Published on Aug 08, 2024 by Arcadia Science **DOI**: 10.57844/arcadia-fa56-ee23

Using protein language models to predict coding and non-coding transcripts with plm-utils

We explored the use of embeddings from protein language models to distinguish between genuine and putative coding open reading frames (ORFs). We found that an embeddings-based approach (shared as a small Python package called plm-utils) improves identification of short ORFs.

Contributors (A-Z)

Audrey Bell, Adair L. Borges, Feridun Mert Celebi, Keith Cheveralls, Megan L. Hochstrasser, Gilad Mishne, Taylor Reiter

Version 2 · Mar 31, 2025

Purpose

We're interested in detecting small open reading frame (sORF)-encoded, bioactive peptides in transcriptomes. sORFs are open reading frames that contain fewer than 300 nucleotides and often use alternate start codons. Computationally detecting real sORFs is challenging, and we wanted to more accurately detect sORFs that encode functional peptides.

We hypothesized that latent information in the embeddings of large protein language models might contain information about the coding propensity of amino acid sequences, even though this wasn't the original use case for such models. We were encouraged toward this line of thinking because other peptide classification tools that identify cleavage peptides **[1]** and predict peptide bioactivity **[2]** have successfully used large protein language models to improve classification accuracy.

We conducted multiple tests across different datasets and observed increased accuracy over leading tools when we applied these innovations to predicting sORF coding potential. On all transcripts from a set of 16 diverse research organisms, our tool performed comparably to the leading tool, RNAsamba. However, our method significantly outperformed that tool for short sequences. Additionally, on the RNAChallenge dataset, where most tools struggle, we achieved an accuracy of 33% compared to the average tool accuracy of 11%. While our approach improves accuracy on this challenging prediction task, the overall accuracy indicates that there's still work to be done.

We packaged our approach as a small Python package called “plm-utils.” Using the Python package infrastructure improved the usability and portability of our tool and will allow us to expand the package in the future if it proves useful.

- This pub is part of the **project**, “Software: Implementing useful and innovative computing.” Visit the project narrative for more background and context.
- The **plm-utils Python package** is available in this GitHub repository.
- The **code** to **train** and **evaluate** the sORF plm-utils **model** is available here.

The context

Advances in ribosome profiling and mass spectrometry have experimentally demonstrated that some small open reading frames (sORFs) are not random sequences in genomes but lead to functional products **[3]**. This has spurred a greater appreciation for and interest in the coding potential of sORFs. Most genomes encode many sORFs, only a fraction of which are transcribed and even fewer translated. Most

transcribed sORFs occur in the 5′ or 3′ UTR of the main coding domain sequence in a transcript and perform a regulatory role by impacting the translation of the mRNA **[4]**. However, some sORFs encode translated peptides with functional roles as small proteins. The majority of sORFs that encode peptides have been identified in presumed long non-coding RNAs **[4][5]**, although some have also been found upstream of, downstream of, and overlapping with, transcripts with longer coding domain sequences **[4]**.

sORFs are alternately referred to as “short” open reading frames **[6]**, while functional peptide products are referred to as sORF-encoded polypeptides (SEPs or sPEPs), microproteins, or micropeptides **[7]**. These peptides are genomically encoded by open reading frames of fewer than 300 nucleotides (100 codons) and are synthesized via DNA transcription and ribosomal translation. Most sORFs use codons that differ from the traditional start codon (AUG) by one nucleotide (UUG, CUG, GUG, and ACG) **[8]**.

Historically, sORFs have been underrepresented in protein annotations **[9]**. sORFs occur frequently throughout genomes, so several heuristic filters are used in tools that predict protein-coding regions to reduce false-positive annotations **[7]**. These filters include length cutoffs of 300 nucleotides **[10]** and the use of the AUG (ATG) start codon **[11]**. Both of these filters preclude the computational annotation of sORFs because sORFs are shorter than 300 bases and the majority start with non-AUG start codons **[8][9]**.

The current tool space

Many computational tools have been developed to identify sORFs in sequencing data. The majority perform sORF discovery on either genome or transcriptome assemblies. These tools generally use evolutionary signatures or sequence heuristics to classify coding versus non-coding sequences.

Tools like phyloCSF **[12]**, PhastCons **[13]**, and micPDP **[14]** use genome alignments and codon substitution patterns to identify sORFs. The three main limitations to these tools are the requirement for a genome assembly, the lack of built-in evidence that the predicted sORF gets transcribed, and dependence on cross-species conservation (in some species, sORFs encode evolutionarily young proteins that aren’t conserved in other species **[6]**).

The other common strategy for computational sORF identification is to predict whether a transcript (or an ORF predicted on a transcript) is coding or non-coding. These tools either use heuristics like codon substitution and nucleotide composition or train machine learning algorithms to predict whether a transcript or an ORF contains a coding sequence. Some tools, like MiPepid or sORFfinder **[15][16]**, are trained specifically on short sequences, while others, like RNAsamba and DeepCPP **[17][18]**, are trained on all transcripts but perform well on sORFs. In both cases, these models often struggle with small training datasets or heuristics that do not generalize to other species or sequence types **[19]**.

Our approach

Foundational models of proteins like AlphaFold2 **[20]** and Evolutionary Scale Modeling (ESM) **[21]** have revolutionized computational approaches to protein research **[22]**. Protein language models are trained on large numbers of protein sequences and other information like multiple sequence alignments or protein structures. After "learning" patterns in this original data, a model can ingest new protein sequences and relate them to the existing information in the model in a process called embedding. In the case of ESM, an embedded protein sequence is represented as a numerical vector, typically a high-dimensional array of floating-point numbers. While embeddings are not directly interpretable by humans, they capture information about the structure of a protein, including orthogonal attributes that correlate with structure, like function **[21][23][24]**.

We wrote a Python package called plm-utils (short for protein language model utilities) that provides a basic set of tools for generating and analyzing embeddings of protein sequences using pre-trained protein language models. It currently only works with ESM2 models, but we may expand it in the future if doing so opens new use cases. If other protein language models would be helpful in your research, please let us know by commenting here or posting an issue on the plm-utils GitHub repository.

While we set this package up as a general tool for using the information in protein language models to improve protein prediction tasks, we developed it specifically to predict whether a transcript is coding or non-coding. We posited that there may be latent information in protein sequences that is not currently used by other tools. This information could be extracted by protein language models, which might help in the classification of coding transcripts. We thought this might particularly be true for

sORFs because embeddings of protein language models have helped improve other difficult tasks with peptides like predicting cleavage peptides **[1]** and annotating peptide bioactivity **[2]**.

The resource: plm-utils

> The **plm-utils Python package** is available in this GitHub repository (DOI: 10.5281/zenodo.12775178).

Plm-utils is a small Python package that includes functions for working with protein language models. Currently, it contains code for building binary classifiers from labeled data using ESM2 embeddings. It also contains helper functions for our first use case, predicting whether a transcript is coding or non-coding.

First use case: predicting coding vs. non-coding transcripts

For the task of classifying coding vs. non-coding transcripts, plm-utils first uses orfipy to find and translate the longest open reading frame on each contig **[25]**. Plm-utils considers multiple potential start codons (AUG, UUG, CUG, GUG, and ACG) as ORFs in general and sORFs in particular can use any of these **[8][26]**. After translating all possible ORFs, we retain only the longest putative ORF from each transcript, assuming that this ORF is the one most likely to be genuine and encode a bioactive protein. Next, plm-utils embeds the putative translated ORF sequences in the ESM2 model embedding space (esm2_t6_8M_UR50D). Plm-utils then uses these embeddings and ground-truth labels to train a random forest classifier. In this case, a classifier is trained to predict whether the ORFs were translated from a coding or non-coding transcript. This results in a model that predicts whether a given amino acid sequence represents a genuine ORF. In more precise terms, the model classifies a given amino acid sequence as “coding” or “non-coding” based on its similarity to the longest ORFs derived from the coding and non-coding transcripts in the training dataset. The primary output of plm-utils is a TSV indicating whether a transcript is coding (positive) or non-coding (negative).

To apply this approach to sORFs specifically, we added a length filtering step after ORF prediction and before embedding and coding vs. non-coding classification.

Throughout this pub, we evaluate the capacity of plm-utils to differentiate between coding and non-coding transcripts, serving as a preliminary test of its effectiveness. To conduct a thorough assessment, we developed several models. The performance of each model varies depending on the specific attributes of the training data. This means that a model trained on all coding ORFs in a transcriptome will perform differently than a model trained on only sORFs. However, every model created using plm-utils is compatible with any sequencing data that ESM can process (the maximum sequence length ESM2 can handle is 1,024 amino acids).

Potential future use cases

We set up the plm-utils Python package so that it can be easily adapted to future use cases. At the moment, we have building blocks in place for embedding sequences, training, and making predictions from a binary classifier. We think this setup could be well suited for predicting whether a protein has a specific function or for predicting traits of a protein such as whether it is membrane-bound. Users would first need to build a new model using labeled training data for new use cases. This model could then be used to predict the traits of new, unseen data.

If you have a use case that would require additional functions in the plm-utils package, we would love to hear your needs either as a comment on this pub or as an issue on the GitHub repository.

Plm-utils is better at classifying short coding sequences than RNAsamba

To assess whether protein language models improve the classification of coding sequences over existing tools, we first compared the performance of plm-utils models against RNAsamba models (version 0.2.5). We chose to compare against RNAsamba because it performed well across various prediction tasks when benchmarked against other tools **[19]**.

To assess model performance with diverse sequencing data, we selected a set of 16 species (Table 1) for which high-quality annotated reference transcriptomes were available on Ensembl. We then trained models separately on transcriptomes from each of the 16 species using the above-mentioned procedure. We used each of the resulting 16 models to make predictions for each of the other 15 species. Note that we did not split the transcriptomes into training and test sets; we trained models on all transcripts from one species and then made predictions for all transcripts from each of the other species.

Species	Abbreviation	Common name	Kingdom	Class
Apis mellifera	Amel	Honey bee	Animals	Insects
Arabidopsis thaliana	Atha	Thale cress	Plants	Eudicots
Caenorhabditis elegans	Cele	Roundworm	Animals	Chromadore
Dictyostelium discoideum	Ddis	Slime mold	Protozoa	Mycetozoa
Drosophila melanogaster	Dmel	Fruit fly	Animals	Insects
Danio rerio	Drer	Zebrafish	Animals	Ray-finned fi
Gallus gallus	Ggal	Chicken	Animals	Birds
Homo sapiens	Hsap	Human	Animals	Mammals
Mus musculus	Mmus	Mouse	Animals	Mammals
Oryza indica	Oind	Rice	Plants	Monocots
Rattus norvegicus	Rnor	Rat	Animals	Mammals
Saccharomyces cerevisiae	Scer	Baker's yeast	Fungi	Saccharomy
Schizosaccharomyces pombe	Spom	Fission yeast	Fungi	Schizosacch
Tetrahymena thermophila	Tthe	Ciliate protozoan	Protozoa	Ciliates
Xenopus tropicalis	Xtro	Western clawed frog	Animals	Amphibians
Zea mays	Zmay	Corn	Plants	Monocots

Table 1

Species used to train and evaluate plm-utils and RNAsamba models in the task of predicting coding versus non-coding transcripts.

We performed this procedure for both plm-utils models and RNAsamba models. We calculated the performance of each model using Matthew's correlation coefficient, a measure that quantifies the quality of binary classifications, ranging from –1 (perfectly

wrong; worse than random) through 0 (no better than random) to +1 (perfect prediction). The RNAsamba models (average MCC 0.51) slightly outperformed the plm-utils models (average MCC 0.43) when trained and evaluated on all transcripts (Figure 1, A and C). However, the plm-utils models (average MCC 0.52) significantly outperformed the RNAsamba models (average MCC 0.15) when the models were trained and evaluated only on transcripts whose longest putative ORF was an sORF (< 100 amino acids) (Figure 1, B and D).

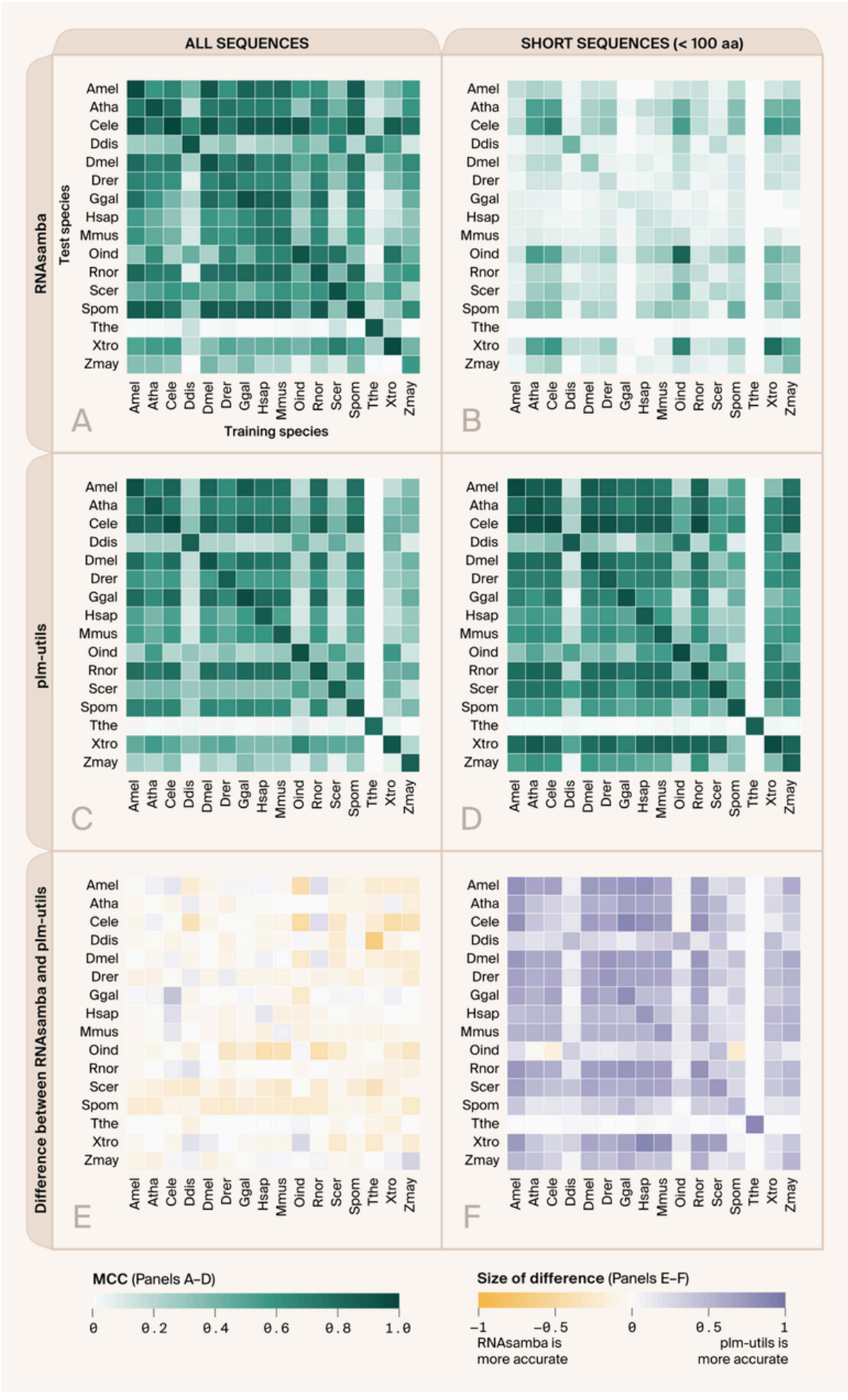
ALL SEQUENCES
SHORT SEQUENCES (< 100 aa)
RNAsamba
plm-utils
Difference between RNAsamba and plm-utils
Test species
Training species
Amel
Atha
Cele
Ddis
Dmel
Drer
Ggal
Hsap
Mmus
Oind
Rnor
Scer
Spom
Tthe
Xtro
Zmay
A
B
C
D
E
F
MCC (Panels A–D)
0
0.2
0.4
0.6
0.8
1.0
Size of difference (Panels E–F)
−1
−0.5
0
0.5
1
RNAsamba is more accurate
plm-utils is more accurate

Figure 1

Comparison of species models trained by RNAsamba (A, B) or plm-utils (C, D).

(A–D) Heatmaps of Matthew's correlation coefficient (MCC) depicting the performance of models trained with whole transcriptomes or short sequences (< 100 amino acids) alone. Model performance is plotted as 16 × 16 heatmaps in which the x-axis corresponds to the species on which the model was trained and the y-axis to the species on which model performance was evaluated. Higher values (dark green) indicate more accurate predictions, while lower values (white) indicate less accurate predictions.

(E–F) We subtracted the MCCs (`plm-utils MCC – RNAsamba MCC`) to compare the two approaches. Positive values (purple) indicate when plm-utils performed better. While the two tools performed similarly for the general task of predicting coding versus non-coding sequences, plm-utils outperforms RNAsamba for predicting short coding sequences.

Our experimental setup comparing plm-utils and RNAsamba has two differences that arise because the tools work differently. First, the models don't use the same sequence data to make predictions. Although both models predict whether a transcript is coding or non-coding, the plm-utils models do so based on the amino-acid sequence of each transcript's longest putative ORF. In contrast, the RNAsamba models do so based on the full nucleotide sequence of the transcript itself. Second, the plm-utils models incorporate a correction for class imbalance (unequal numbers of coding and non-coding transcripts in the training data) by using a balanced class weight in the random forest classifier. This ensures that both classes are treated equally despite their unequal proportions. The RNAsamba models don't include this correction. Because many species contain relatively few coding transcripts whose longest ORF is an sORF, this difference likely partially explains the difference in performance we observed between plm-utils and RNAsamba models trained only on transcripts whose longest putative ORF was an sORF. In addition, embedding sORFs using ESM allows plm-utils to take advantage of information in a larger corpus of protein sequences, even when there are very few input sequences.

Plm-utils generally predicts coding vs. non-coding sORFs more accurately than other tools

Next, we assessed how well plm-utils performed on a challenging prediction task compared to other tools. A recent large-scale benchmarking study identified 27,283 transcripts (16,243 coding; 11,040 non-coding) that were challenging for many tools to classify as coding **[19]**. The authors named the dataset "RNAChallenge" and observed an average accuracy of 10.8% (Figure 2). The protein-coding transcripts in this dataset are shorter than the average transcript: approximately 80% of the ORFs on the protein-coding transcripts were less than 300 nucleotides long, highlighting that most tools struggle with classifying sORFs as coding or non-coding.

We first built a model to predict coding versus non-coding transcripts using diverse species input. Using the same species listed in Table 1, we separated coding from non-coding transcripts. We reduced homology between our input sequences by clustering at 80% sequence identity using MMseqs2 (version 15.6f452) **[27]**. We then used plm-utils to translate sequences, limiting to sORFs by filtering to transcripts with a maximum predicted ORF of < 100 amino acids. We then embedded these sequences and trained a model. We ran the plm-utils model on the RNAChallenge dataset and calculated the performance (Figure 2). The F1 score, a metric that balances precision (the accuracy of positive predictions) and recall (the ability to identify all actual positive cases), is the highest for plm-utils. However, the RNAChallenge dataset contains some sequences that are highly similar to some sequences that we used to train the plm-utils model. While this was also true for models and tools evaluated by the benchmark, we wanted to control for this in our evaluation. We therefore removed sequences from RNAChallenge that were at least 80% similar to sequences used during training. This reduced the RNAChallenge dataset to 16,180 sequences (8,847 coding; 7,333 non-coding). Evaluating the performance on this dataset, the F1 Score decreased by ~6%. Plm-utils still outperformed all but two tools covered in the benchmarking paper, longdist and NCResNet (Figure 2) **[28][29]**. Both of these tools performed poorly on other benchmarks that assessed their ability to predict coding vs. non-coding transcripts in non-human species (regardless of transcript or ORF length) **[19]**. This likely indicates over- or under-fitting to the RNAChallenge dataset and an inability to generalize well across diverse biological datasets **[19]**. While we haven't compared directly, we expect plm-utils to perform better across species and sequencing contexts.

Taken together, we find that a simple classifier built atop the ESM2 large protein language model improves the classification of sORFs as coding or non-coding. However, the overall performance of plm-utils on the RNAChallenge dataset is still low (~33% accurate). This dataset is enriched in sORFs, highlighting that there's still room for improvement on this classification task. We think using ESM embeddings captures information about the secondary structure of proteins, as large protein language models have previously been shown to have high accuracy on structural predictions for some peptides **[30]**. This may mean this method struggles with very short peptides or peptides with certain structural features. Indeed, evidence of functional sORFs in the 3–15 amino acid length range suggests we may be missing this class of sORFs **[6]**.

View the workflow code we used to build the plm-utils classification model and evaluate its performance on RNAChallenge.

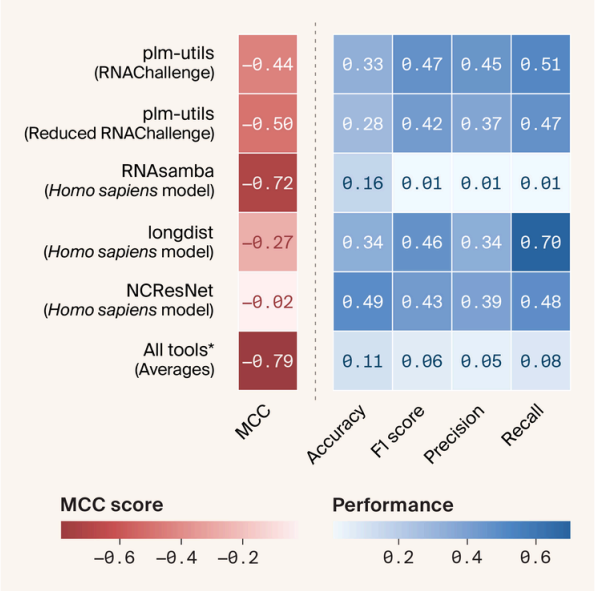

Figure 2

Performance of plm-utils and other tools on the RNAChallenge dataset.

We produced the two rows reporting on plm-utils while the rest of the rows reporting on specific tools are copied from Singh and Roy. We calculated the "All tools*" row by averaging all values in Table 1 of Singh and Roy; note that this is *not* an average for values in this figure, but for the 58 tool and model combinations evaluated in an independent benchmarking paper.

Additional validation

We built a pipeline, peptigate, that predicts and annotates peptides from transcriptomes **[31]**. Peptigate uses plm-utils to predict sORF-encoded peptides. While evaluating this pipeline, we ran plm-utils on the human transcriptome to identify sORF-encoded peptides. We compared these results against orthogonal datasets like databases of known peptides, ribosomal profiling, and strength of translation initiation site sequences. We found orthogonal support for 22% of plm-utils predictions and didn't detect any false positives (true non-coding sequences predicted to be coding). For more insights on plm-utils outputs and predictions, view those results here.

Methods

We used ChatGPT and GitHub Copilot to help write, clean up, and add comments to our code. We also used ChatGPT to suggest wording ideas and then chose which small phrases or sentence structure ideas to use.

Key takeaways

> The **plm-utils Python package** is available in this GitHub repository.

- The plm-utils Python package encodes a set of helper functions for working with protein language models. It currently only works with Evolutionary Scale Model (ESM), a protein language model trained on millions of protein sequences. It has functions to embed sequences in ESM2, train a binary classifier using labeled protein sequences, and predict the classification of new proteins using that model.
- Our first use case for plm-utils is predicting whether a transcript is coding or non-coding. We used plm-utils to predict coding versus non-coding transcripts in general, as well as when the transcript encodes an sORF. sORFs are small open reading frames of less than 300 bases that sometimes encode peptides.
- In a set of diverse research organisms, plm-utils is outperformed by the state-of-the-art tool RNAsamba for the general task of predicting coding versus non-coding transcripts [plm-utils Matthew's correlation coefficient (MCC) = 0.43; RNAsamba

MCC = 0.51]. However, plm-utils significantly improves prediction when the transcript encodes an sORF (plm-utils MCC = 0.52; RNAsamba MCC = 0.15). This is likely due, in part, to latent information captured by protein language models.

- Plm-utils also improves the prediction accuracy (33%) on a challenging dataset, RNAChallenge, over most tools (average 11%).

Next steps

While plm-utils improves prediction accuracy over most tools, predicting the coding potential of short sequences (< 100 amino acids) remains challenging. We would love feedback or ideas on how to improve accuracy in this task, with or without using protein language models.

We have several ideas to potentially improve accuracy:

1. **Using larger models**: We currently use the smallest model (esm2_t6_8M_UR50D), but there are larger models available (esm2_t48_15B_UR50D, esm2_t36_3B_UR50D, esm2_t33_650M_UR50D, esm2_t30_150M_UR50D, esm2_t12_35M_UR50D). Other prediction tasks on peptides haven't seen improved accuracy with larger ESM models **[2]**. In preliminary testing, we didn't see an improvement in accuracy for sORF coding prediction, but this should be more extensively tested and validated.
2. **Exploring ESM3**: The newly released ESM3 model **[32]** may offer potential improvements. ESM2 was trained on 49.9 million protein sequences from UniRef50 **[33][34]**, while ESM3 was trained on 2.78 billion protein sequences **[32]**. These new sequences may improve ESM's ability to encode information about short sequences.
3. **Refining training data sources**: We used Ensembl for labeled training data (coding vs. non-coding transcripts). Some transcripts initially labeled as non-coding are later found to encode sORFs **[35][36][37][38][39][40][41][42][43]**. Building models that only include validated coding and non-coding transcripts from diverse sources could improve model accuracy. However, this type of curation task would likely take a substantial amount of time.

In the meantime, we plan to use plm-utils to identify sORF-encoded peptides in transcriptome assemblies using the peptigate pipeline **[31]**.

References

1. Teufel F, Refsgaard JC, Madsen CT, Stahlhut C, Grønborg M, Winther O, Madsen D. (2023). DeepPeptide predicts cleaved peptides in proteins using conditional random fields. https://doi.org/10.1093/bioinformatics/btad616

2. Fernandez-Diaz R, Cossio-Pérez R, Agoni C, Lam HT, Lopez V, Shields DC. (2023). AutoPeptideML: A study on how to build more trustworthy peptide bioactivity predictors. https://doi.org/10.1101/2023.11.13.566825

3. Kute PM, Soukarieh O, Tjeldnes H, Trégouët D-A, Valen E. (2022). Small Open Reading Frames, How to Find Them and Determine Their Function. https://doi.org/10.3389/fgene.2021.796060

4. Andrews SJ, Rothnagel JA. (2014). Emerging evidence for functional peptides encoded by short open reading frames. https://doi.org/10.1038/nrg3520

5. Yeasmin F, Yada T, Akimitsu N. (2018). Micropeptides Encoded in Transcripts Previously Identified as Long Noncoding RNAs: A New Chapter in Transcriptomics and Proteomics. https://doi.org/10.3389/fgene.2018.00144

6. Sandmann C-L, Schulz JF, Ruiz-Orera J, Kirchner M, Ziehm M, Adami E, Marczenke M, Christ A, Liebe N, Greiner J, Schoenenberger A, Muecke MB, Liang N, Moritz RL, Sun Z, Deutsch EW, Gotthardt M, Mudge JM, Prensner JR, Willnow TE, Mertins P, van Heesch S, Hubner N. (2023). Evolutionary origins and interactomes of human, young microproteins and small peptides translated from short open reading frames. https://doi.org/10.1016/j.molcel.2023.01.023

7. Wright BW, Yi Z, Weissman JS, Chen J. (2022). The dark proteome: translation from noncanonical open reading frames. https://doi.org/10.1016/j.tcb.2021.10.010

8. Cao X, Slavoff SA. (2020). Non-AUG start codons: Expanding and regulating the small and alternative ORFeome. https://doi.org/10.1016/j.yexcr.2020.111973

9 Leong AZ-X, Lee PY, Mohtar MA, Syafruddin SE, Pung Y-F, Low TY. (2022). Short open reading frames (sORFs) and microproteins: an update on their identification and validation measures. https://doi.org/10.1186/s12929-022-00802-5

10 Delcourt V, Staskevicius A, Salzet M, Fournier I, Roucou X. (2017). Small Proteins Encoded by Unannotated ORFs are Rising Stars of the Proteome, Confirming Shortcomings in Genome Annotations and Current Vision of an mRNA. https://doi.org/10.1002/pmic.201700058

11 Brůna T, Hoff KJ, Lomsadze A, Stanke M, Borodovsky M. (2021). BRAKER2: automatic eukaryotic genome annotation with GeneMark-EP+ and AUGUSTUS supported by a protein database. https://doi.org/10.1093/nargab/lqaa108

12 Lin MF, Jungreis I, Kellis M. (2011). PhyloCSF: a comparative genomics method to distinguish protein coding and non-coding regions. https://doi.org/10.1093/bioinformatics/btr209

13 Siepel A, Bejerano G, Pedersen JS, Hinrichs AS, Hou M, Rosenbloom K, Clawson H, Spieth J, Hillier LW, Richards S, Weinstock GM, Wilson RK, Gibbs RA, Kent WJ, Miller W, Haussler D. (2005). Evolutionarily conserved elements in vertebrate, insect, worm, and yeast genomes. https://doi.org/10.1101/gr.3715005

14 Bazzini AA, Johnstone TG, Christiano R, Mackowiak SD, Obermayer B, Fleming ES, Vejnar CE, Lee MT, Rajewsky N, Walther TC, Giraldez AJ. (2014). Identification of small ORFs in vertebrates using ribosome footprinting and evolutionary conservation. https://doi.org/10.1002/embj.201488411

15 Zhu M, Gribskov M. (2019). MiPepid: MicroPeptide identification tool using machine learning. https://doi.org/10.1186/s12859-019-3033-9

16 Hanada K, Akiyama K, Sakurai T, Toyoda T, Shinozaki K, Shiu S-H. (2009). sORF finder: a program package to identify small open reading frames with high coding potential. https://doi.org/10.1093/bioinformatics/btp688

17 Camargo AP, Sourkov V, Pereira GAG, Carazzolle MF. (2020). RNAsamba: neural network-based assessment of the protein-coding potential of RNA sequences. https://doi.org/10.1093/nargab/lqz024

18 Zhang Y, Jia C, Fullwood MJ, Kwoh CK. (2020). DeepCPP: a deep neural network based on nucleotide bias information and minimum distribution similarity feature selection for RNA coding potential prediction. https://doi.org/10.1093/bib/bbaa039

19 Singh D, Roy J. (2022). A large-scale benchmark study of tools for the classification of protein-coding and non-coding RNAs. https://doi.org/10.1093/nar/gkac1092

20 Bryant P, Pozzati G, Elofsson A. (2022). Improved prediction of protein-protein interactions using AlphaFold2. https://doi.org/10.1038/s41467-022-28865-w

21 Rives A, Meier J, Sercu T, Goyal S, Lin Z, Liu J, Guo D, Ott M, Zitnick CL, Ma J, Fergus R. (2021). Biological structure and function emerge from scaling unsupervised learning to 250 million protein sequences. https://doi.org/10.1073/pnas.2016239118

22 Yang Z, Zeng X, Zhao Y, Chen R. (2023). AlphaFold2 and its applications in the fields of biology and medicine. https://doi.org/10.1038/s41392-023-01381-z

23 Vitale R, Bugnon LA, Fenoy EL, Milone DH, Stegmayer G. (2024). Evaluating large language models for annotating proteins. https://doi.org/10.1093/bib/bbae177

24 Marquet C, Heinzinger M, Olenyi T, Dallago C, Erckert K, Bernhofer M, Nechaev D, Rost B. (2021). Embeddings from protein language models predict conservation and variant effects. https://doi.org/10.1007/s00439-021-02411-y

25 Singh U, Wurtele ES. (2021). orfipy: a fast and flexible tool for extracting ORFs. https://doi.org/10.1093/bioinformatics/btab090

26 Andreev DE, Loughran G, Fedorova AD, Mikhaylova MS, Shatsky IN, Baranov PV. (2022). Non-AUG translation initiation in mammals. https://doi.org/10.1186/s13059-022-02674-2

27 Steinegger M, Söding J. (2017). MMseqs2 enables sensitive protein sequence searching for the analysis of massive data sets. https://doi.org/10.1038/nbt.3988

28 Schneider HW, Raiol T, Brigido MM, Walter MEMT, Stadler PF. (2017). A Support Vector Machine based method to distinguish long non-coding RNAs from protein coding transcripts. https://doi.org/10.1186/s12864-017-4178-4

29 Yang S, Wang Y, Zhang S, Hu X, Ma Q, Tian Y. (2020). NCResNet: Noncoding Ribonucleic Acid Prediction Based on a Deep Resident Network of Ribonucleic Acid Sequences. https://doi.org/10.3389/fgene.2020.00090

30 McDonald EF, Jones T, Plate L, Meiler J, Gulsevin A. (2023). Benchmarking AlphaFold2 on peptide structure prediction. https://doi.org/10.1016/j.str.2022.11.012

31 Borges AL, Celebi FM, Cheveralls K, Chou S, Reiter T, Weiss ECP. (2024). Predicting bioactive peptides from transcriptome assemblies with the peptigate workflow. https://doi.org/10.57844/ARCADIA-6500-9BE8

32 Hayes T, Rao R, Akin H, Sofroniew NJ, Oktay D, Lin Z, Verkuil R, Tran VQ, Deaton J, Wiggert M, Badkundri R, Shafkat I, Gong J, Derry A, Molina RS, Thomas N, Khan Y,

Mishra C, Kim C, Bartie LJ, Nemeth M, Hsu PD, Sercu T, Candido S, Rives A. (2024). Simulating 500 million years of evolution with a language model. https://doi.org/10.1101/2024.07.01.600583

33 Lin Z, Akin H, Rao R, Hie B, Zhu Z, Lu W, Smetanin N, Verkuil R, Kabeli O, Shmueli Y, dos Santos Costa A, Fazel-Zarandi M, Sercu T, Candido S, Rives A. (2023). Evolutionary-scale prediction of atomic-level protein structure with a language model. https://doi.org/10.1126/science.ade2574

34 Consortium TU. (n.d.). UniProt: the universal protein knowledgebase in 2021. https://doi.org/10.1093/nar/gkaa1100

35 Anderson DM, Anderson KM, Chang C-L, Makarewich CA, Nelson BR, McAnally JR, Kasaragod P, Shelton JM, Liou J, Bassel-Duby R, Olson EN. (2015). A Micropeptide Encoded by a Putative Long Noncoding RNA Regulates Muscle Performance. https://doi.org/10.1016/j.cell.2015.01.009

36 Jackson R, Kroehling L, Khitun A, Bailis W, Jarret A, York AG, Khan OM, Brewer JR, Skadow MH, Duizer C, Harman CCD, Chang L, Bielecki P, Solis AG, Steach HR, Slavoff S, Flavell RA. (2018). The translation of non-canonical open reading frames controls mucosal immunity. https://doi.org/10.1038/s41586-018-0794-7

37 Tajbakhsh S. (2017). lncRNA-Encoded Polypeptide SPAR(s) with mTORC1 to Regulate Skeletal Muscle Regeneration. https://doi.org/10.1016/j.stem.2017.03.016

38 Nelson BR, Makarewich CA, Anderson DM, Winders BR, Troupes CD, Wu F, Reese AL, McAnally JR, Chen X, Kavalali ET, Cannon SC, Houser SR, Bassel-Duby R, Olson EN. (2016). A peptide encoded by a transcript annotated as long noncoding RNA enhances SERCA activity in muscle. https://doi.org/10.1126/science.aad4076

39 Min K-W, Davila S, Zealy RW, Lloyd LT, Lee IY, Lee R, Roh KH, Jung A, Jemielity J, Choi E-J, Chang JH, Yoon J-H. (2017). eIF4E phosphorylation by MST1 reduces translation of a subset of mRNAs, but increases lncRNA translation. https://doi.org/10.1016/j.bbagrm.2017.05.002

40 Huang J-Z, Chen M, Chen D, Gao X-C, Zhu S, Huang H, Hu M, Zhu H, Yan G-R. (2017). A Peptide Encoded by a Putative lncRNA HOXB-AS3 Suppresses Colon Cancer Growth. https://doi.org/10.1016/j.molcel.2017.09.015

41 Wu S, Zhang L, Deng J, Guo B, Li F, Wang Y, Wu R, Zhang S, Lu J, Zhou Y. (2020). A Novel Micropeptide Encoded by Y-Linked LINC00278 Links Cigarette Smoking and AR Signaling in Male Esophageal Squamous Cell Carcinoma. https://doi.org/10.1158/0008-5472.can-19-3440

42 Guo B, Wu S, Zhu X, Zhang L, Deng J, Li F, Wang Y, Zhang S, Wu R, Lu J, Zhou Y. (2019). Micropeptide CIP2A-BP encoded by LINC00665 inhibits triple-negative breast cancer progression. https://doi.org/10.15252/embj.2019102190

43 Niu L, Lou F, Sun Y, Sun L, Cai X, Liu Z, Zhou H, Wang Hong, Wang Z, Bai J, Yin Q, Zhang J, Chen L, Peng D, Xu Z, Gao Y, Tang S, Fan L, Wang Honglin. (2020). A micropeptide encoded by lncRNA MIR155HG suppresses autoimmune inflammation via modulating antigen presentation. https://doi.org/10.1126/sciadv.aaz2059