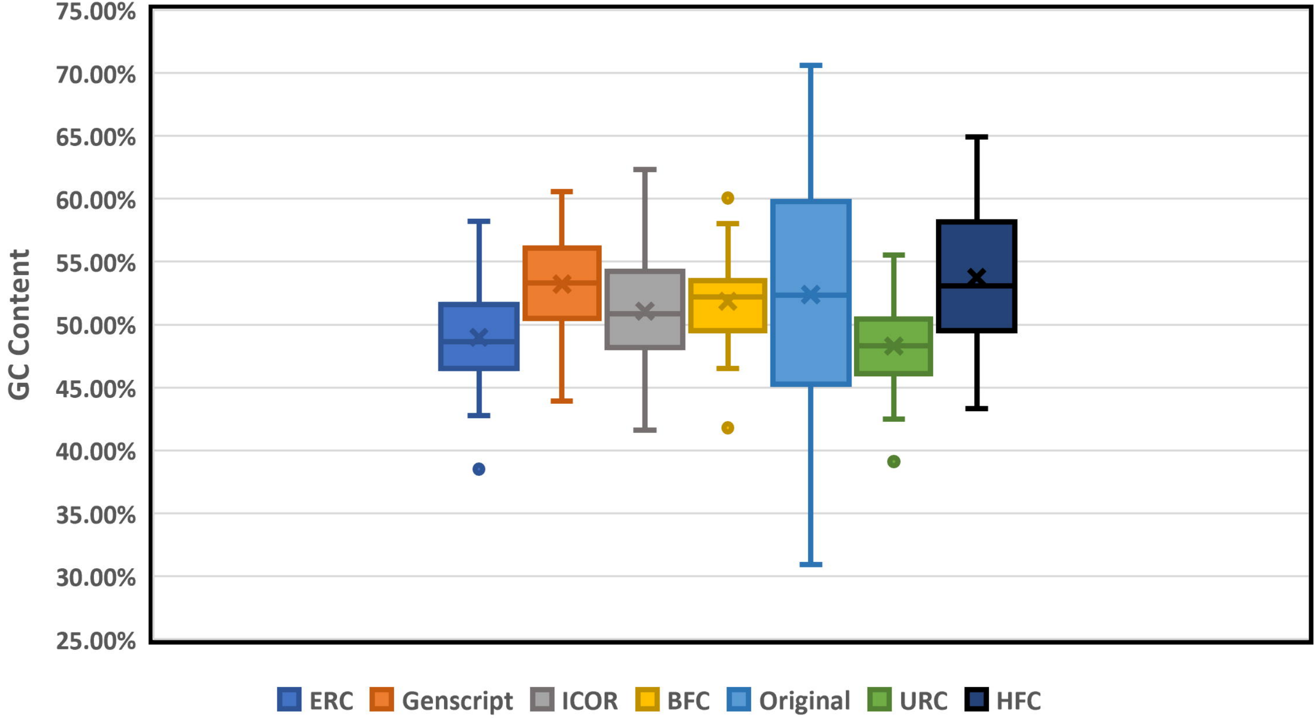
Model Training & Hyperparameter Optimization

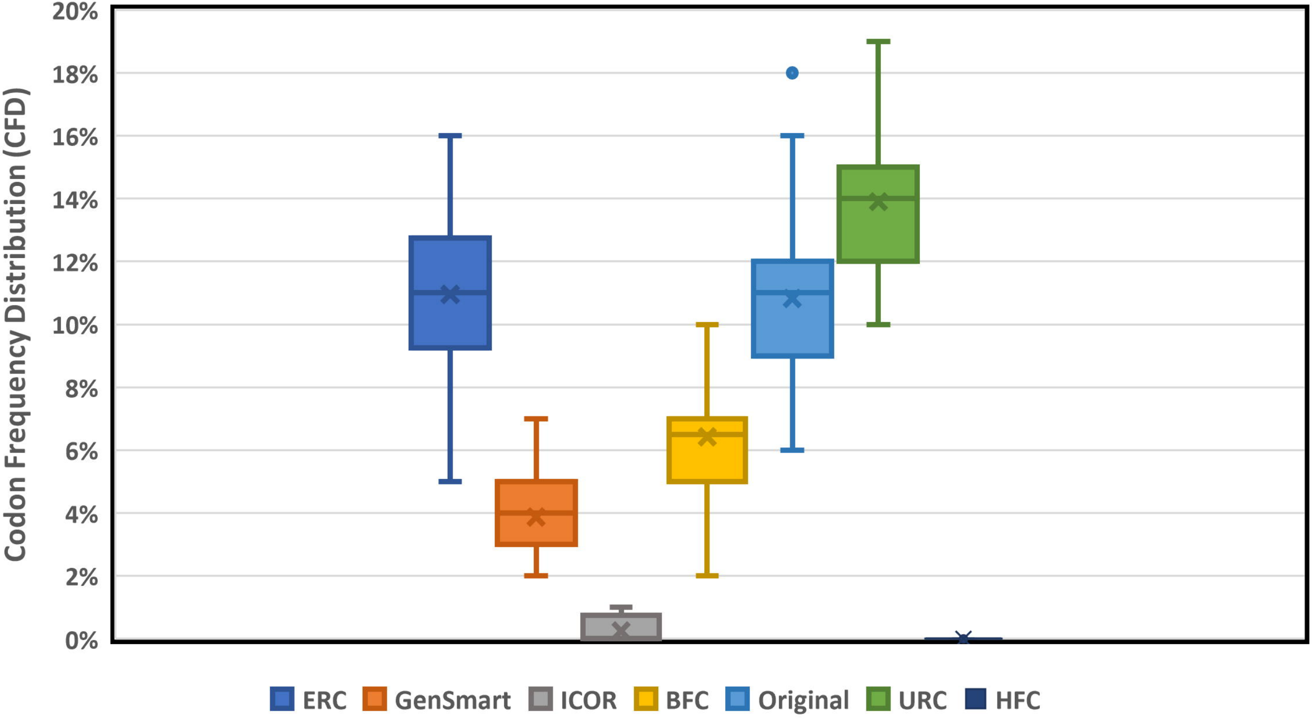
Model Benchmarking

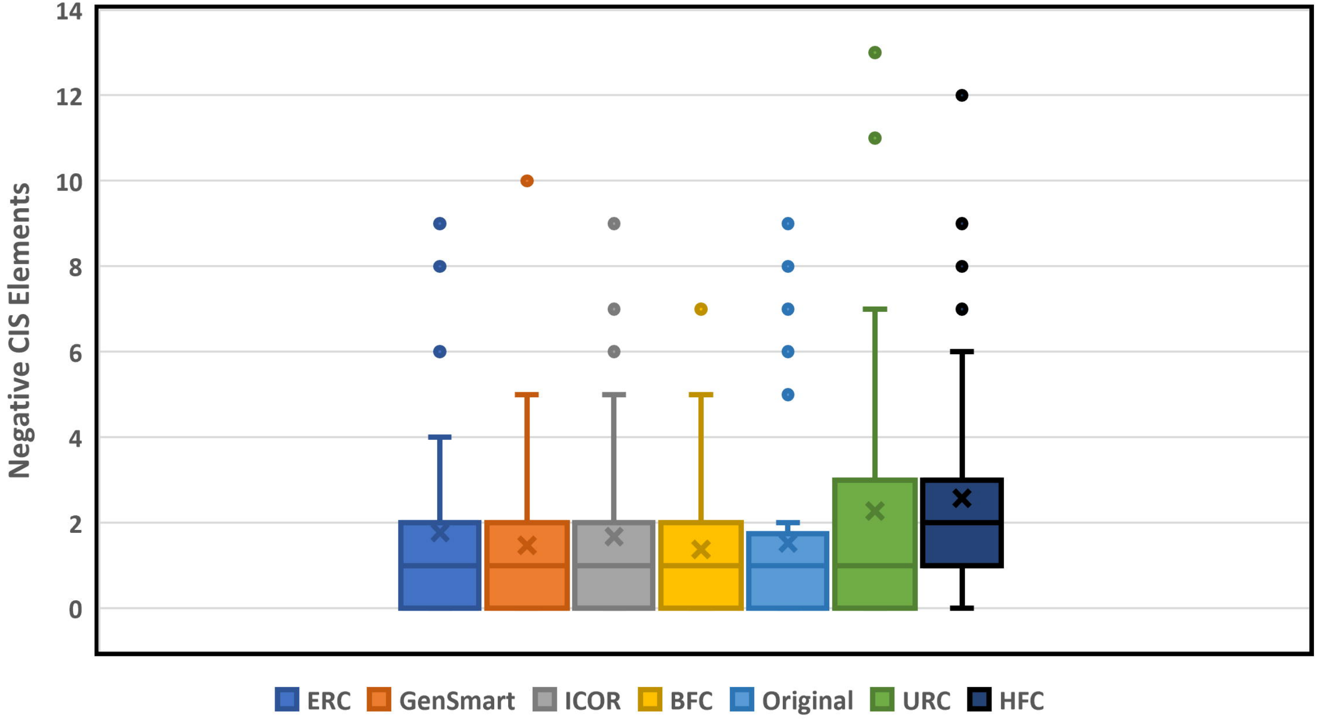
Packaged ONNX Model

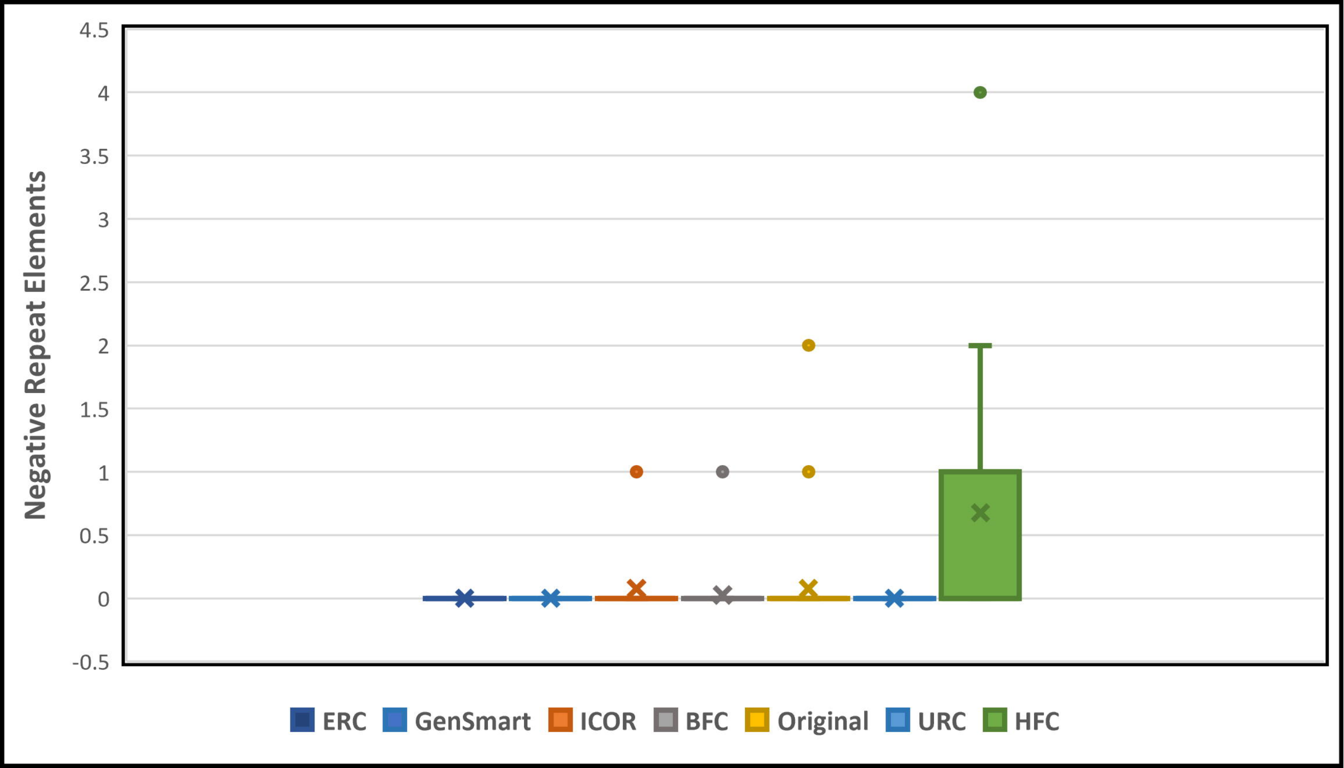

0.95
0.90
0.85
0.80
0.75
0.70
0.65
0.60
0.55
Codon Adaptation Index
Original
ICOR


Codon Adaptation Index (CAI)
1
0.95
0.9
0.85
0.8
0.75
0.7
0.65
0.6
0.55
ERC
Genscript
ICOR
BFC
Original
URC
HFC


75.00%
70.00%
65.00%
60.00%
55.00%
50.00%
45.00%
40.00%
35.00%
30.00%
25.00%
GC Content
ERC
Genscript
ICOR
BFC
Original
URC
HFC


Codon Frequency Distribution (CFD)
20%
18%
16%
14%
12%
10%
8%
6%
4%
2%
0%
ERC
GenSmart
ICOR
BFC
Original
URC
HFC


Negative CIS Elements
14
12
10
8
6
4
2
0
ERC
GenSmart
ICOR
BFC
Original
URC
HFC


Negative Repeat Elements
4.5
4
3.5
3
2.5
2
1.5
1
0.5
0
-0.5
ERC
GenSmart
ICOR
BFC
Original
URC
HFC