RNA translation model based on RNABERT and bert2bert

INTRODUCTION

mRNA vaccines are a promising means of vaccine development and one of the key novel strategies for dealing with outbreaks of infectious diseases. mRNA vaccines have the advantages of low production cost, no metabolism and toxicity, and easy modification, but they also have limitations, such as translation instability. Studies have shown that mRNA's stability and translation efficiency can be improved by optimizing codons in the untranslated region (UTR) region and open reading frame (ORF) of mRNA. Therefore, the sequence optimization of mRNA vaccines is an urgent problem for covid-19 vaccines. Deep-learning systems can enhance the quality of biomarker assays targeting DNA sequences [1]. We map machine translation (MT) tasks to UTR region optimization. Transformer, an attention-based architecture that has achieved state-of-the-art performance in most natural language processing (NLP) tasks [2]. Transformer-based models are good at capturing contextual dependencies, but the memory and computational requirements of self-attention can affect the efficiency of transformer-based models applied to long sequences. Transformer-based models that can accommodate long sequences (e.g., DNA, RNA) have been explored and are burgeoning. We propose OUT (ORF to UTR translator), a novle NLP-based MT model from open reading frame (ORF) to UTR and make the translation model adaptable to ultra-long ORF sequences. Specifically, it can be divided into two steps: find the optimal encoding method of ORF and build a more effective translation model that adapts to the length of long gene sequences.

METHODOLOGY

Problem Setup

We analogized the UTR optimization task to a machine translation (MT) task, where an MT model is trained to output the corresponding optimal UTR sequence. For notation, we define the gene sequence (the input ORF) G as a series of n tokens, $G = \{g_1, g_2, \ldots, g_n\}$. Simultaneously, we define the real output UTR U as a target sequence consisting of m tokens, $U = \{u_1, u_2, \ldots, u_m\}$. The ultimate goal of our model is to maximize the probability $P(U|G)$. Wahab et al. revealed that genetic data can be used as language, whether in DNA or RNA samples within cellular structures [3]. Accordingly, we adopted the machine translation model of natural language processing to divide the gene sequences into source and target languages and perform the machine translation task to optimize the sequences.

The OUT pipeline

The whole structure of UTR generation is shown in **Figure 1A**. It mainly consists of a tokenizer and an MT model. In recent years, pre-training sequence-to-sequence models and fine-tuning them on downstream tasks such as machine translation and text summarization has proven to be state-of-the-art in natural languages. However, pre-trained models regarding gene sequence data are still scarce due to the lack of vast amounts of data in proprietary domains and computational resources. The dataset we filtered through the NCBI human gene dataset was not sufficient to meet the pre-training criteria. Rather than pre-training from scratch using mBart, we adopted the pre-trained weights of DNABERT by Ji et al. on the human genome regarding different k-mer [4], which can be well applied since the RNA sequence differs from the DNA sequence by only one base (thymine to uracil) while the syntax and semantics remain essentially unchanged. In this work, we first fine-tuned a separate encoder model as RNABERT using selected ORF and UTR pairs, while importing the pre-training weights of DNABERT. Then the warm-start sequence-to-sequence (bert2bert) model is equipped with an encoder and decoder for RNABERT initialization to perform the biological sequence translation task.

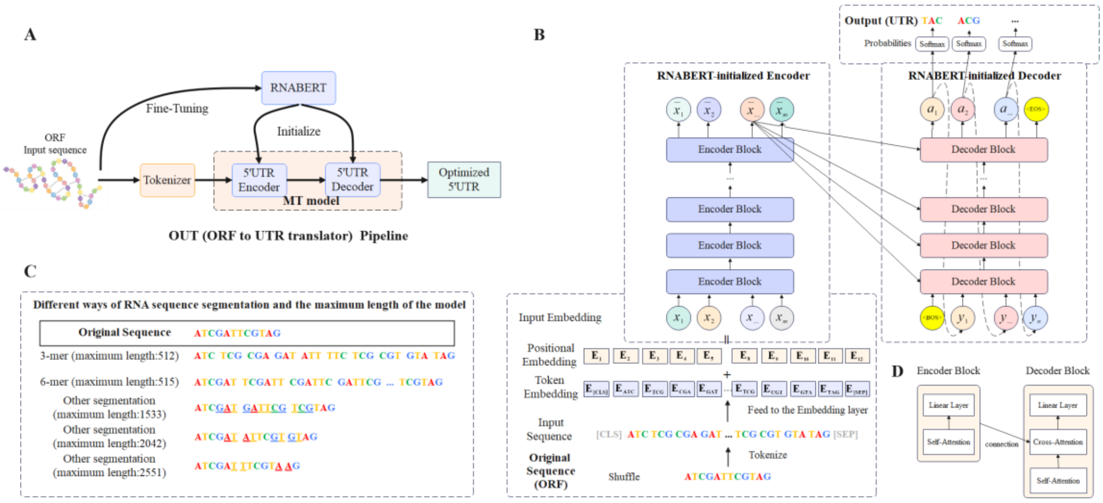
A
ORF
Input sequence
Fine-Tuning
RNABERT
Initialize
Tokenizer
5'UTR Encoder
5'UTR Decoder
MT model
Optimized 5'UTR
OUT (ORF to UTR translator) Pipeline
C
Different ways of RNA sequence segmentation and the maximum length of the model
Original Sequence ATCGATTCGTAG
3-mer (maximum length:512) ATC TCG CGA GAT ATT TTC TCG CGT GTA TAG
6-mer (maximum length:515) ATCGAT TCGATT CGATTC GATTCG ... TCGTAG
Other segmentation (maximum length:1533) ATCGAT GATTCG TCGTAG
Other segmentation (maximum length:2042) ATCGAT ATTCGT GTAG
Other segmentation (maximum length:2551) ATCGAT TTCGTA AG
B
Output (UTR) TAC ACG ...
Probabilities Softmax Softmax Softmax
RNABERT-initialized Encoder
RNABERT-initialized Decoder
Encoder Block
Encoder Block
Encoder Block
Encoder Block
Decoder Block
Decoder Block
Decoder Block
Decoder Block
$\bar{x}_1$ $\bar{x}_2$ $\bar{x}_{...}$ $\bar{x}_m$
a_1 a_2 $a_{...}$ <EOS>
x_1 x_2 $x_{...}$ x_m
<BOS> y_1 $y_{...}$ y_n
Input Embedding
Positional Embedding E_1 E_2 E_3 E_4 E_5 E_8 E_9 E_{10} E_{11} E_{12}
Token Embedding $E_{[CLS]}$ E_{ATC} E_{TCG} E_{CGA} E_{GAT} ... E_{TCG} E_{CGT} E_{GTA} E_{TAG} $E_{[SEP]}$
Feed to the Embedding layer
Input Sequence [CLS] ATC TCG CGA GAT ... TCG CGT GTA TAG [SEP]
Tokenize
Original Sequence (ORF)
Shuffle
ATCGATTCGTAG
D
Encoder Block
Decoder Block
Linear Layer
Self-Attention
connection
Linear Layer
Cross-Attention
Self-Attention

Figure 1: Integrated OUT pipeline and model details. (A) OUT Pipeline for machine translation (MT) of gene sequences (take 5'UTR as an example). (B) MT model. Architecture diagram of RNAbert2RNAbert and the flow chart of the sequence translation task of the model. (C) Tokenizer. Different representations of the translated input sequences, namely different tokenization methods. (D) The cross-attention mechanism connecting encoder and decoder.

Input Representation

The input to our model is the ORF sequence $G = \{g_1, g_2, \ldots, g_n\}$. The tokenized ORF sequences consist of tokens in a pre-trained DNABERT dictionary, each element of which is represented as a k-mer token, regardless of subsequence types. Next, similar to BERT [5], we place a special separator token embedding [SEP] at the end of each G, and a label [CLS] at the beginning indicating the meaning of the whole sentence, thus forming the model input, as shown in **Figure 1B**. The Bert-based models achieve state-of-the-art performance on NLU tasks by mapping input sequences to output sequences with a priori known output lengths. However, since the output of the generation class task (translation task) does not depend on the input, it is not possible to use Bert-based models directly for the translation task. Therefore, we replace the [CLS] and [SEP] of the input in the training phase of the decoder with the translation initial position marker [BOS] and the translation end position [EOS]. Finally, each input embedding is summed with the corresponding positional embedding and token embedding to prepare the final input to the model.

MT model Architecture and Tokenization

Devlin et al. proposed BERT as a bidirectional transformer network pretrained on a large corpus with two pretraining objectives: masked language modeling and next sentence prediction [5]. BERT is an encoder-only model, however pre-trained checkpoints can be used to initialize the encoder and decoder parts with pre-trained model checkpoints, allowing a sequence-to-sequence model to be built [6]. The bert2bert model from Rothe et al. achieved this possibility and produced exceptional results on MT and Sentence Fusion [6]. Based on the SOTA performance of DNABERT in a variety of downstream tasks (e.g., identifying transcription factor binding sites), we developed the RNAbert2RNAbert model for the first time and applied the bert2bert framework to the gene sequence translation task. We fine-tuned our data based on DNABERT weights to construct the RNABERT model. The OUT adopt bert2bert structure which is a pair of RNABERT encoder and RNABERT decoder. In this architecture, the encoder is the same as the RNABERT, which consists of 12 Transformer layers, each with 768 hidden cells and 12 attention headers. The input sequence $G = \{g_1, g_2, \dots, g_n\}$ is mapped to a contextualized encoded sequence $G' = \{g'_1, g'_2, \dots, g'_n\}$ by going through 12 transformer blocks. The decoder layers are also the same as RNABERT, with a few changes. Cross-attention layers are added between self-attention and feed-forward layers in order to condition the decoder on the contextualized encoded sequence (e.g., the output of the RNABERT model) as shown in **Figure 1D**. While warm-starting the decoder, only the cross-attention layer weights are initialized randomly. The output layer containing Softmax is added on top of the decoder component to define a conditional probability distribution while generating k-mer outputs.

In practice, the ORF sequences are first tokenized before starting machine translation. Bostrom et al. demonstrated that tokenization is a crucial feature of MT models because the tokenization approach directly determines the input tokens and therefore may affect the model's performance [5]. We provided different splitting methods so that the base RNABERT can accommodate longer sequences **(Figure 1C)**. We tokenized ORF sequences with an approximate k-mer representation to make RNABERT hold up to 2551 valid tokens instead of the original 510 valid segments. After that, the input embedding consisting of the tokenized sequence and the rest of the embedding is fed into the encoder block, the result of the encoder block is fed to the decoder block, which in turn generates the output translated UTR.

Comparison of the OUT and RNABart

OUT outperforms RNABart in BLEU because of the following four reasons. Firstly, OUT pre-trained on a larger dataset. It utilizes DNABERT pre-trained weights and we fine-tune them using our data (all ORF and UTR pairs), therefore it is trained using priori parameters. Due to the lack of an adapted Bart-based [7] pre-training model and the domain shift, RNA-Bart must be trained from scratch. Secondly, OUT has a more stable encoder since it adopts the bert2bert structure instead of the auto-regressive model used in Bart. Thirdly, RNA-Bart performs tokenization using the WordPiece model [8] and produces a large dictionary containing 70,000 tokens, which may lead to data insufficiency in representing tokens. On the contrary, OUT uses the separation of k-mer, and the largest 6-mer dictionary has 4101 tokens, which can continue to be well-trained. Finally, Bart contains roughly 10% more parameters than the equivalently sized BERT model [9]. OUT is more efficient than RNA-Bart for training.

BIBLIOGRAPHY

Uncategorized References

1. Esteva, A., et al., *A guide to deep learning in healthcare.* Nature Medicine, 2019. **25**(1): p. 24-29.
2. Vaswani, A., et al., *Attention is all you need.* 2017. **30**.
3. Wahab, A., et al., *DNA sequences performs as natural language processing by exploiting deep learning algorithm for the identification of N4-methylcytosine.* 2021. **11**(1): p. 1-9.
4. Ji, Y., et al., *DNABERT: pre-trained Bidirectional Encoder Representations from Transformers model for DNA-language in genome.* 2021. **37**(15): p. 2112-2120.
5. Devlin, J., et al., *Bert: Pre-training of deep bidirectional transformers for language understanding.* 2018.
6. Rothe, S., S. Narayan, and A.J.T.o.t.A.f.C.L. Severyn, *Leveraging pre-trained checkpoints for sequence generation tasks.* 2020. **8**: p. 264-280.
7. Baykara, B. and T.J.N.L.E. Güngör, *Turkish abstractive text summarization using pretrained sequence-to-sequence models.* 2022: p. 1-30.
8. Wu, Y., et al., *Google's neural machine translation system: Bridging the gap between human and machine translation.* 2016.
9. Lewis, M., et al., *Bart: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension.* 2019.