CodonT5: A Multi-task Codon Language Model to Perform Generative Codon Optimization

1st Ashley Babjac
Department of Computer Science
University of Tennessee
Knoxville, USA
ababjac@vols.utk.edu

2nd Scott J. Emrich
Department of Computer Science
University of Tennessee
Knoxville, USA
semrich@utk.edu

2024 IEEE International Conference on Bioinformatics and Biomedicine (BIBM) | 979-8-3503-8622-6/24/$31.00 ©2024 IEEE | DOI: 10.1109/BIBM62325.2024.10822720

***Abstract*—Codon language modeling is an emerging area of research interest. Prior efforts in codon optimization have mostly focused on the effect codon preferences have on protein translation speed either at single codons (e.g., selection preference measures) or entire gene sequences (e.g., Codon Adaptation Index and related estimates of gene expression). Model parameters are almost always trained based on a specific organism/task. Here we explore a first ever look at the impact of multi-task model pre-training with the end goal of performing codon "translations" between species. Significantly, our model, CodonT5, is able to accurately predict across multiple tasks and generate comparable codon optimizations to previously established methods using *only* the reference sequences without being necessarily dependent on any pre-computed metrics of codon preference. This is important as it paves the way for more diverse sequence-to-sequence modeling that will be necessary in many applications involving replicating a reference protein in a host (e.g., mRNA vaccines).**

***Index Terms*—Transformers, Protein Language Modeling, Codon Usage, Translation, Heterologous Expression**

I. Introduction

Heterologous expression is the term used to describe synthesizing proteins from one species within a different host species [18]. Disregarding codon usage preferences in the host can directly result in poor production due to changes in translation rate being induced by synonymous codon changes [24].It is well known that certain proteins do not fold as well when codon preferences are "shuffled" [24], which implies that co-translational folding is sometimes required because the only change is in the localized (read: within-sequence) codon usage; hence, the sequence of amino acids translated in these experiments remains the same.

Previous bioinformatics methods have attempted to provide alternative codon choices for potentially improved heterologous expression. This alternative sequence of codons is often referred to as a "codon optimization," and a naive method is simply to swap all codons for their most "common" synonym based on the estimated biological preference(s) of the host organism [8, 16, 23]. This is often done using Codon Adaptation Index (CAI) frequencies based on known highly expressed genes in the host [20], but can also be done using other metrics [22, 18]. An alternative method has been called "harmonization", where the goal is to preserve more localized codon preferences. For example, the CHARMING algorithm [25] uses windows of codons (usually around nine) such that the resulting codon usage in a given window in a host organism more closely matches the codon usage of its source organism. While these methods have been shown to work well *in vitro* [18], they can be dependent on the seeding (initial sequence) as well as what measure of codon preference is used. This is further complicated because there are an exponential number of unique codon sequences that encode a specific protein making it infeasible to test all of them experimentally. We aim to expedite this process using recent advances in generative artificial intelligence; we introduce a new paradigm for codon optimization—sequence-to-sequence "translation"—where we translate codons in specific proteins from species A (e.g., human) to species B (e.g., *E. coli*) using a transformer-based model that incorporates diverse evolutionary preferences from multiple tasks.

A. Related Work

Outeiral and Deane have recently shown that codon language models can outperform protein language models by a large margin on a wide variety of benchmarking tasks even when they have substantially fewer parameters [15]. Li et al. used BERT (encoder-only transformer) to perform sequence optimization for mRNA vaccine development as well as predict several downstream tasks [11] and very recently Sidi et al. employed mBART (multi-lingual encoder-decoder architecture with a denoising autoencoder) to both predict codon sequences from the amino acid sequence as well as generate sequences in a given organism based on an orthologous sequence [21]. Our aim is a bit different; we would like to utilize aligned sequences of different species to better target applications such as heterologous expression, especially since regions important for co-translational folding are conserved in such alignments (e.g., [4]). Jain et al. [10] attempt something similar to a deep learning codon optimization but they only train on ~7,000 sequences of *E. coli* and use BiLSTMs which are becoming obsolete in comparison to transformers [6].

B. Our Contributions

Since our last paper using BERT [2], we have been building an improved codon language model inspired by the Unified Text-to-Text Transfer Transformer (T5), which has shown that prompted, multi-task, pre-training created a more

979-8-3503-8622-6/24/$31.00 ©2024 IEEE

verbose shared embedding space that improved performance across all tasks [17]. We contribute CodonT5: the first ever codon-based multi-task model to perform prediction of four important protein language tasks: expression, disorder, and species prediction, as well as the newly defined "translation". Because each of the "core" tasks have pre-established deep learning-inspired models that are closely related ([2], [14], [3]), there exists a clear opportunity to assess if a multi-task pre-training generates better results. We initially pre-train our model using six different species across all tasks (see Section II-A and Figure 1) before fine-tuning specifically towards "translations" (potential codon optimizations). Notably, our codon optimization approach does not rely on any one codon preference metric because of its generative nature, and instead learns context concerning evolutionary codon bias from the intentionally chosen multi-task pre-training. We show that (i) our best multi-task model outperforms similar benchmarks on the four core tasks, and (ii) our translation models can create "zero-shot" translations for the protein chloramphenicol acetyltransferase (CAT) [24] with comparable codon usage preferences to both the wild type and experimentally tested mutants (*in vivo*) [18].

II. Materials and Methods

A. Data

We collected expression measurements from six model organisms deposited in Genbank: *Saccharomyces cerevisiae* (Baker's yeast), *Escherichia coli*, *Caenorhabditis elegans* (Roundworm), *Drosophila melanogaster* (Common fruit fly), *Arabidopsis thaliana* (Thale cress), and *Mus musculus* (House mouse). For each species, the median expression values from baseline control experiments is used as documented previously in [2]. We include an additional dataset of yeast disorder scores that we used previously in [1].

To perform "translations", we pre-process the data to determine pairs of alignments between the six available species. We utilize a multiple sequence alignment (MSA) algorithm as follows: (i) reciprocal best BLAST hits between all pairs of organisms are computed using translated BLAST (tBLASTx), which converts input nucleotide sequences into all six frames for comparison; (ii) next, a custom Perl script generates mutual best hits by processing BLAST output generated using a minimum e-value of 1e-8; and (iii) another custom Perl script takes the original sequences along with the n nucleotide sequence pairs from the prior step, generates n unique FASTA files with two sequences (one from each organism), and then aligns these pairs using the evolutionary (and codon)-aware alignment tool PRANK [13] using default parameters (v.170427).

After getting the paired alignments, we remove all codons that align to gaps or mismatches, which guarantees the same amino acid sequence corresponds to both codon sequences. We additionally pre-process all sequences such that they start from the start codon (ATG), stop at a stop codon, are the same length as the unaligned sequence and are a multiple of three. This ensures that the resulting sequence will create the same functional protein. These paired sequences will be utilized for both pre-training and fine-tuning of the "translation" task, where we will refer to the input sequence from the pair as the "source" (aka the reference) and the label sequence from the pair as the "target" (aka the host).

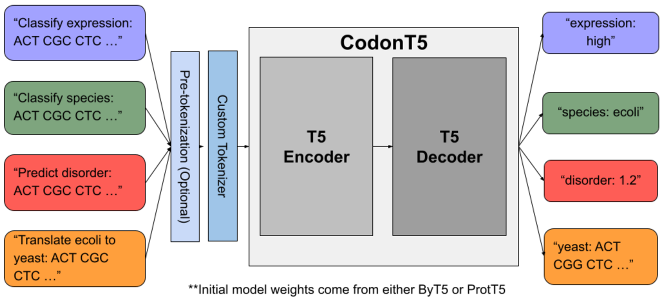
"Classify expression: ACT CGC CTC ..."
"Classify species: ACT CGC CTC ..."
"Predict disorder: ACT CGC CTC ..."
"Translate ecoli to yeast: ACT CGC CTC ..."
Pre-tokenization (Optional)
Custom Tokenizer
CodonT5
T5 Encoder
T5 Decoder
"expression: high"
"species: ecoli"
"disorder: 1.2"
"yeast: ACT CGG CTC ..."
**Initial model weights come from either ByT5 or ProtT5

Fig. 1. A diagram of our codon-based multi-task sequence-to-sequence framework. Our multi-task model is trained on four distinct tasks following the original T5 paradigm [17]: (i) expression classification (blue; referred to as "sentiment", see Babjac et al. [2]), (ii) species classification (green), (iii) disorder prediction (red), and (iv) sequence-to-sequence translation (orange). We test two different pre-training checkpoints for initializing CodonT5 with different theoretical advantages (see Section II-B).

Using these same sequences, we also define labels for the other three tasks to be used during pre-training. For the expression classification (aka "sentiment") task, we simply map the median expression into high, medium and low categories as done in Babjac et al. [2]. Similarly, for the species classification, each sequences' label is just its species name (e.g., *E. coli*, yeast, etc.) to create a multi-class classification problem. Finally, for disorder we simply use the associated disorder value as a continuous variable (see Figure 1).

We note that for each task, we randomly sub-sample stratified partitions of varying labels individually and use these to construct the train/val/test sets. This ensures that there will be distributed expression, species, disorder and translation labels across all facets of the data during pre-training.

B. Models

The chosen base model is the Unified Text-to-Text Transfer Transformer (T5) [17]. The idea of T5 is to handle multi-task pre-training by creating a "unifying" framework for all given tasks. To do so, the inputs are all cast into a "text-to-text" format, where each separate task has an input prompt during pre-training. This text-to-text framework has the advantage of directly applying the same model, objective, training procedure, and decoding process to every task considered. We choose to compare two different pre-trained variations of T5: (i) ProtT5[1], and (ii) ByT5[2]. Each variation is chosen because of the distinct architectural choices in each that make these two ideal candidates to start with.

ProtT5 [5], which was pre-trained on all of Uniref-50 in a self-supervised fashion (no prompts/labels), is shown to

[1] https://huggingface.co/Rostlab/prot_t5_xl_uniref50

[2] https://huggingface.co/docs/transformers/model_doc/byt5

understand some elements of protein shape, which implies it may also have an understanding of factors impacting protein synthesis. We chose the smaller of the two available architectures (equivalent to XL-T5) and further implement a PEFT (Parameter Efficient Fine-Tuning) model utilizing the LoRA (Low Rank Adaptation) optimizer [9], such that we only affect 10% of the pre-trained weights.

As an additional baseline, we utilize another modification of T5, ByT5 [26], which is based on mT5 (similar to Sidi et al. who used mBART [21]). ByT5 modifies mT5 to facilitate handling byte-level encoded words in two ways that are advantageous to our task. It first removes the SentencePiece tokenizer, opting to instead feed UTF-8 bytes directly to the model input layer which removes the need for specialized tokenization. The second modification involves unbalancing the encoder-decoder architecture; specifically, the encoder in ByT5 is 3X larger than the decoder, making it more similar to encoder-only models like BERT which we have utilized previously [2]. We start training from the "small" checkpoint of ByT5 such that we fine-tune a similar number of weights as our ProtT5 model with LoRA.

C. Tokenization

We utilize a separate tokenization strategy for each model. For ProtT5, we overload the standard T5 tokenizer[3] to handle the codon vocabulary and assign tokens for each of the words representing prompts. For ByT5, we utilize the default tokenizer; however, we engage in a pre-processing step to better help ByT5 understand codon language by mapping each codon to a unique unicode character. We do this by using the python function, `chr()`, and ensure that none of the unicode characters mapped to codons overlap with the characters used in prompts and labels. This guarantees that ByT5 will "see" each codon as a unit (i.e. vocabulary 64) rather than as a composite of nucleotides (vocabulary 4, e.g. DNABERT), allowing us to shorten the sequence length for the same result.

D. Pre-Training

Following the T5 paradigm, we initially "pre-train" our models to perform multi-task prediction by casting each task to text-to-text format using the aforementioned data set-up and prepend prompts to both the input and output sequences. For the input sequence we use a simple uniform prompt of format: "[task type] [task] :"; for the output sequence we prompt with just the output task (see Figure 1). Importantly, we separate a 10% subset of translation sequences and label them as "unknown". This allows for generalization to species outside of those included in the pre-train corpus (e.g., orthologs).

We pre-train both ProtT5 and ByteT5 using standard training parameters and train until convergence (~10 epochs for ProtT5 and ~40 epochs for ByT5; graphs not shown). We train our model using Adafactor, with a learning rate of 1e-3 because this was shown to work better when the original T5 model was trained [17]. Additional parameters include weight decay of 0.01, batch size of 8, and max sequence length (and generation length) of 512. The generation length was chosen based on the average sequence length of our data after pre-processing but can be extended to accommodate longer sequences in future iterations of the model.

E. Evaluation Metrics

We utilize four evaluation metrics to assess our model translations: (i) BLEU[4] (represents precision), (ii) ROUGE[5] (represents recall), (iii) METEOR[6] (harmonic mean of precision and recall, aka, F1-score), and (iv) TER[7] (percentage of edits relative to average sequence length). The latter two metrics, while more uncommon in traditional language task evaluation, were chosen as they better penalize words appearing out of order which affect functional protein production. When pre-training we simply calculate global versions of these metrics across the stratified evaluation set. During fine-tuning we calculate these metrics separately for both the generated codon sequence and its amino acid equivalent to determine if the amino acid sequences will create the same functional protein. We further benchmark all classification/regression tasks using the appropriate metrics—either AUROC, F1-score, and Accuracy for classification or Spearman ρ, R^2 correlation and root mean squared error (RMSE) for regression.

F. Benchmarking Individual Tasks

For the most fair benchmarking, we fine-tune each classification/regression task separately without prompts for up to 30 epochs with a lower learning rate between 4e-5 and 5e-6 (stopping early with best performance/convergence). Note that the fine-tuning occurs using the `AutoModelForSequenceClassification` override of T5 that removes the decoder and adds a CLS (classification) layer on top of the pre-trained encoder to make this benchmarking more comparable to previous architectures. We select two additional state-of-the-art models for each task and fine-tune them in the exact same manner (i.e., Adam optimizer, learning rate = [4e-5, 5e-6], weight decay of 0.01, batch size of 8, and sequence length of 512) for up to 30 epochs and report the best evaluation performance using the best performing learning rate and epoch with lowest loss. One of the baselines selected for all tasks is the highly regarded ESM-2 [12]. We then also selected a task-specific baseline for each task: DR-BERT [14] for disorder, CodonBERT [2] for expression/sentiment, and ProtBERT [3] for species.

G. Fine-Tuning for Codon Optimization

We further fine-tune our models to perform "sequence-to-sequence translation" by creating separate models intended to optimize towards a specific organism. Using the paired sequences defined in Section II-A, we remove the prompts and

[3] https://huggingface.co/docs/transformers/model_doc/t5#transformers.T5TokenizerFast

[4] https://huggingface.co/spaces/evaluate-metric/sacrebleu

[5] https://huggingface.co/spaces/evaluate-metric/rouge

[6] https://huggingface.co/spaces/evaluate-metric/meteor

[7] https://huggingface.co/spaces/evaluate-metric/ter

translate from all included species to the species of interest. This effectively makes it such that any input sequence can be translated as we also "mask" some species as "unknown" (see Section II-A) to make the model more generalizable. We start fine-tuning from the best pre-training checkpoint from each CodonT5 model (either ByT5 or ProtT5 init) and reuse the same training parameters and tokenization strategies as defined before in Sections II-C and II-D but with a lower learning rate ([4e-5 to 5e-6]).

H. Post-Processing Codon Optimizations

Generating protein sequences using deep learning is not necessarily subject to the same constraints as nature (i.e., start from start codon, stop at stop codons, etc.); therefore, we perform post-processing for zero-shot evaluations to better compare our generated sequences with the target. We apply a global alignment between the generated sequence and the source sequence with match=1, gap=-1, and mismatch=0 (prioritizing mismatches over gaps). We then fill gaps in the generated sequence with the appropriate codon from the source sequence and remove extraneous codons. This guarantees our final, post-processed generation will be the same length as the target sequence.

I. Zero-Shot Evaluation using the Protein CAT

As an additional unbiased test of the "unknown" species functionality of our translation model, we decided to use the *E. coli* protein, Chloramphenicol acetyltransferase (CAT), which has been experimentally studied in previous codon harmonization efforts [18, 24, 25]. We run a "scrambled" version of the CAT protein (created using random reverse translations, refer to Wright et al. [25]) through our fine-tuned *E. coli* model and assess the results using windowed-CAI [20] since it has been the most popular measure of evolutionarily important (read: fast) codons. We compare the performance against the "wild type" CAT (natural evolutionary preferences) as well as two other previously computed mutants using harmonization [18] with two different codon preference algorithms (%MinMax [25] and Φ [7]) that were previously shown to be among the best performing *in vitro* [18, 22].

III. Results

A. Both CodonT5 pre-trained models are able to achieve good performance for multi-task prediction

Both the ProtT5 and ByT5 initialized models perform well as a codon-based multi-task model (see Table I). ProtT5 significantly outperforms ByT5 in terms of evaluation for translation (Text-to-Text) garnering an impressive BLEU of 0.987, ROUGE of 0.994, METEOR of 0.991 and TER of 0.002 – near perfect scores. While BLEU and ROUGE are not word order dependent and thus likely a little inflated, METEOR and TER also achieve near perfect performance. Specifically, TER of 0.002 implies that only ~2 (character-level) edits are necessary per 1,000 generated codons to perfectly match our target prompts and labels. When we look at the breakdown of pre-training evaluation per task (Table I), both models have different advantages, with ProtT5 doing comparably (but slightly worse) on the classification/regression tasks, and significantly better on the novel task of translation. We discuss reasoning for this result in Section IV.

B. ByT5-initialized CodonT5 outperforms most other models when fine-tuning towards individual tasks

Table II shows a benchmarking of our CodonT5 models against the four comparable transformer baseline models built for individual tasks as well as ESM-2. Interestingly, ByT5 outperforms the fine-tuned ProtT5 and the chosen state-of-the-art baseline models across nearly all tasks. ByT5 achieves a Spearman correlation of 0.830 and 0.694 for disorder and expression prediction, respectively, and an AUROC of 0.814 and 0.983 for sentiment and species classification, respectively. As expected, the next best models for each task are ones specifically designed with that task in mind: DR-BERT (ρ = 0.801) for disorder, CodonBERT (ρ = 0.690, AUROC = 0.818) for expression and sentiment respectively, and ESM-2 (AUROC = 0.934) for species classification. CodonBERT does perform slightly better than CodonT5 in terms of sentiment prediction (as expected [2]); however, our ByT5 initialized model has similar performance in fewer epochs. Initializing from ProtT5 performs worse, although the classification performance is still comparable to other methods. We further note that our CodonT5 models tend to fine-tune more quickly (fewer epochs) than the alternatives considered.

C. ProtT5-initialized CodonT5 generates accurate synonymous codon translations

Chaney et al. previously showed that less common codon choices are conserved using a very diverse set of organisms [4]. To confirm the validity of our ProtT5-initialized codon translation scores using a subset of the organisms included in [4], we fine-tune two separate translation models towards *E. coli* and yeast respectively (see Section II-G for methodology details) and evaluate performance by both the generated codons and the associated amino acid sequence. Importantly, both models achieve a TER == 0.0 for the amino acid evaluation after fine-tuning (data not shown), meaning that the generated codon sequences are on average a perfect amino acid alignment with the target sequence in our comparisons. The codon performance is equally respectable (BLEU, ROUGE, METEOR $>$ 0.5 and TER $<$ 0.5) given the evolutionary distance between some of the paired organisms. The *E. coli* model attains a BLEU score of 0.815, ROUGE score of 0.667, METEOR score of 0.530 and TER of 0.423, and yeast performs similarly with BLEU score of 0.823, ROUGE score of 0.669, METEOR score of 0.541 and TER of 0.430. Hence, for either species we see a large number of overlap in correct codons (BLEU/ROUGE)—implying the correct global codon preferences—with many of the correct synonymous codons being chosen locally (METEOR/TER). Further, a TER around 0.4 implies that less than or equal to 40% of codons are synonymous which is similar to previous methods in selecting similar (but not exactly the same) codon swaps relative to the

TABLE I
BREAKDOWN OF TEXT-TO-TEXT PREDICTION EVALUATIONS FOR ALL PRE-TRAINING TASKS (BEFORE FINE-TUNING) USING OUR CODONT5 PRE-TRAINED MODELS. THE TASKS ARE LISTED IN EXPECTED ORDER OF DIFFICULTY FROM LIKELY EASY (SPECIES CLASSIFICATION) TO UNKNOWN (TRANSLATION/TEXT-TO-TEXT). THE BEST PERFORMING METRIC FOR EACH TASK/MODEL COMBINATION IS UNDERLINED.

Task	Model	BLEU	ROUGE-1	ROUGE-L	METEOR	TER
Species (Classification)	CodonT5 (ByT5 init)	0.977	0.976	0.976	0.964	0.005
	CodonT5 (ProtT5 init)	0.973	0.972	0.972	0.962	0.003
Expression (Classification)	CodonT5 (ByT5 init)	0.886	0.830	0.830	0.860	0.341
	CodonT5 (ProtT5 init)	0.844	0.793	0.793	0.834	0.202
Disorder (Regression)	CodonT5 (ByT5 init)	0.889	0.760	0.760	0.141	0.719
	CodonT5 (ProtT5 init)	0.619	0.500	0.500	0.647	0.500
Translation (Text-to-Text)	CodonT5 (ByT5 init)	0.563	0.683	0.669	0.719	0.665
	CodonT5 (ProtT5 init)	0.987	0.994	0.994	0.991	0.002

TABLE II
BENCHMARKING INDIVIDUAL TASK RESULTS OF OUR PRE-TRAINED CODONT5 MODELS AGAINST SIMILAR STATE-OF-THE-ART MODELS. WE REPORT THE APPROPRIATE METRICS (CLASSIFICATION OR REGRESSION) AT THE EPOCH OF LOWEST EVALUATION LOSS FOR EACH MODEL.

Species					Sentiment (Expression Classification)				
Model	**Epoch**	**AUROC**	**Accuracy**	**F1-Score**	**Model**	**Epoch**	**AUROC**	**Accuracy**	**F1-Score**
ESM-2	2	*0.934*	*0.693*	*0.676*	ESM-2	2	0.804	0.619	0.622
ProtBERT	21	0.518	0.294	0.141	CodonBERT	8	0.818	0.645	0.647
CodonT5 (ByT5 init)	9	0.983	0.877	0.863	CodonT5 (ByT5 init)	5	*0.814*	*0.643*	*0.643*
CodonT5 (ProtT5 init)	19	0.853	0.656	0.552	CodonT5 (ProtT5 init)	7	0.781	0.616	0.618

Disorder					Expression (Regression)				
Model	**Epoch**	**Spearman ρ**	**R^2**	**RMSE**	**Model**	**Epoch**	**Spearman ρ**	**R^2**	**RMSE**
ESM-2	24	0.762	*0.160*	*0.094*	ESM-2	22	0.438	-0.011	673.947
DR-BERT	19	*0.801*	0.158	0.095	CodonBERT	30	*0.690*	*0.062*	*648.820*
CodonT5 (ByT5 init)	16	0.830	0.850	0.040	CodonT5 (ByT5 init)	26	0.694	0.310	556.485
CodonT5 (ProtT5 init)	30	0.268	0.067	0.100	CodonT5 (ProtT5 init)	7	0.135	-0.023	677.853

target organism (e.g., [25]). We note that we omit the results of fine-tuning ByT5 for *E. coli* and yeast as it does not do well with translations (see Table I).

D. Generated translations are in line with previous harmonized sequences for the protein CAT

We further evaluate our results by generating zero-shot translations for the protein CAT (see Section II-I for methodolgy details) using the "unknown" functionality of our model which allows for translating on a protein not accounted for during the initial training. We predict from both our pre-trained CodonT5 models (ByT5 and ProtT5 initialized) as well as a fine-tuned version for *E. coli* (see Section II-G). We calculate the RMSE of the codon usage curves [25] shown in Figure 2 between the wild type sequence and our model translations as well two harmonized mutants which have been shown to work well *in vitro* (High-Φ and %MinMax, see Rodriguez et al. [18]). The RMSE are as follows: Scrambled - 4.566, CodonT5 (ByT5 init) - 5.431, CodonT5 (ProtT5 init) - 4.739, CodonT5 (fine-tuned) - 4.143, %MinMax - 3.472, High-Φ - 3.737. The ByT5 and ProtT5-initialized models do worse than the scrambled, but fine-tuning does better. Our generated sequences are not as close to the wild type CAT as %MinMax and High-Φ using RMSE but are similarly close when looking at rank differences (data not shown); we further note that the alternative sequences (%MinMax, High-Φ) were harmonized to the wild type while our generated sequences used a randomization of the codon choices (aka "shuffled"). We therefore generated codon usage subprofiles without knowledge of the wild type, and the generated sequences are better than the scrambled mutant (see Figure 2), which is promising for future experimental studies.

IV. DISCUSSION

Based on our results, ByT5-initialized CodonT5 seems to better handle generalized, codon, multi-task modeling and fine-tuning, whereas ProtT5-initialized CodonT5 tends to work better for accurate translations. This is somewhat expected given ByT5's original language-based pre-training more geared towards classification/regression of words and ProtT5's masked-language model objective intended to understand the pattern of amino acid sequences. We note that considering ByT5 and ProtT5 fine-tuned towards each task individually (i.e., without our multi-task pre-training) performs much worse (data not shown) and in some cases does not converge. Further, ByT5 fine-tunes far better on the benchmarking tasks (or nearly equivalent for the task of "sentiment") to state-of-the-art baseline models: DR-BERT, CodonBERT, ESM-2, and ProtBERT. This suggests that the selected tasks help create a shared context that is beneficial compared to other plain masked language model pre-training (e.g., ProtBERT, ESM) consistent with the intuition of the original T5 model [17].

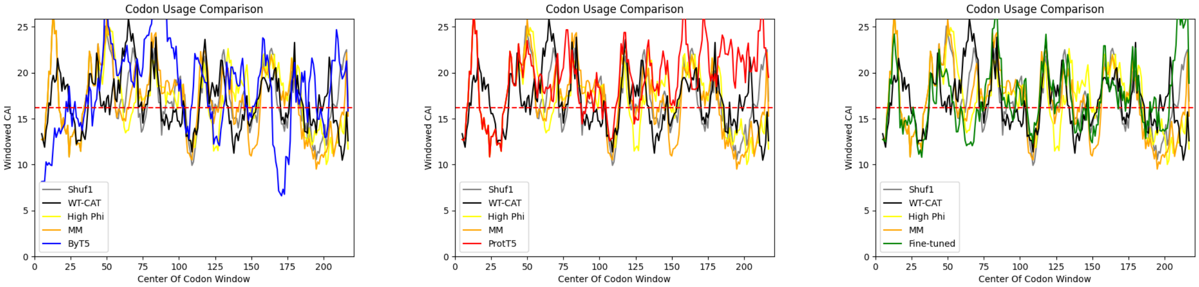

Fig. 2. A comparison of windowed codon usage bias between our generated sequences (after post-processing), the original wild type sequence (CAT - black), and two harmonized mutants of CAT using CHARMING with codon usage metrics of particular interest (%MinMax - orange, High-Φ - yellow) [25]. The plots are ordered from top to bottom: ByT5-initialized CodonT5 (blue), ProtT5-initialized CodonT5 (red), CodonT5 fine-tuned towards *E. coli* (green) relative to windowed-CAI [20] calculated using the code from Wright et al. [25] with the default window size of 9. CAI was chosen since its the most popular measure of evolutionary important codons (fast codons) and therefore the most consistent with our multi-task model that includes expression data.

We suspect ProtT5-initialized CodonT5 does not fine-tune as well for classification/regression (see Tables II) because of its pre-training being LoRA-based and the fine-tuning involving only the encoder, as its pre-training performance was similar to ByT5 (see Table I). Similarly, we speculate that ByT5 does not generate good translations because it lacks understanding of general intermediary amino acid orderings (unlike ProtT5); however, it is successful in making select synonymous codon swaps (after post-processing) which supports that it can pick up codon preference patterns from the multi-task pre-training.

Additionally we note when looking specifically at translations, each model (ByT5-initialized CodonT5, ProtT5-initialized CodonT5, and CodonT5 fine-tuned) encapsulates different evolutionary preferences. Both ByT5-initialized CodonT5 (blue) and ProtT5-initialized CodonT5 (red) appear to capture the codon pattern of CAT using the well-established data-driven CAI measure, and therefore lean towards more preferred codons given that CAI is trained from known high expression genes. The fine-tuned CodonT5 (green) also tends to mirror High-Φ, which is a computational model for predicting codons under natural selection [19] (and therefore more likely highly expressed codons). This may be beneficial in future studies to cater towards alternative research objectives.

While there is no "best" method, our ProtT5-initialized CodonT5 model fine-tunes near perfectly on data comparable to the initial training set (data not shown) and generates good translations on a protein outside the initial dataset (CAT; Figure 2), which is known to have clear codon sensitivity with respect to its functional protein production [24]. Further, translating ten random reverse translations (created using code from Wright et al. [25] and inspired by our experimental approach in [18]) always results in a synonym that is predicted to be translated faster than random, consistent with "wild type" codon choices (data not shown). It follows that similar to CAI [20] and Φ [19], our models may have a better overall understanding of evolutionary codon preferences linked to expression; however, it is still unclear whether our CodonT5 understands relevant co-translational folding intervals where the "tempo" of translation is also important. To truly verify this, future work will be needed to express CAT *in vivo* using our generated sequences relative to known non-functional variants.

V. Conclusion and Future Work

We presented a first-ever look at using multi-task pre-training to aid in translating codons to codon preferences of another organism. Significantly, we have established that selective multi-task pre-training can improve "learning" of codon preference patterns when compared to single task state-of-the-art models on the four tasks considered here. Further, both our pre-trained and fine-tuned CodonT5 models capture different codon usage patterns and lend varying translations that are consistent with previously computed mutants for a sequence with clear codon "sensitivity," i.e. swapping codons can severely affect functional protein production [18]. This is a valuable finding as it suggests our model could be generalizable to generic species outside its pre-training (e.g., human).

While our models have proven effective on a small scale, we recognize that more extensive benchmarking, as well as *in vitro* experimentation, need to be performed particularly regarding human data which was not included in this initial study. We further recognize that in terms of translation, conventional methods of synonymous codon replacement can generate 100% correct sequences whereas our methodology currently requires post-processing to produce a functional sequence. After evaluating two initial models (ByT5 and ProtT5), we anticipate certain architectural changes can address the concerns in ByT5 and allow it to better converge across all tasks, including: (i) starting from a masked-language codon pre-training (of dataset similar to Uniref) prior to the multi-task training; (ii) creating multiple checkpoints (notably "base" and "large") in addition to the "small" presented here; (iii) adding in a custom loss when fine-tuning on translations that accounts for both amino acid sequences and codon preferences of the target organism; and, (iv) adding additional tasks to strengthen the shared embedding space during pre-training. The tasks selected for this study were less comprehensive as we focused specifically on codon optimization; however, this type of architecture would likely prove beneficial for many common protein language benchmarks. We look forward to future studies which may expand upon the ideas presented here regarding both multi-task pre-training and generative codon optimization.

Acknowledgments

ANB was supported by the NSF GRFP. We thank the UTK HPC team for their support of our hardware requirements.

References

[1] Ashley Babjac, Jun Li, and Scott Emrich. "Fine-Grained Synonymous Codon Usage Patterns and their Potential Role in Functional Protein Production". In: *2021 IEEE International Conference on Bioinformatics and Biomedicine (BIBM)*. IEEE. 2021, pp. 2187–2193.

[2] Ashley Nicole Babjac, Zhixiu Lu, and Scott J Emrich. "CodonBERT: Using BERT for Sentiment Analysis to Better Predict Genes with Low Expression". In: *Proceedings of the 14th ACM International Conference on Bioinformatics, Computational Biology, and Health Informatics*. 2023, pp. 1–6.

[3] Nadav Brandes et al. "ProteinBERT: a universal deep-learning model of protein sequence and function". In: *Bioinformatics* 38.8 (2022), pp. 2102–2110.

[4] Julie L. Chaney et al. "Widespread position-specific conservation of synonymous rare codons within coding sequences". In: *PLOS Computational Biology* 13.5 (May 2017), pp. 1–19.

[5] Ahmed Elnaggar et al. "Prottrans: Toward understanding the language of life through self-supervised learning". In: *IEEE transactions on pattern analysis and machine intelligence* 44.10 (2021), pp. 7112–7127.

[6] Aysu Ezen-Can. "A Comparison of LSTM and BERT for Small Corpus". In: *arXiv preprint arXiv:2009.05451* (2020).

[7] M.A. Gilchrist et al. " Estimating gene expression and codon-specific translational efficiencies, mutation biases, and selection coefficients from genomic data alone". In: *Genome Biology and Evolution* 7.6 (May 2015), pp. 1559–1579.

[8] Andreas Grote et al. "JCat: a novel tool to adapt codon usage of a target gene to its potential expression host". In: *Nucleic acids research* 33.suppl_2 (2005), W526–W531.

[9] Edward J Hu et al. "Lora: Low-rank adaptation of large language models". In: *arXiv preprint arXiv:2106.09685* (2021).

[10] Rishab Jain et al. "ICOR: improving codon optimization with recurrent neural networks". In: *BMC bioinformatics* 24.1 (2023), p. 132.

[11] Sizhen Li et al. "CodonBERT: Large Language Models for mRNA design and optimization". In: *bioRxiv* (2023), pp. 2023–09.

[12] Zeming Lin et al. "Evolutionary-scale prediction of atomic-level protein structure with a language model". In: *Science* 379.6637 (2023), pp. 1123–1130.

[13] Ari Löytynoja. "Phylogeny-aware alignment with PRANK and PAGAN". In: *Multiple Sequence Alignment: Methods and Protocols* (2021), pp. 17–37.

[14] Ananthan Nambiar et al. "DR-BERT: a protein language model to annotate disordered regions". In: *bioRxiv* (2023), pp. 2023–02.

[15] Carlos Outeiral and Charlotte M Deane. "Codon language embeddings provide strong signals for use in protein engineering". In: *Nature Machine Intelligence* 6.2 (2024), pp. 170–179.

[16] Pere Puigbo et al. "OPTIMIZER: a web server for optimizing the codon usage of DNA sequences". In: *Nucleic acids research* 35.suppl_2 (2007), W126–W131.

[17] Colin Raffel et al. "Exploring the limits of transfer learning with a unified text-to-text transformer". In: *The Journal of Machine Learning Research* 21.1 (2020), pp. 5485–5551.

[18] Anabel Rodriguez et al. "Synonymous codon substitutions regulate transcription and translation of an upstream gene". In: *bioRxiv* (2022). DOI: 10.1101/2022.08.05.502938. eprint: https://www.biorxiv.org/content/early/2022/08/06/2022.08.05.502938.full.pdf. URL: https://www.biorxiv.org/content/early/2022/08/06/2022.08.05.502938.

[19] Premal Shah and Michael A. Gilchrist. "Explaining complex codon usage patterns with selection for translational efficiency, mutation bias, and genetic drift". In: *Proceedings of the National Academy of Sciences* 108.25 (2011), pp. 10231–10236.

[20] P.M. Sharp and W.-H. Li. "The codon adaptation index-a measure of directional synonymous codon usage bias, and its potential applications". In: *Nucleic Acids Research* 15.3 (1987), pp. 1281–1295.

[21] Tomer Sidi et al. "Predicting gene sequences with AI to study evolutionarily selected codon usage patterns". In: *bioRxiv* (2024), pp. 2024–02.

[22] Tien T. Sword et al. "Profiling expression strategies for a type III polyketide synthase in a lysate-based, cell-free system". In: *Scientific Reports* 14.1 (2024), p. 12983.

[23] Alan Villalobos et al. "Gene Designer: a synthetic biology tool for constructing artificial DNA segments". In: *BMC bioinformatics* 7 (2006), pp. 1–8.

[24] I.M. Walsh et al. "Synonymous codon substitutions perturb cotranslational protein folding *in vivo* and impair cell fitness". In: *Proceedings of the National Academy of Sciences* 117.7 (2020), pp. 3528–3534.

[25] Gabriel Wright et al. "CHARMING: Harmonizing synonymous codon usage to replicate a desired codon usage pattern". In: *Protein Science* 31.1 (2022), pp. 221–231.

[26] Linting Xue et al. "Byt5: Towards a token-free future with pre-trained byte-to-byte models". In: *Transactions of the Association for Computational Linguistics* 10 (2022), pp. 291–306.